COMPUTATION OF CURVATURE IN SEISMIC DATA

by

REBEKKA MØRKEN VALDMANIS

THESIS

for the degree of

MASTER OF SCIENCE

(Master i Anvendt matematikk og mekanikk)

Faculty of Mathematics and Natural Sciences
University of Oslo

April 2013

Det matematisk- naturvitenskapelige fakultet
Universitetet i Oslo

Abstract

Digital analysis of seismic images is a key component in the automation of seismic data interpretation. Other authors have explored using curvature as an attribute in this kind of analysis, but only on data which has been manually interpreted.

In this project the subject is the use of curvature as a feature for automatic highlighting of saltdomes in seismic images. Saltdomes are of particular interest in seismic exploration, as they are often linked to hydrocarbon finds. By using the so-called dip to extract discrete curve segments representing the seismic horizons in the image, the local curvature may be estimated. This in turn requires approximation of first and second order derivatives. Derivative approximation is done by locally fitting a parabola to each data point, and approximating the derivatives in that point by the derivatives of the parabola.

The presented method is applied to a set of inline seismic saltdome images. The results show the saltdomes clearly highlighted from their surroundings, with distinct areas of positive and negative curvature values.

Acknowledgements

First, I would like to thank my supervisor Anne H. S. Solberg for her guidance, patience and encouragement throughout the work with this thesis.

I would also like to thank my fellow students in B1002 for their company and encouragement, and for helpful input on my work.

A particular thank you to my father, Knut Mørken, for his invaluable advice and support. Thank you for reminding me to keep the overall perspective of my work in mind, and not get lost in the details.

Last, but certainly not least, thank you to my wonderful husband Fredrik for his constant love, encouragement and support.

Rebekka Mørken Valdmanis
Oslo 2013

Contents

1. Introduction

Automation of manual tasks constitutes one of the most important developments in modern society. Throughout human history, mechanical devices and machines have replaced human labour to an increasing extent. Recently, digital image analysis has been an important component in this evolution. Fundamentally, digital image analysis involves teaching computers to extract information from digital images, much in the way human eyes would. However, teaching a computer to perceive an image with the same knowledge and intuition as a human being is an extremely complex task. Therefore, the algorithms used in image analysis are usually tailored to a specific problem or group of problems.

One such problem is interpretation[1] of seismic images. To my knowledge, interpretation is currently performed manually by geologists. Because this is a very time consuming and expensive task, developing methods for automatic interpretation is a research field of interest.

The seismic data used in this project is from the Diskos repository, a Norwegian common national data repository for oil exploration and production.

1.1. Problem statement

In this project the estimation of *curvature* in seismic images, in the context of the specific task of highlighting salt structures, is explored. Salt structures are of particular interest because they are often linked to hydrocarbon finds. Previous work on curvature in seismic data addresses manually interpreted data, leaving much room for improvement in terms of automation. The method that will be presented in this project is fully automatic.

The problem statement for this project is as follows:

Develop a method for highlighting saltdomes in inline seismic images using dip and curvature as attributes. Explore how dip can be used to identify seismic horizons, and the suitability of curvature as a feature for saltdome highlighting.

Curvature is a mathematical feature that describes the shape of curves or surfaces. The main challenge in estimating curvature in a seismic image is represent-

1 *Interpretation* in the context of seismic images means identifying the sedimentary layers and other interesting structures in the data.

ing the image in terms of curves. To this end, the dip feature is used to extract curve segments that follow the shape of the seismic horizons. The image curvature can then be estimated as the curvature of these curve segments.

In order to reduce the complexity in computation and visualisation, the work in this project is done on two-dimensional sections of the three-dimensional seismic data.

1.2. Project structure

This project is presented in two parts. Part I covers seismic background material and establishes the mathematical concepts and tools needed in the rest of the project. Part II describes the method developed followed by a study of the results.

Part I.

Background & Mathematical Theory

2. Introduction to seismic imaging

The earth consists of layers of sediments such as rocks, clay and minerals. Exploring these structures is of particular interest in the oil and gas industry. By gathering seismic data, images of the Earth's crust can be reconstructed and inspected in order to find areas of interest.

The following sections give a short introduction to the process of seismic data acquisition, and are based on information collected from *GeoCLASS* [25] and *Schlumberger Oilfield Glossary* [26].

2.1. Seismic acquisition

Seismic data acquisition is performed by sending energy waves into the earth. When a wave hits a stratigraphic layer, or sedimentary bed, it is refracted. Part of the wave is reflected and travels back to the surface, while the rest of the energy continues to travel into the earth until it hits another layer and is refracted again. The reflected waves are recorded by receivers. The amplitudes of the recorded signals can be used to create images of the subsurface structure.

The process described above is the basic principle of seismic acquisition. In marine seismics, data acquisition is usually performed by large ships that traverse the sea-surface in a row-by-row manner. The ships tow *streamers*, which are large cables that may be several kilometers long. The streamers hold the source- and receiver instruments, which are usually *airguns* and *hydrophones*. An airgun sends

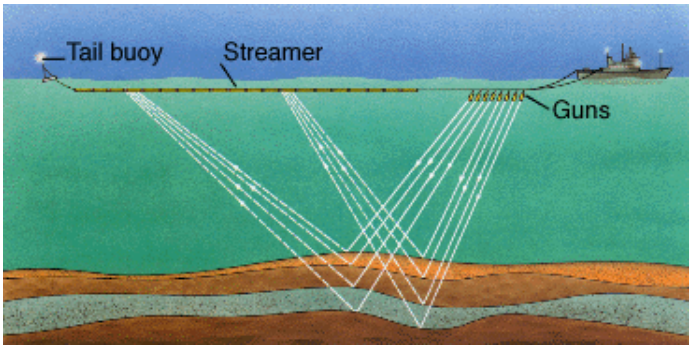


Figure 2.1.: Illustration of seismic surveying using streamers [26].

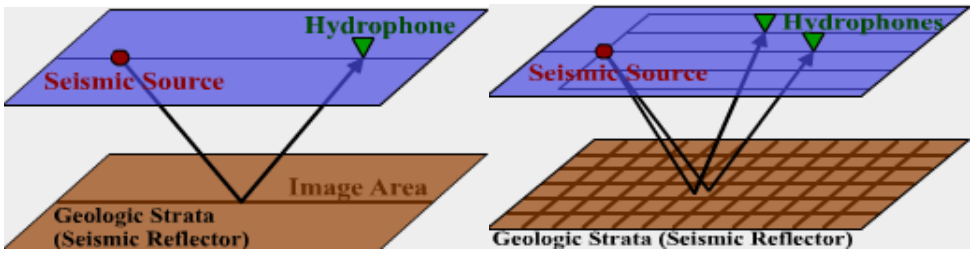


Figure 2.2.: (a) 2D and (b) 3D acquisition [25].

a signal in the form of highly pressurised air into the sea, and the signal is reflected by the layers in the seabed, as described above. The returning signal is recorded by hydrophones, which are the receiver part of the streamers. An illustration of this process can be found in Figure 2.1

Other forms of marine seismic acquisition do exist. One is ocean-bottom seismics, where either the source or the receiver (or both) are placed on the ocean bottom, rather than towed along the sea surface. Another variation is down-hole seismics, where the instruments are placed below the ocean-bottom, in wells.

2.2. 2D and 3D imaging

As described in the previous section, the data used to create seismic images is collected by streamers. Using a single streamer would result in a two-dimensional image of a vertical seabed-slice directly below the streamer. The only returning signals that will be recorded by the receivers are the ones that travel along the vertical plane directly below the streamer. Signals that are reflected in other directions will be lost. The advantage of this 2D acquisition method is of course that less of the costly surveying equipment is required, making it cheaper. However, although more expensive and complex, three-dimensional surveying is much more efficient and widespread.

In the three-dimensional case, multiple streamers are dragged by each ship. The streamers are placed in parallel, so they form a matrix of sources and receivers. This way, signals that are shot or reflected at an angle will be recorded as well as the ones recorded in the two-dimensional case. The result is that a large area is imaged, rather than a single line as in two-dimensional surveying. Figure 2.2 gives an illustration of these two methods.

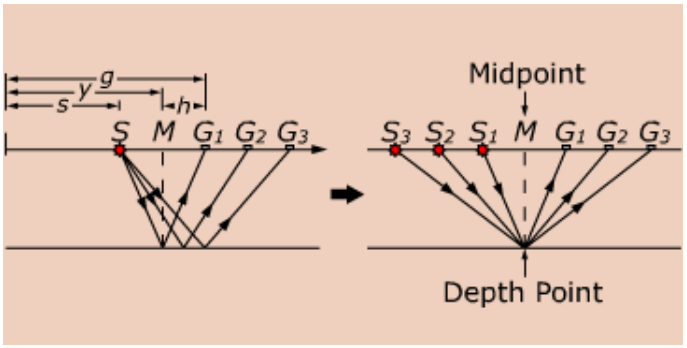


Figure 2.3.: Illustration of traces with common midpoint [26].

2.3. Processing

In order to obtain a satisfactory visualization of the seismic data once it is collected, it needs to be processed. Problems that may need to be tackled are aliasing, multiples and noise, which can all be corrected by various processing methods.

Before studying some of these processing methods, an important concept in seismics must be established. A seismic *trace* is the data corresponding to a single signal journey, from the source, through the sea-floor layers, and finally back to the receiver. In other words, a trace is a data observation that belongs to a single source-receiver pair.

Every trace has a midpoint, which is the point where the signal is reflected. During processing, the traces are grouped according to their midpoint, see Figure 2.3. Traces with the same midpoint are grouped together, because they describe the same spot in the seabed. Once the traces are grouped by common midpoint the groups can be *stacked*. Stacking is the process of summing all traces with common midpoint. This is an effective way of reducing noise in the data. The noise present will have both positive and negative values, and when the traces are added some of the noise from one trace will cancel some of noise from another trace. The result is a reduction in noise in the signals. This concept is illustrated in Figure 2.4. The final result, after stacking the traces in a common midpoint group, is an image of a vertical line through the seabed. All the midpoint stacks can then be put together to create a three-dimensional cube which depicts a piece of the seabed.

Another processing step that may be performed is correcting for signal offset. When the seismic signals are both sent into the earth and reflected back at an angle, there is an offset compared to if the signal had hit the earth at a 90° angle. This means that common midpoint signals may have very different source-receiver travel distances. The offset concept is illustrated in Figure 2.5. Signal offset is corrected during processing, so all signals have zero offset.

Other processing methods which may be performed are correction for the

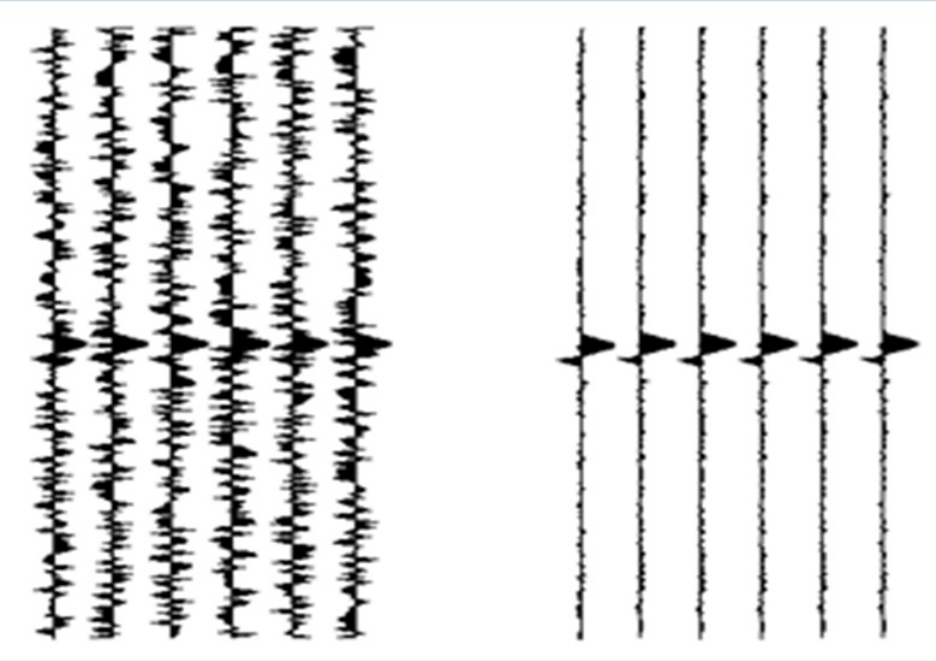


Figure 2.4.: Illustration of stacking, where traces with common midpoint are summed resulting in noise reduction. The noisy signals before stacking are to the left, and the resulting post-stacked signals are to right [27].

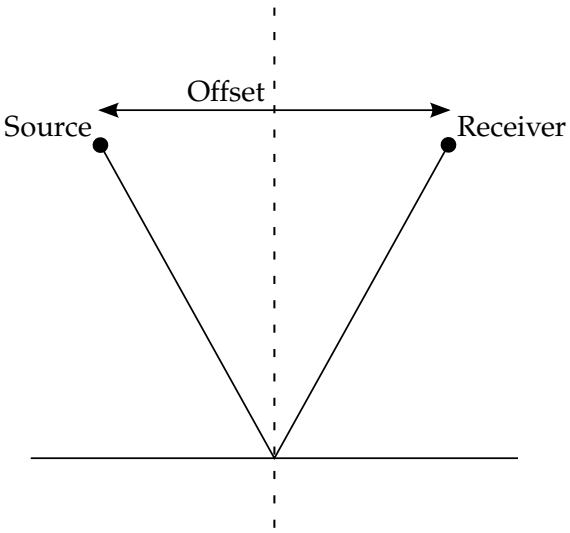


Figure 2.5.: An offset may occur when the signal travels from source to receiver.

depth of the source and receiver, correction for amplitude losses, and noise reduction, which is performed by filtering the signals.

2.4. Inlines, crosslines and time slices

After processing, the seismic data can be combined to form a three-dimensional data cube, which depicts a cube of the earth. Three-dimensional data is computationally heavy to work with and not as easy to display as two-dimensional images are. It is common to instead work on two-dimensional sub-sections of the data cube. Of course, there are several ways the cross sections could be made. In seismics, however, they are usually horizontal or vertical.

Definition 2.1 (Inlines and crosslines)**.** *Inline* and *crossline* sections are vertical subsections in seismic images. An inline is parallel to the direction in which the data was acquired. Crosslines are lines perpendicular to the inlines.

An inline or crossline image shows all the layers of sediment as thick horizontal lines. Commonly both section types are used in combination when exploring a seabed area. Figure 2.6 on the next page shows an inline seismic image. We easily see the layers of sediment in the layered structure in the image. This particular inline image has a salt body in the center, which is why the layers slope here. In an area with no salt or other structural disturbances the layers would be horizontal, as in the edges of the image.

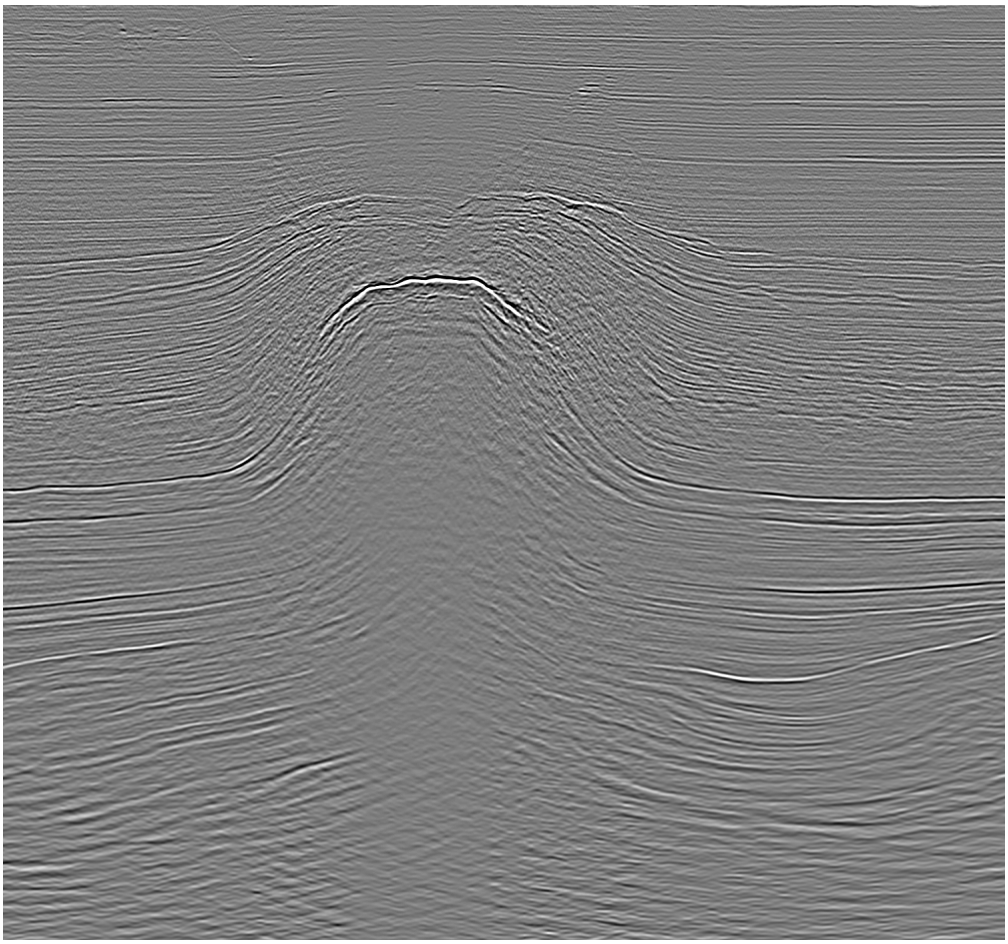


Figure 2.6.: Example of an inline seismic image.

Definition 2.2 (Time slice)**.** A *time slice* image is a horizontal cross section of the seabed.

A single time slice contains data points with common *arrival time*, the elapsed time between the source and the receiver. A time slice does not show the structure of the seabed layers like an inline section does. This is because a time slice depicts the points with common arrival time from above, and these points are not necessarily part of the same sedimentary layer. Figure 2.7 shows an example of a time slice image.

Figure 2.8 gives a visualisation of inlines, crosslines and time slices relative to each other. The inlines are horizontal and in the same direction as the seismic vessel, and the crosslines are horizontal lines perpendicular to the inlines. This makes the inline and crossline sections vertical slices of the seabed. The time slice sections on the other hand, are horizontal slices of the seabed.

2.5. Salt structures

The structure of the sediments constituting the Earth's crust has been formed over the span of different geological periods. Geological processes cause movement and deformation in the layers of sediment, which in turn result in various characteristic structures. These structures are visible in seismic images. Some structures are of particular interest because they are often associated with hydrocarbon finds. This section gives a brief description of salt structures, which are the main subject of this project. Before this, another important seismic term needs to be established.

Definition 2.3 (Horizon)**.** A *horizon* is a layer of rock or sediment in the seabed.

Salt in the seabed pushes upwards, piercing or changing the structure of the above sediment layers. The salt usually forms a dome- or mushroom-like shape, depending on how thick the layers above it are and the salts sedimentation rate [1]. During the process the horizons are pushed upward, and their structure is changed. In the area around the salt the horizons will slope steeply. Sometimes they are broken, resulting in faults.[1] Because of their characteristic shape, these salt structures are often referred to as *saltdomes*. Figure 2.9 on page 22 shows an illustration of a saltdome. The salt has the characteristic dome-shape, and the layers of sediment have been pushed and bent in the area surrounding the salt. The illustration also includes a trap where oil is trapped against the side of the salt body.

Another characteristic feature of a saltdome is that its shape is locally *isotropic* [16], meaning that the shape is roughly the same in any direction regardless of the orientation.

1 A fault is a break or planar surface in rock across which there is observable displacement [26].

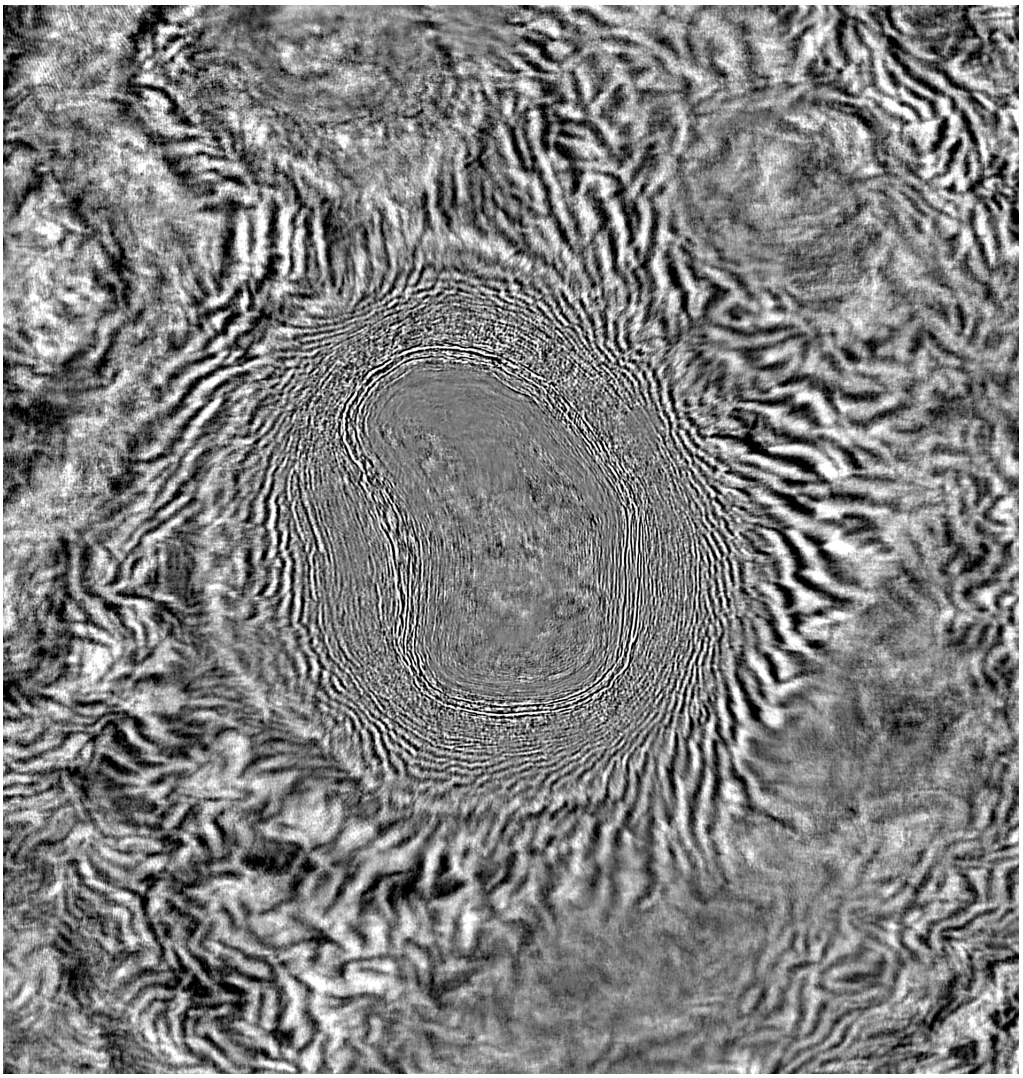


Figure 2.7.: Example of a time slice image.

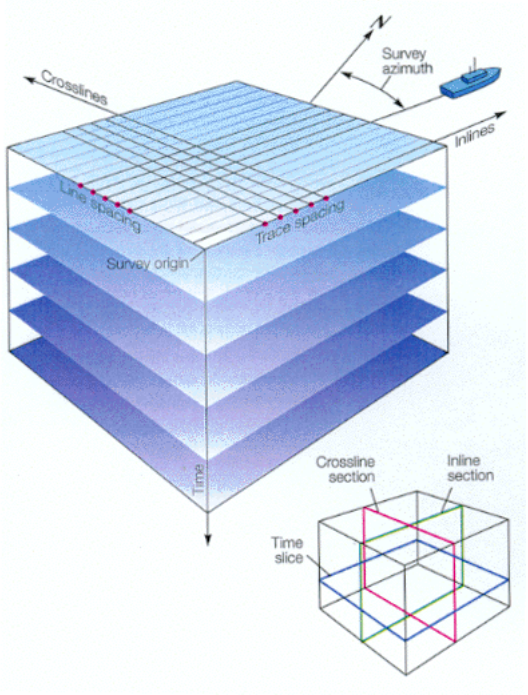


Figure 2.8.: Illustration of crosslines, inlines and a time slice [26].

The texture within a saltdome is radically different to that of the surrounding area. The surrounding horizons have an even, layered structure and a high graylevel variability in the perpendicular direction. In seismic, the layers with high graylevel variability in the vertical direction are referred to as *strong reflectors*. In contrast to the horizontally layered areas, the are within the saltdome is somewhat chaotic, with an incoherent pattern and low variability in contrast and graylevel.

Now that the most important background concepts in seismic have been established, we are ready to go into more specifics for this project. First, we will look at the data we will be working with. The following chapter gives a brief geologic description of the dataset. Following that, in Chapter 4, we will define the seismic concept *dip*, and look at ways of estimating it.

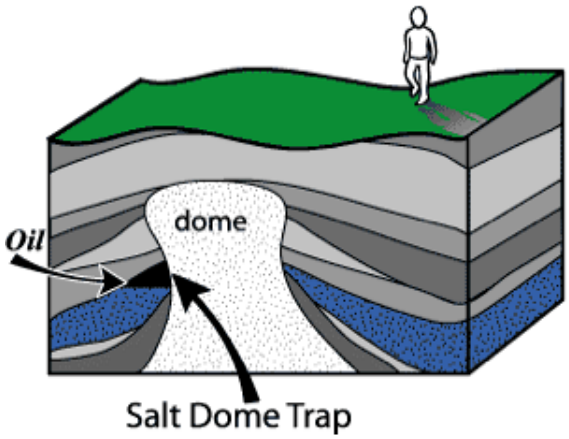


Figure 2.9.: Illustration of a saltdome.
This illustration is from `www.cartografareilpresente.org/article132.html`.

3. The dataset

The following geological description of the North Sea dataset, including the figure, is from Berthelot et al. [2], who used the same dataset in their work.

A subset of a 3D dataset from Central graben (Figure 3.1) is used in this study. The area includes a salt diapir and a well 1/6-5 which was used to verify the top salt. In the well 1/6-5, the Cenozoic succession overlies approximately 25 meters of Ekofisk chalk deposits of Maastrichtian age, which again overlies Zechstein salt of Late Permian age. In the area around the salt diapir a complete Cenozoic succession is present, indicating that the chalk located at the top of the structures was pushed upwards through the stratigraphy in front of the salt during diapirism. The boundary between the Chalk group and the Zechstein salt, as seen at the well site of 1/6-5, coincides with an amplitude anomaly in the seismic data. This amplitude anomaly was therefore interpreted as the top salt seismic reflector. Sedimentary rocks are easily recognized by parallel to sub-parallel seismic reflectors, which are horizontal to sub-horizontal in the area away from the salt diapir and dipping upward alongside it. These upward dipping seismic reflectors represent sedimentary layers which have been re-oriented from a sub-horizontal orientation during deposition to become upward dipping alongside it during diapirism and successive cutting of the sedimentary layers. Sedimentary rocks are interpreted towards the salt diapir as close as dipping seismic reflectors are observed. However, the boundary between the sedimentary rocks and the salt itself is not observed directly in the seismic reflection data.

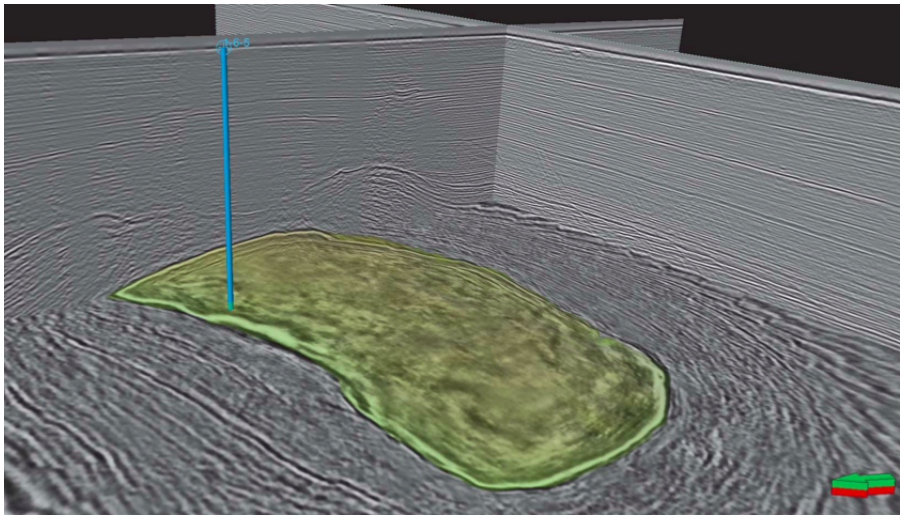


Figure 3.1.: Subset of a North Sea 3D data volume together with a well 1/6-5 used to verify the top salt (transparent area).

4. Dip

In this project we wish to explore the seismic data described in the previous chapter mathematically. In order to do so, a description of how the image pixels relate to each other is needed. One way to obtain such a description is in terms of the dip attribute. This chapter will give an introduction to what dip is, followed by a study of methods for dip estimation. First, the term dip must be defined.

Definition 4.1 (Dip)**.** The term *dip* refers to the angle at which a rock layer is inclined from the horizontal [21]. In other words, dip refers to the angle a horizon makes with the horizontal axis.

The dip angle is illustrated in Figure 4.1. Intuitively, the dip tells us how much, and in what direction, a curve deviates from being horizontal. In order to estimate dip, the image gradient is put to use.

4.1. The image gradient

Dip calculation for seismic images is often based on estimating the *gradient* of the image. The details on image gradients in this chapter are based on material from the book by Gonzalez and Woods [8].

Definition 4.2. The gradient at a point (x, y) in an image f is denoted by $\nabla f(x, y)$, and is defined as the vector

$$\nabla f(x,y) = \begin{bmatrix} g_h(x,y) \\ g_v(x,y) \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial h}(x,y) \\ \frac{\partial f}{\partial v}(x,y) \end{bmatrix},$$

where h and v denote the horizontal and vertical axes, respectively.

The gradient vector points in the direction of the greatest rate of change in f at the point (x, y). The *direction* of the gradient, θ, is the angle between the gradient vector and the horizontal axis. Figure 4.2 on page 27 illustrates the gradient vector, its components g_h and g_v, and the gradient direction, θ. The *gradient magnitude* is the length of the gradient vector. It represents the rate of change in the direction of

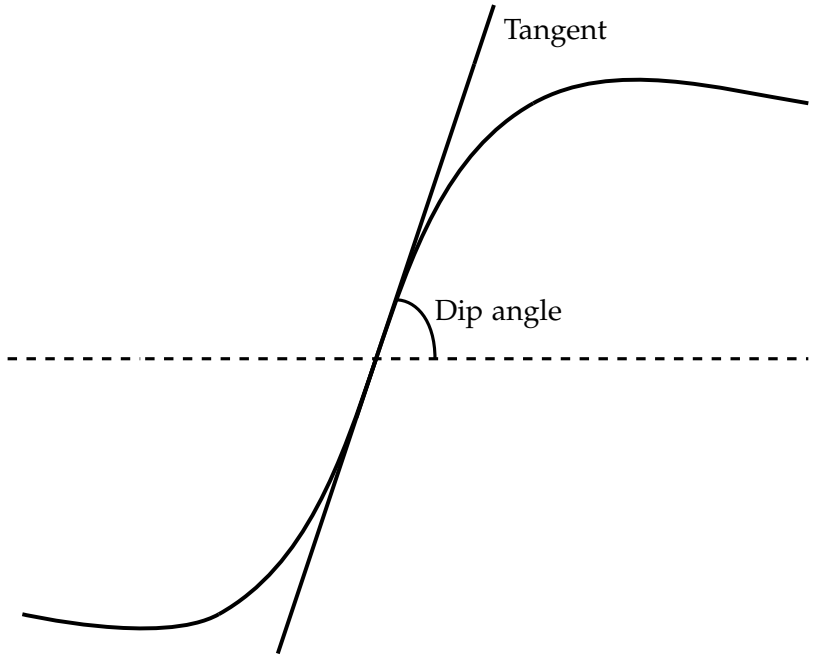


Figure 4.1.: The dip angle is the angle the tangent in a point on the horizon makes with the horizontal axis.

the gradient vector in the location(x, y), and is given by the square root of the sum of the squared gradient components,

$$M(x, y) = \|\nabla f(x, y)\|.$$

4.2. Estimating gradients

Gradient estimation is one of the key operations in image analysis, and is most commonly used in edge detection. Image gradients are usually estimated by filtering the image with a set of two gradient filter masks, one working in the horizontal and one in the vertical direction. This results in a horizontal and a vertical gradient component. Letting G_h and G_v represent the horizontal and vertical filter masks, respectively, the gradient components g_h and g_v are obtained by convolving the image I with the filter masks,

$$g_h = G_h * I,$$
$$g_v = G_v * I.$$

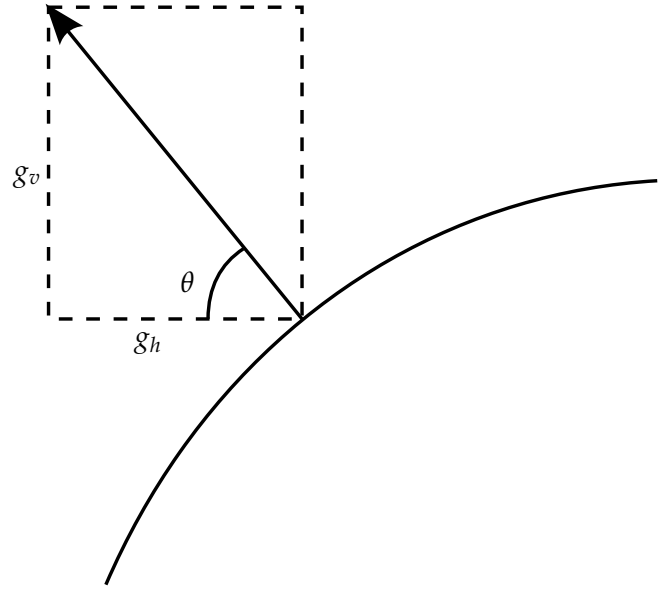


Figure 4.2.: The gradient vector points in the direction of the greatest rate of change in a point, and is decomposed in the horizontal and vertical components g_h and g_v. The direction of the gradient is given by the angle θ.

The $*$ operator represents the convolution operator. For more on image filtering and convolution, see Chapter 3 in Gonzalez and Woods [8]. We will now look at a few of the most common gradient filter masks.

We have already seen that gradients represent partial derivatives. The simple way to estimate discrete partial derivatives is by means of one-sided differences,

$$\frac{\partial f(x_i, y)}{\partial x} \approx \frac{f(x_i, y) - f(x_{i-1}, y)}{\Delta x},$$
$$\frac{\partial f(x, y_i)}{\partial y} \approx \frac{f(x, y_i) - f(x, y_{i-1})}{\Delta y}$$

Formulating the above expressions in image filter terms results in a pair of filter masks which simply take the difference between the pixel and one of its vertical and horizontal neighbours, respectively:

$$g_h(i, j) = f(i, j) - f(i, j - 1),$$
$$g_v(i, j) = f(i, j) - f(i - 1, j).$$

In image analysis it is more common to use centered, symmetric versions of these filters. The equations for the centered filter masks are

$$g_h(i, j) = f(i, j + 1) - f(i, j - 1),$$
$$g_v(i, j) = f(i + 1, j) - f(i - 1, j),$$

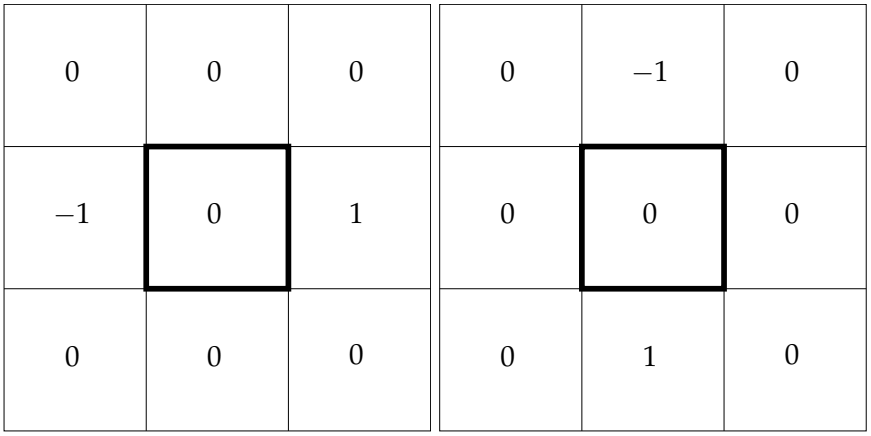


Figure 4.3.: Simple difference filter masks. Filter mask (a) results in the vertical gradient component, and filter mask (b) results in the horizontal component.

and the filters are illustrated in Figure 4.3.

Unfortunately, the difference filters described above are very sensitive to image noise. A more robust set of gradient filter masks are the *Sobel* filters in Figure 4.4. Rather than the single sided difference in the simple filter masks, these filters utilize a two-sided difference calculation. In addition the filters are two-dimensional, including more of the surrounding pixels in the gradient calculation. In the particular case of the Sobel filter, the two-sided difference calculation in one direction is combined with a smoothing operation in the other. This reduces the presence of noise in the resulting gradient estimates.

A third filter alternative is the *Derivative of Gaussian* filter, which is created by convolving a Gaussian filter kernel,

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}},$$

with the centered difference filters in Figure 4.3. Figure 4.5 shows one-dimensional Gaussian filter kernel of size 100 with standard deviation and $\sigma = 5$. Figure 4.6 shows the derivative of this Gaussian, which was obtained by convolving the filter kernel with a horizontally centered difference filter.

4.3. Calculating dip using the gradient

Once the gradient components have been calculated, they can be used to find the dip. To do so, observe that the gradient, pointing in the direction of the greatest rate of change in the image, is orthogonal to the direction with the least change.

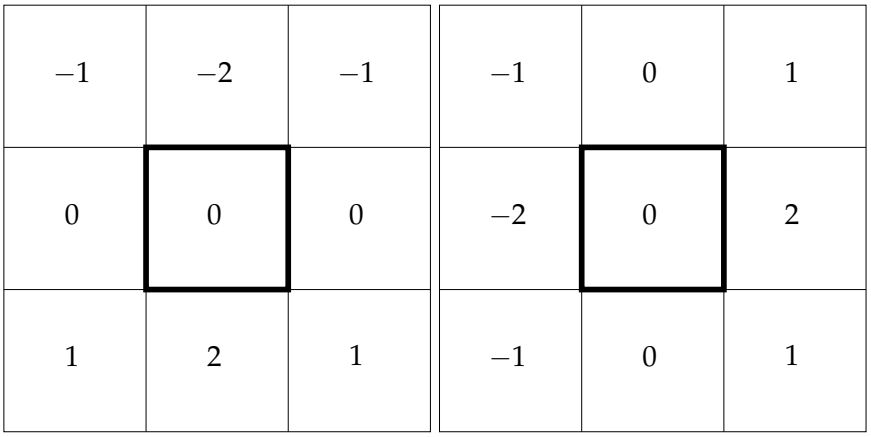


Figure 4.4.: Sobel filter masks. Mask (a) is the horizontal filter mask and mask (b) is the vertical filter mask.

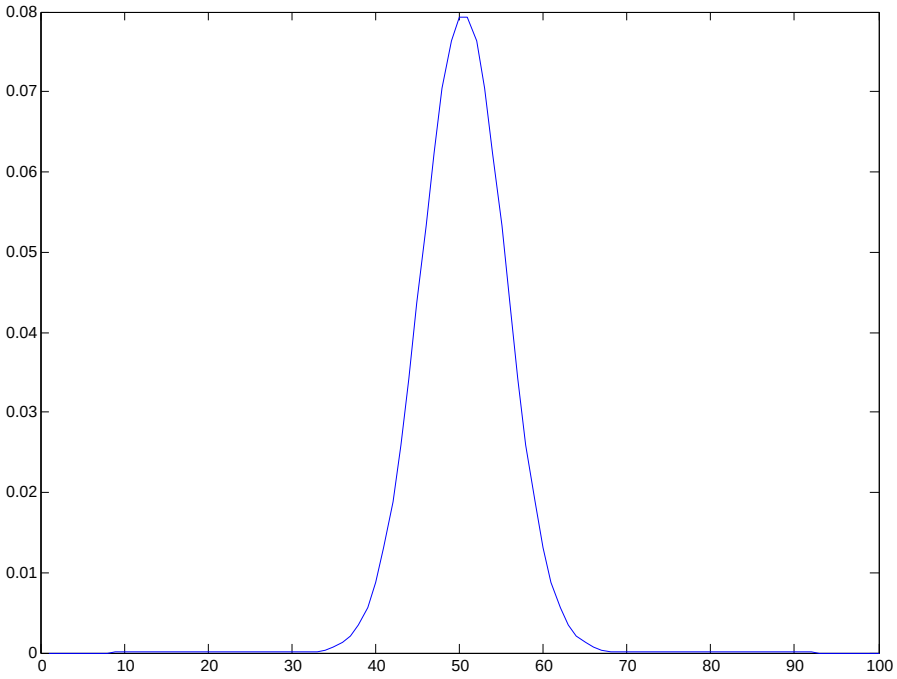


Figure 4.5.: A Gaussian filter kernel of size 100 and standard deviation 5.

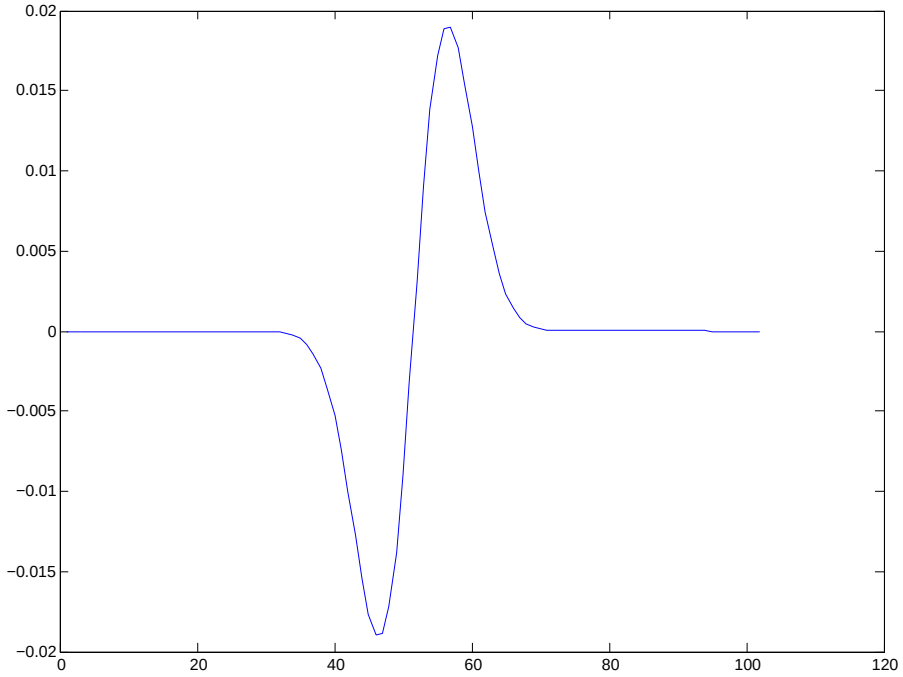


Figure 4.6.: Derivative of the Gaussian in Figure 4.5.

This direction is equivalent to the angle of the horizon tangent, and thereby the dip.

Observation 4.1. For a given image point, the direction of the gradient and the direction of the dip are orthogonal.

Figure 4.7 illustrates this observation. The figure shows the gradient vector, with the two gradient components g_h and g_v, and the gradient direction θ. The dip angle we are interested in is the angle denoted by ϕ in triangle B. Because A and B are similar triangles, we know that this angle is equal to ϕ in triangle A. Basic trigonometry leads us to the following expression for ϕ

$$\tan(\phi) = \frac{g_h}{g_v},$$

which is equivalent to

$$\phi = \arctan\left(\frac{g_h}{g_v}\right).$$

This results in the following dip estimate,

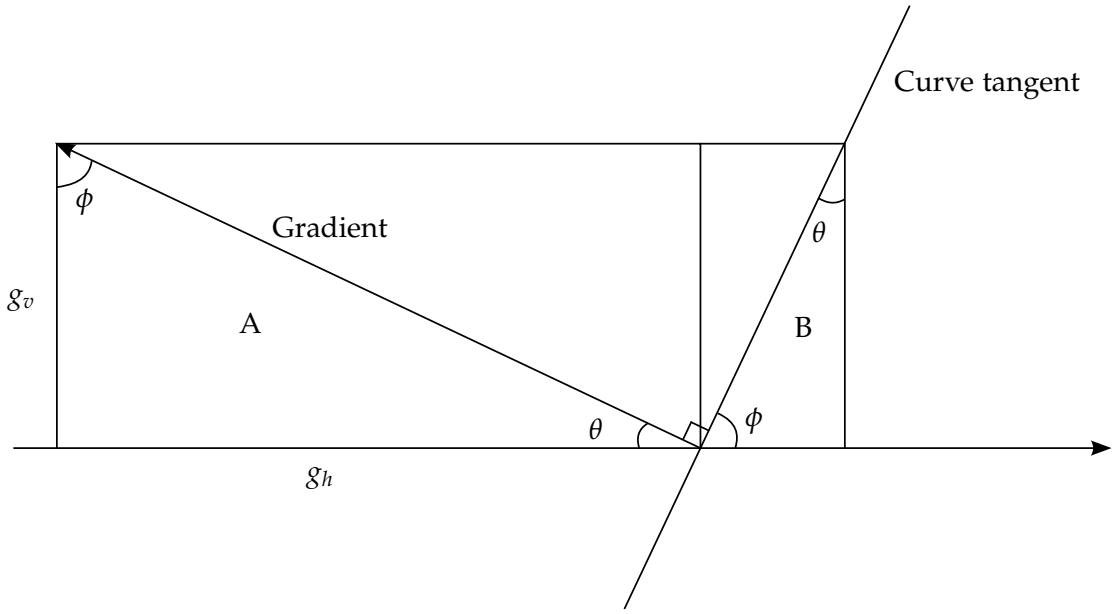


Figure 4.7.: The similarity of the triangles A and B allows us to calculate the dip, ϕ, using the gradient components g_h and g_v.

Observation 4.2 (Dip estimate)**.** Dip may be estimated as

$$\text{dip} = \arctan\left(\frac{g_h}{g_v}\right),$$

where g_h and g_v are the horizontal and vertical gradient components, respectively [1].

To summarize, we have found that a simple way to calculate dip consists of the following steps:

1. Estimate gradient components g_h and g_v by filtering image with a pair of gradient filter masks.

2. Calculate dip as $\arctan(g_h/g_v)$.

Figure 4.8 shows the result of performing the simple dip estimation on the inline image presented in Figure 2.6 on page 18. The graylevels in the image range from black to white, corresponding to the range of dip values from -90° to 90°. This means that the light area on the left side of the image has positive dip, while the darker area to the right has negative dip. The gradients here were computed with the Derivative of Gaussian filter described in the previous section, with size 15 and $\sigma = 1.5$. Although there are clear regions of positive and negative dip, the result is

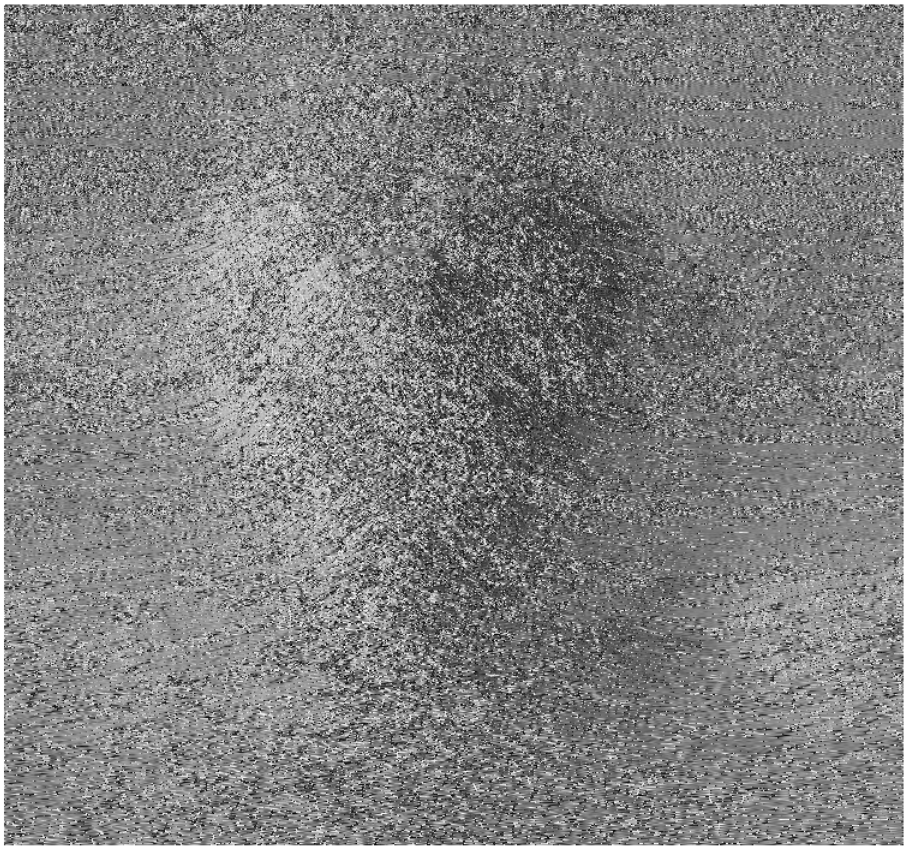


Figure 4.8.: Simple dip estimate, calculated as described in Section 4.3.

very noisy. In the next section we will look at a dip estimation method designed to reduce noise.

4.4. More robust dip estimates

In order to tackle the problem of noise sensitivity in dip estimation, Randen et al. [19] present a method consisting of the following three steps:

1. Gradient vector estimation, as explained in the previous sections, and noise reducing filtering. The gradients are estimated using a Derivative of Gaussian filter.

2. Gradient covariance matrix estimation in a local window.

3. Eigenvector decomposition of the covariance matrix. The dip is the direction of the principal eigenvector of the covariance matrix.

The article presents the method in $\mathbb{R}^3$. Applying the method in $\mathbb{R}^2$ is equivalent to the application in $\mathbb{R}^3$.

The method operates within a local window of size $N = n \times n$ centered at the current pixel. The first step is simply estimating the gradient components g_h and g_v for every pixel within the window. In statistical terms, the gradient components form the *matrix of observations*,

$$\begin{bmatrix} \boldsymbol{g}_h \\ \boldsymbol{g}_v \end{bmatrix} = \begin{bmatrix} g_h^1 & g_h^2 & \cdots & g_h^N \\ g_v^1 & g_v^2 & \cdots & g_v^N \end{bmatrix}, \tag{4.1}$$

where g_h^i and g_v^i are the horizontal and vertical gradient components for pixel i, when the pixels have been sequentially numbered from 1 to N. Step 2 involves calculating the covariance matrix of the matrix of observations. The covariance matrix is given by

$$\begin{bmatrix} \mathrm{E}[(\boldsymbol{g}_h - \mu_h)(\boldsymbol{g}_h - \mu_h)^T] & \mathrm{E}[(\boldsymbol{g}_h - \mu_h)(\boldsymbol{g}_v - \mu_v)^T] \\ \mathrm{E}[(\boldsymbol{g}_v - \mu_v)(\boldsymbol{g}_h - \mu_h)^T] & \mathrm{E}[(\boldsymbol{g}_v - \mu_v)(\boldsymbol{g}_v - \mu_v)^T] \end{bmatrix}, \tag{4.2}$$

where the observation means μ_h and μ_v are given by

$$\mu_h = \mathrm{E}[\boldsymbol{g}_h] = \frac{1}{N} \sum_{i=1}^{N} g_h^i, \tag{4.3}$$

$$\mu_v = \mathrm{E}[\boldsymbol{g}_v] = \frac{1}{N} \sum_{i=1}^{N} g_v^i. \tag{4.4}$$

The covariance matrix measures how strongly the gradient components are related to one another. Once the covariance matrix has been calculated, its eigenvectors and eigenvalues are computed. The eigenvector corresponding to the largest of the eigenvalues, the *first principal component*, points in the dominating direction of the gradient components. This direction is equivalent to the dip. Randen et al. also include a reliability measure of the dip estimate, based on the eigenvalues of the covariance matrix.

Figure 4.9 presents the result of applying the above dip estimation method to the same inline image as before, again using a Derivative of Gaussian filter of size 15 with $\sigma = 1.5$. Although there is still some noise present, this is clearly an improvement from the result presented in the previous section. This method will be used for dip estimation throughout this project.

In the next chapter we will leave seismics for a while and move on to studying mathematic curves.

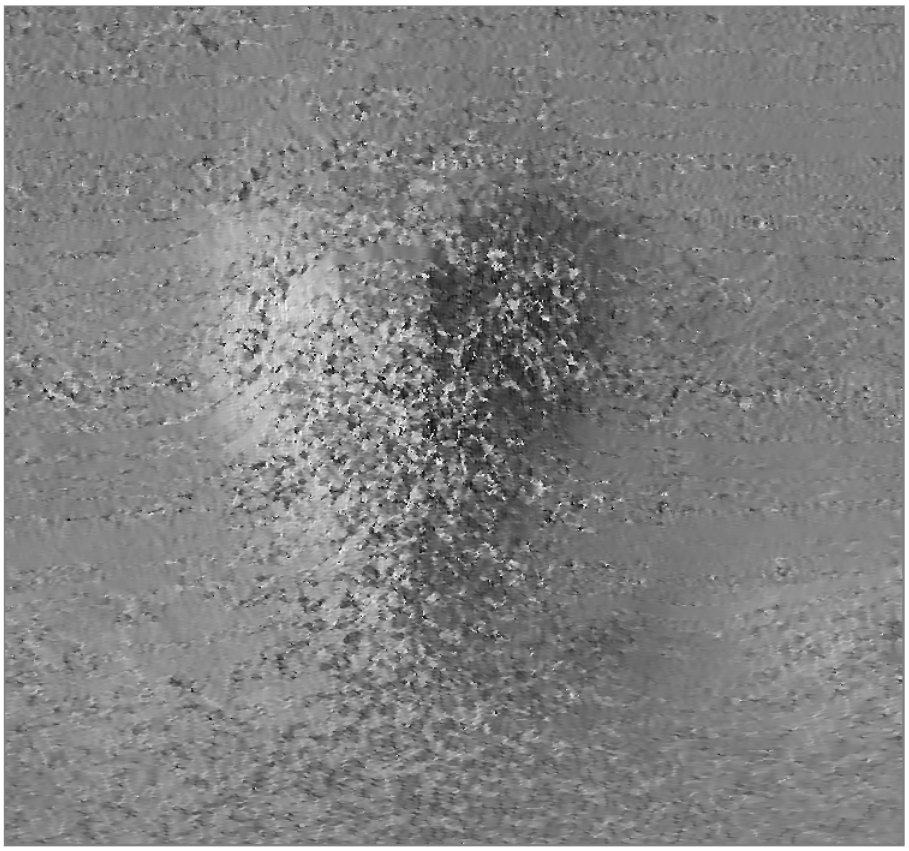


Figure 4.9.: Dip estimation as presented by Randen et al. [19].

5. Curvature

Curvature is a geometric property of a curve which measures how a curve bends. It is one of the most characteristic properties of a curve [13].

This chapter is based on *Curvature at Wolfram MathWorld* [24], Haralick and Shapiro [9], *Curvature at Encyclopedia of Mathematics* [23], and the first pages of an article by Roberts [20] which discusses the use of curvature as an attribute in seismics.

5.1. What is curvature?

Simply put, *curvature* describes how a curve bends, in terms of how much it deviates from being straight, or flat. Given a particular point on the curve, the curvature is a measure of the rate of change of the direction of the curve.

Curvature can be defined for curves in both $\mathbb{R}^2$ and $\mathbb{R}^3$. As the work in this project addresses two dimensional data, the definitions in the following are given in $\mathbb{R}^2$. A short presentation of curvature in $\mathbb{R}^3$ will be given in Section 5.4.

Before we get into the mathematical details of curvature, a few concepts need to be defined. The following definitions are found in Section 1.3 of Do Carmo [4], but have been adapted from $\mathbb{R}^3$ to $\mathbb{R}^2$.

Definition 5.1 (Tangent Angle)**.** Given an interval $I = [a, b] \subset \mathbb{R}$, let $\alpha : I \to \mathbb{R}^2$ be a continuous parametrized differentiable curve given by $\alpha(t) = (x(t), y(t))$. The derivative $\alpha'(t)$ is called the *tangent vector* at t. The **tangent angle** is the angle the tangent vector makes with the horizontal axis and is denoted by ϕ.

The tangent vector and tangent angle are illustrated in Figure 5.1 on the following page. The illustration shows the tangent vector passing through the point t, which is how it is commonly imagined, although in reality all vectors start at the origin.

Definition 5.2 (Arc Length)**.** Given $t \in I$, the **arc length** of a curve $\alpha : I \to \mathbb{R}^2$ from the point t_0 is

$$s(t) = \int_{t_0}^{t} |\alpha'(t)| dt,$$

where $|\alpha'(t)| = \sqrt{(x'(t))^2 + (y'(t))^2}$.

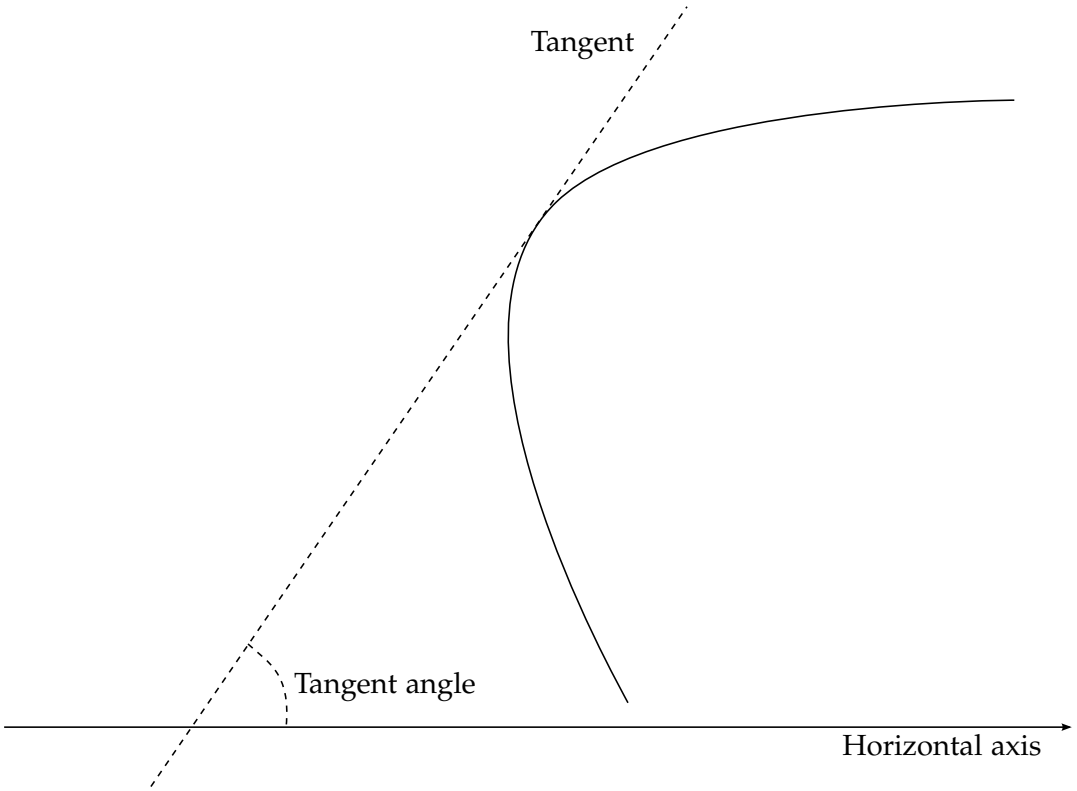


Figure 5.1.: The tangent angle is the angle the curves tangent makes with the horizontal axis.

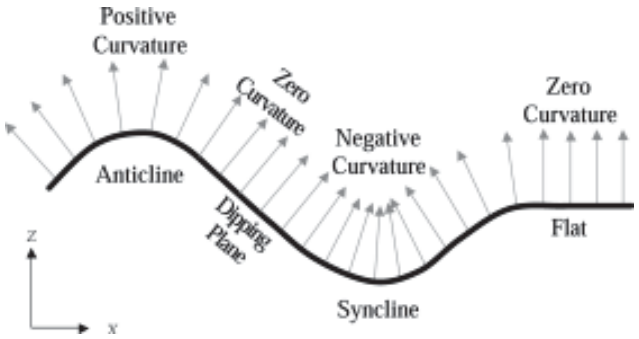


Figure 5.2.: Sign convention for curvature [20].

Given a $t \in I = [a, b]$, the function α maps t to the point $\alpha(t) = (x(t), y(t))$. The variable t is the *parameter* of α. When describing curvature, it is common to let α be parametrized by arc length, and we write $\alpha(s) = (x(s), y(s))$. However, it can be shown that the concepts in this chapter hold for curves parameterized by any parameter [4, pp. 21–22].

With the above concepts in place, curvature can be defined mathematically as follows:

Definition 5.3 (Curvature [22])**.** The curvature κ of a curve $\alpha : I \to \mathbb{R}^2$ parameterized by arc length, is defined as

$$\kappa = \frac{d\phi}{ds},$$

where ϕ and s are the tangent angle and the arc length of α, respectively.

So, curvature is defined as the rate of change of the tangent angle with respect to the arc length.

The sign of κ provides further information about the local shape of the curve. Definition 5.3 results in the same sign convention as used by Roberts [20], which is illustrated in Figure 5.2. As shown here, a concave segment of the curve has positive curvature, whilst the convex segment has negative curvature. Flat areas have zero curvature. This is the sign convention used throughout this project.

5.2. Deriving an expression for curvature

Definition 5.3 defines curvature in terms of the tangent angle and arc length. However, curvature may also be expressed in terms of other parameters. In this section, an expression for the curvature of a general parameterized curve will be derived.

When the curve α is not parameterized by arc length, but by a general parameter t, it can be expressed as $\alpha(t) = (x(t), y(t))$. When expressed in terms of t, the

derivatives in Definition 5.3 can be restated as

$$\kappa = \frac{d\phi(t)}{dt} \Big/ \frac{ds(t)}{dt} .$$

For a simpler notation the parameter t will be omitted in the following. It will be clear from the context whether we are talking about the general function, for example ϕ, or the value of the function ϕ in the point t. We will also let $\cdot'$ denote $\frac{d\cdot}{dt}$ and $\cdot''$ denote $\frac{d^2\cdot}{dt^2}$.

We begin by looking at the derivative of the tangent angle, $\frac{d\phi}{dt}$. The tangent of ϕ can be expressed as

$$\tan\phi = \frac{y'}{x'}. \tag{5.1}$$

Taking the derivatives of both parts of equation (5.1) yields

$$\frac{d}{dt}\tan\phi = \frac{1}{\cos^2\phi}\phi'$$

and

$$\frac{d}{dt}\frac{y'}{x'} = \frac{y''x' - y'x''}{(x')^2}.$$

Equation (5.1) now becomes

$$\frac{1}{\cos^2\phi}\phi' = \frac{y''x' - y'x''}{(x')^2}.$$

This gives

$$\phi' = \cos^2\phi\frac{y''x' - y'x''}{(x')^2} \tag{5.2}$$

$$= \frac{1}{1+\tan^2\phi}\frac{y''x' - y'x''}{(x')^2} \tag{5.3}$$

$$= \frac{1}{1+\dfrac{(y')^2}{(x')^2}}\frac{y''x' - y'x''}{(x')^2} \tag{5.4}$$

$$= \frac{y''x' - y'x''}{(x')^2 + (y')^2}. \tag{5.5}$$

Next, consider the arc length. Definition 5.2 defined arc length as

$$s(t) = \int_{t_0}^{t} |\alpha'(t)|dt.$$

Assuming that $\alpha : I \to \mathbb{R}^2$ is continuous, the Fundamental theorem of calculus [14] says that

$$\frac{d}{dt}\int_{t_0}^{t} |\alpha'(t)|dt = |\alpha'(t)|,$$

given that $[t_0, t] \in I$. This allows us to take the derivative of either side of Equation 5.2, resulting in an expression for the derivative of the arc length,

$$\frac{ds}{dt} = \sqrt{x'(t)^2 + y'(t)^2}.$$

Substituting this expression and (5.5) into Definition 5.3 results in

$$\kappa = \frac{\phi'}{s'} = \frac{y''x' - y'x''}{(x')^2 + (y')^2} \Big/ \sqrt{(x')^2 + (y')^2} \tag{5.6}$$

$$= \frac{y''x' - y'x''}{\left((x')^2 + (y')^2\right)^{\frac{3}{2}}} \tag{5.7}$$

as the final expression for curvature. So, all one needs to calculate curvature are the first and second derivatives of the x and y-components of the curve.

5.3. The osculating circle

An alternative way to define curvature is by the *osculating circle*. Given a point on a curve in $\mathbb{R}^2$, its osculating circle is the circle that makes the greatest contact with the curve without ever crossing it [20]. In other words, the osculating circle is the circle that best fits inside the curve. A sharp bend will have a small osculating circle. The straighter the curve gets, the larger the osculating circle will be. The key observation is that locally the curvature of the curve equals the curvature of the osculating circle. So, calculating the curvature of the curve in a point amounts to calculating the curvature of the osculating circle in that point. An illustration of osculating circles for two different curves is displayed in Figure 5.3 on the following page.

The radius of the osculating circle defines the *radius of curvature* [20]. The osculating circle and radius of curvature of a curve are illustrated in Figure 5.4. Again, a sharply bent curve with high curvature will have a small osculating circle and radius of curvature, and the radius of curvature will increase as the curve straightens out. This indicates that the relationship between the curvature and the radius of curvature at a given point on the curve is inverse. This relationship can be derived directly from Equation (5.7). To do so, observe that a circle is bent by the same amount at every point, and thereby has constant curvature. The parameterisation

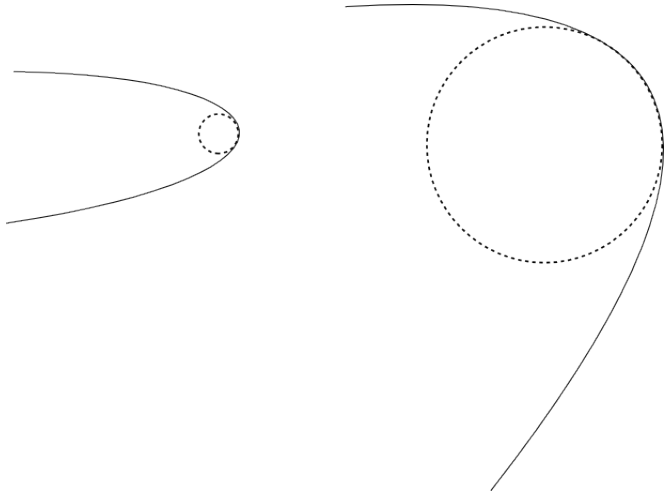


Figure 5.3.: Osculating circles. The sharp curve to the left has a smaller osculating circle than the straighter curve to the right.

of a circle γ, parameterised by arc length s, is $\gamma(s) = (\mathrm{R}\cos s, \mathrm{R}\sin s)$. In the context of Equation (5.7) the variables are

$$\begin{aligned} x &= \mathrm{R}\cos s, \quad x' = -\mathrm{R}\sin s, \quad x'' = -\mathrm{R}\cos s, \\ y &= \mathrm{R}\sin s, \quad y' = \mathrm{R}\cos s, \quad y'' = -\mathrm{R}\sin s. \end{aligned} \tag{5.8}$$

Substituting these variables into (5.7) gives

$$\kappa = \frac{y''x' - y'x''}{\left((x')^2 + (y')^2\right)^{\frac{3}{2}}} \tag{5.9}$$

$$= \frac{\mathrm{R}^2\sin^2 s + \mathrm{R}^2\cos^2 s}{\left((-\mathrm{R}\sin s)^2 + (\mathrm{R}\cos s)^2\right)^{\frac{3}{2}}} \tag{5.10}$$

$$= \frac{\mathrm{R}^2}{\left(\mathrm{R}^2\sin^2 s + \mathrm{R}^2\cos^2 s\right)^{\frac{3}{2}}} \tag{5.11}$$

$$= \frac{\mathrm{R}^2}{\left(\mathrm{R}^2\right)^{\frac{3}{2}}} \tag{5.12}$$

$$= \frac{1}{\mathrm{R}}. \tag{5.13}$$

This proves that the curvature is equal to the reciprocal of the radius of curvature, just as suggested.

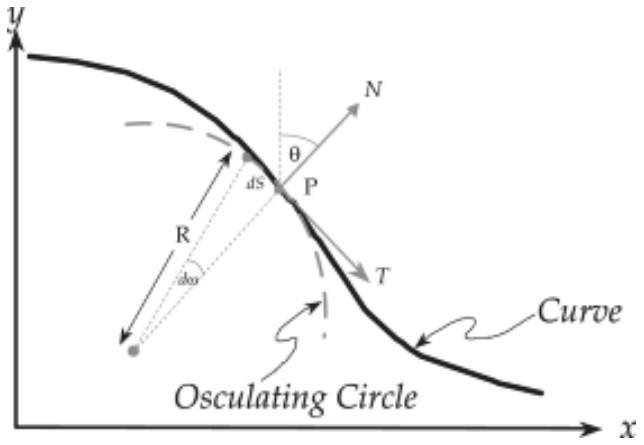


Figure 5.4.: A curve with its osculating circle and radius of curvature, R. This figure is copied from Roberts [20].

5.4. Curvature in three-dimensional data

The curves studied in this project are curves in $\mathbb{R}^2$. However, most of the relevant literature deals with curves in $\mathbb{R}^3$ in manually interpreted seismic surfaces. Before studying some of this literature in Section 5.5, the curvature definitions above need to be adapted to the three-dimensional case. To do so, simply imagine cutting through the surface with a plane. The imprint the surface makes on the plane results in a two dimensional curve to which the above curvature definitions may be applied. However, given a point on a 3D surface, there are infinitely many possible two dimensional slices, and thus a surface has infinitely many curvatures at a single point. In the following, a few of the most common 3D curvatures will be defined.

The most commonly used 3D curvatures are *normal curvatures*. The normal curvatures are those of curves defined by planes that are orthogonal to the surface in question [20].

Many curvature attributes are formed by a combination of normal curvatures. The *mean curvature*, K_m, is defined as the average of two orthogonal normal curvatures, K_1 and K_2, through the same point on a surface,

$$K_m = \frac{K_1 + K_2}{2}.$$

The mean curvature through a point on a surface is constant.

Out of all the infinitely many normal curvatures at a given point on the surface, the one with the highest absolute curvature defines the *maximum curvature*, K_{max}. The curve perpendicular to this defines the *minimum curvature*, K_{min}. The maximum and minimum curvatures constitute the *principal curvatures*.

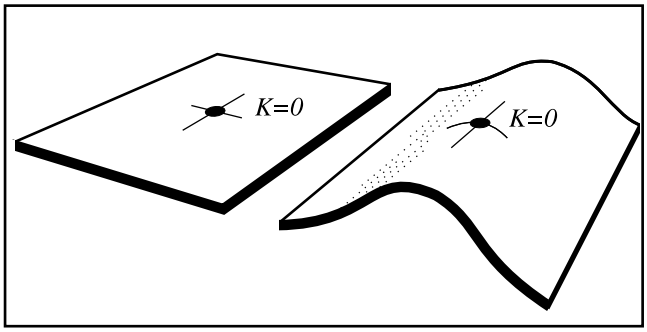


Figure 5.5.: Illustration of Gauss's *Theorema Egregium* [7]. Isometric bending of the surface does not change the Gaussian curvature. This illustration is copied from Lisle [15].

The principal curvatures give rise to a new curvature measure, the *Gaussian curvature*. Gaussian curvature, or total curvature, is defined as the product of the two principal curvatures, K_{min} and K_{max} [20],

$$K_g = K_{min} K_{max}.$$

Gauss stated that the Gaussian curvature, or *total curvature*, does not change if the surface is bended isometrically [7]. An illustration of this theorem is displayed in Figure 5.5.

5.5. From $\mathbb{R}^2$ to the discrete case

The definition of curvature presented in this chapter applies to continuous differentiable curves in $\mathbb{R}^2$. In the case of discrete data it is not obvious how to calculate curvature. In this section we will study some of the previous work that has been done on this subject.

Coeurjolly, Miguet, and Tougne [3] and Flynn and Jain [6] compare the two different approaches one can take to applying continuous mathematics on discrete data. The first approach is to approximate the data by a continuous function, so the continuous definitions can be applied to the disctrete data. This will usually involve interpolation or regression. The second approach is to derive discrete versions of the continuous definitions, that can be applied directly to the discrete data.

Coeurjolly, Miguet, and Tougne [3] go on to argue that using the first approach, and interpolating the data with a continuous function, is not a good choice as it depends on picking the right parameters and can be very computationally heavy. This critique will be addressed in Section 8.2.5. They then point out that when taking the second approach and finding a discrete curvature definition, there are three different ways to proceed. These are to define curvature in terms of either

tangent orientation, discrete derivatives or by osculating circles. The method presented by Coeurjolly, Miguet, and Tougne uses the last definition, and calculates curvature based on circle estimation and arc fitting.

Given a curve parameterized by arc-length, Lewiner et al. [13] present a method for curvature estimation consisting of fitting a second-order polynomial considered as a function of arc-length. The local arc-length is estimated, allowing the first- and second-order derivatives to be estimated. The fitting of the polynomial is done by a weighted least squares method.

The method presented by Pal and Bhowmick [18] applies to curves expressed in terms of chain codes. This is a purely discrete method, where curvature is calculated directly from the data. Earlier algorithms for calculating curvature based on chain codes look at the difference of sums of chain codes, while Pal and Bhowmick base their method on the sum of pairwise differences. They present the following equation for curvature calculation,

$$\kappa(p_i, k) = \frac{1}{k} \sum_{j=1}^{k} \min(f'_{i+j}, 8 - f'_{i+j}),$$

where $f'_{i+j} = |f_{i+j} - f_{i-j+1}|$. The chain code difference is defined to be the shortest of the two differences f'_{i+j} and $8 - f'_{i+j}$ in order to accurately estimate the true curvature. The method shows good results, particularly in high curvature points such as corners. It is, according to the authors, a definite improvement from the difference of sums approach.

Roberts [20] applies the theory of discrete curvature estimation to seismic data in particular. More specifically, he studies curvature in three-dimensional manually interpreted horizons. The method involves locally fitting a quadratic surface to the mapped surface using a least squares regression. In order to calculate the curvature in a particular point the eight surrounding neighbour points are used. A general quadratic surface is given by the equation

$$z = ax^2 + by^2 + cxy + dx + ey + f. \tag{5.14}$$

This expression has six free variables. Because a neighbourhood of eight points is used in the estimation this results in a overdetermined system which is solved using the method of least squares. The result is a set of expressions for the six coefficients in Equation 5.14. Roberts goes on to provide expressions for the 3D curvatures presented in Section 5.4, and a few others, using these coefficients.

One of the numerical curvature estimation methods discussed by Flynn and Jain [6] is the surface normal change method presented by Hoffman and Jain [10]. This is a 3D method that estimates the curvature in moving from pixel p to pixel q as

$$k(p, q) = \frac{||n_p - n_q||}{||p - q||} s(p, q),$$

where n_p is the unit vector normal to the tangent plane at pixel p and s is a sign factor which is 1 when the curvature is positive and -1 when the curvature is negative. The notation $||\cdot||$ denotes the Euclidean distance from the origin in 3D space. This curvature estimate is used to define a series of 3D curvature measures.

There are clearly several challenges related to calculation of curvature for digitized curves, and various approaches exist. If we follow the approach of fitting a continuous curve to the data in order to apply the continuous curvature definitions, this will usually involve some kind of derivative estimation. In this case, the challenge is to find good and robust derivative estimates. Depending on the shape of the curve in question, this can be a source of errors. If we choose the second approach, and derive a discrete curvature expression, we will have to make some assumptions about the curve. This could, again, be a source of error. However, depending on the application, it may well be sensible to make such assumptions about the curve.

6. Estimating derivatives

In order to calculate the curvature of a discrete curve, the first and second order derivatives need to be approximated. This can be done in several ways, and three methods will be explored in this chapter. What they have in common is that they approximate the derivatives of the discrete curve by calculating the derivatives of a continuous function. This function is fitted to the discrete data, allowing it to be a continuous approximation to the discrete curve.

Discrete partial derivatives have already been introduced in Chapter 4, where they were used in image gradient estimation. The derivative estimates presented in the following section are essentially the same as the filters introduced in Section 4.2, only in the setting of discrete curves in $\mathbb{R}^2$ rather than discrete images. However, in spite of being a little repetitive, the finite difference derivative estimate will be fully derived in the following for completeness.

The finite difference and parabola interpolation methods presented in Sections 6.1 and 6.2 are based on material from the compendium by Mørken [17]. The theoretical background for the least squares method in Section 6.3 is from the linear algebra textbook by Lay [12].

6.1. Finite differences

The simplest and most common way to approximate derivatives is with *finite differences*. The finite difference approach to derivative estimation is based on approximating a curve segment with a straight line, and differentiating this line. The line that interpolates f in the points a and $a+h$ is given by

$$l(x) = f(a) + \frac{f(a+h) - f(a)}{h}(x-a). \tag{6.1}$$

This line is commonly known as the *secant* of f in the points a and $a+h$. Taking the derivative of (6.1) yields

$$l'(x) = \frac{f(a+h) - f(a)}{h}. \tag{6.2}$$

So, letting $l(x)$ approximate f locally we have

$$f'(a) \approx l'(a) = \frac{f(a+h) - f(a)}{h}. \tag{6.3}$$

Alternatively, this relation can be derived through the definition of $f'(x)$,

$$f'(a) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h}. \tag{6.4}$$

If we drop the limit in (6.4), and let h be an arbitrary small constant, we end up with (6.3).

An approximation to the second derivative of f is obtained by applying the first derivative estimate in (6.3) to $f'(a)$:

$$f''(a) \approx \frac{f'(a+h) - f'(a)}{h} \tag{6.5}$$

$$= \frac{f(a+h) - 2f(a) + f(a-h)}{h^2}. \tag{6.6}$$

6.2. Parabola interpolation

The finite difference method for derivative estimation can be viewed as an interpolation method, where the data is interpolated with straight line segments. If prior knowledge about the shape of the data is available, it may be sensible to interpolate with more complex functions than straight lines. In this section we will look at estimating derivatives by interpolating with second degree polynomials, or *parabolas*.

A general parabola has three free coefficients. Therefore, three points in a local neighbourhood are picked and the discrete curve is approximated by forcing a parabola through these points. Given any set of three points there exists a unique parabola passing through those points, so this problem has a unique solution.

Parabolas are most commonly written in the form

$$r(t) = a_0 + a_1 t + a_2 t^2, \tag{6.7}$$

where a_0, a_1 and a_2 are constants, and we have let $r(t)$ represent the curve as a function of time, t. However, for the purpose of interpolation, the *Newton form* is more beneficial [17]. Given a set of three points, $\{t_k\}_{k=i-1}^{i+1}$, the Newton form of a parabola is as follows,

$$r(t) = a_0 + a_1(t - t_{i-1}) + a_2(t - t_{i-1})(t - t_i). \tag{6.8}$$

We can now set up the following system of equations which can be solved to find the constants a_0, a_1 and a_2:

$$r(t_{i-1}) = a_0 + a_1(t_{i-1} - t_{i-1}) + a_2(t_{i-1} - t_{i-1})(t_{i-1} - t_i) \tag{6.9}$$

$$r(t_i) = a_0 + a_1(t_i - t_{i-1}) + a_2(t_i - t_{i-1})(t_i - t_i) \tag{6.10}$$

$$r(t_{i+1}) = a_0 + a_1(t_{i+1} - t_{i-1}) + a_2(t_{i+1} - t_{i-1})(t_{i+1} - t_i). \tag{6.11}$$

Several of the t's cancel out, so we end up with

$$r(t_{i-1}) = a_0 \tag{6.12}$$
$$r(t_i) = a_0 + a_1(t_i - t_{i-1}) \tag{6.13}$$
$$r(t_{i+1}) = a_0 + a_1(t_{i+1} - t_{i-1}) + a_2(t_{i+1} - t_{i-1})(t_{i+1} - t_i). \tag{6.14}$$

The above equations can easily be solved using simple algebra. We have

$$a_0 = r(t_{i-1}). \tag{6.15}$$

Substituting a_0 in (6.13) gives

$$r(t_i) = r(t_{i-1}) + a_1(t_i - t_{i-1}). \tag{6.16}$$

So we have

$$a_1 = \frac{r(t_i) - r(t_{i-1})}{t_i - t_{i-1}}. \tag{6.17}$$

For a_3 the solutions for a_0 and a_1 are substituted into (6.14)

$$r(t_{i+1}) = r(t_{i-1}) + \frac{r(t_i) - r(t_{i-1})}{t_i - t_{i-1}}(t_{i+1} - t_{i-1}) + a_2(t_{i+1} - t_{i-1})(t_{i+1} - t_i). \tag{6.18}$$

Rearranging (6.18) results in

$$a_2 = \frac{\frac{r(t_{i+1}) - r(t_{i-1})}{t_{i+1} - t_{i-1}} - \frac{r(t_i) - r(t_{i-1})}{t_i - t_{i-1}}}{t_{i+1} - t_i}. \tag{6.19}$$

We now have a general expression for a parabola that passes through the three points $r(t_{i-1})$, $r(t_i)$ and $r(t_{i+1})$,

$$r(t) = a_0 + a_1(t - t_{i-1}) + a_2(t - t_{i-1})(t - t_i), \tag{6.20}$$

where

$$\begin{aligned} a_0 &= r(t_{i-1}) \\ a_1 &= \frac{r(t_i) - r(t_{i-1})}{t_i - t_{i-1}} \\ a_2 &= \frac{\frac{r(t_{i+1}) - r(t_{i-1})}{t_{i+1} - t_{i-1}} - \frac{r(t_i) - r(t_{i-1})}{t_i - t_{i-1}}}{t_{i+1} - t_i}. \end{aligned} \tag{6.21}$$

Equation (6.20) is easy to differentiate. The first and second derivatives are

$$r'(t) = a_1 + a_2(2t - t_i - t_{i-1}), \tag{6.22}$$
$$r''(t) = 2a_2. \tag{6.23}$$

6.3. Least-squares fitting

In the interpolation method presented in the previous section, a parabola was fitted locally to the discrete curve by forcing it through three data points in a local neighbourhood. An alternative approach to the parabola fitting problem is to include more data points in the estimation. This results in an overdetermined set of equations, to which a solution must be approximated.

Given a point, (x_i, y_i), we wish to locally fit a parabola to a set of points consisting of (x_i, y_i) and a certain number of neighbours. As we saw in the previous section, the form of a general parabola is

$$r(t) = a_0 + a_1 t + a_2 t^2. \tag{6.24}$$

Fitting a parabola to data involves determining the parameters a_0, a_1 and a_2. If more than three data points are included in the calculation, the result is an overdetermined system that, in general, has no solution. Solving this problem consists of finding the parabola that *best fits* the set of data points, rather than forcing the parabola through the points as in the interpolation approach described in the previous section. In the following the best fit parabola problem will be mathematically formulated.

Given a set of points, $S = \{(x_j, y_j)\}_{j=1}^{n}$, in the neighbourhood of (x_i, y_i), we wish to find the parabola that best fits the points in S. Had the data points fit the parabola in Equation 6.24 perfectly, the parameters a_0, a_1 and a_2 would satisfy the equations

$$\begin{aligned} r(t_1) &= a_0 + a_1 t_1 + a_2 t_1^2, \\ r(t_2) &= a_0 + a_1 t_2 + a_2 t_2^2, \\ &\vdots \\ r(t_n) &= a_0 + a_1 t_n + a_2 t_n^2. \end{aligned} \tag{6.25}$$

This system can be written as

$$r = Aa, \tag{6.26}$$

where

$$r = \begin{bmatrix} r(t_1) \\ r(t_2) \\ \vdots \\ r(t_n) \end{bmatrix}, \quad A = \begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ \vdots & \vdots & \vdots \\ 1 & t_n & t_n^2 \end{bmatrix}, \quad a = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}. \tag{6.27}$$

However, because the data points most likely do not fit perfectly to the parabola, the system has no solution. The system is, however, a least-squares problem, and can be solved as such. For details on solving least-squares problems, see Chapter 6 in the linear algebra book by Lay [12].

The least-squares solution to 6.26 is the solution of the *normal equations* [12],

$$\begin{aligned} A^T r &= A^T A a \\ &\Downarrow \\ \hat{a} &= \begin{bmatrix} \hat{a}_0 \\ \hat{a}_1 \\ \hat{a}_2 \end{bmatrix} = (A^T A)^{-1} A^T r. \end{aligned} \tag{6.28}$$

The solutions are denoted by $\hat{a}_i$ because they are approximations to the true a_i.

Finding estimates for the derivatives now simply amounts to differentiating (6.24) and substituting a_i with $\hat{a}_i$, much like in Section 6.2. The derivatives of (6.24) are

$$r'(t) = a_1 + 2a_2 t, \tag{6.29}$$

$$r''(t) = 2a_2. \tag{6.30}$$

Substituting the coefficients with the approximations results in the following derivative estimates

$$\hat{r}'(t) = \hat{a}_1 + 2\hat{a}_2 t, \tag{6.31}$$

$$\hat{r}''(t) = 2\hat{a}_2. \tag{6.32}$$

It must be noted that the basis $\{1, t, t^2, t^3, ...\}$ is numerically ill-conditioned, particularly in high dimensions. However, in the quadratic case it is reasonable to assume that this will not cause problems. Furthermore, in the context of estimating curvature of seismic horizons, details of which will be explained in Part II, computational accuracy is not of great importance. The main concern is the size of the curvature values relative to each other.

Three methods for estimation of the derivatives of a discrete curve have been presented in this chapter. The choice of method depends on the nature of the problem being solved. In Chapter 8, the three methods presented here will be compared, and the method best suited for derivative approximation in the context of curvature estimation in seismic images will be determined.

This chapter concludes the background material constituting Part I of this project. In Part II the material presented here will be compiled into a method for estimating curvature in a seismic image.

Part II.

Method & Results

7. Introduction

In Part I, the theoretical background needed for this project was established. In this part of the project, all the theory from Part I will be put together in an attempt to solve the problem of highlighting saltdomes in seismic images, as described in Chapter 1.

Chapter 8 presents the method used to create an image representing the curvature of the seismic data. The method consists of two parts. The first part involves automatically extracting segments of seismic horizons in the form of discrete curves. This is followed by estimation of the curvature of the discrete curve segments. The result of performing these steps for every image pixel will be an image depicting curvature.

In Chapter 9, the results of performing the method from Chapter 8 are presented. The method is first tested on the single inline image that has been studied so far in the project, followed by an analysis of the results. Thereafter, for completeness in the analysis of the method, two new inline saltdome images are presented and the method is tested on these as well. The chapter concludes with an analysis of the results.

In the projects final chapter, a summary of the presented method and results will be given. This will be followed by a few thoughts on limitations, possible improvements and further work.

8. Pixel based curvature estimation

In this chapter, a method for estimating the curvature of an inline seismic image will be presented. The method consists of two parts. First, the dip, as presented in Chapter 4, is used to track horizons. This allows segments in the form of *discrete curves* starting at a given pixel to be extracted.

Definition 8.1 (Discrete curve)**.** A *discrete curve* is an ordered set of points of the form $\{(x_i, y_i)\}_{i=1}^{N}$ which form a curve segment.

After extracting the curve segment the curvature around the pixel is estimated. This is done by estimating the curvature of the tracked curve segment, using the curvature expression derived in Chapter 5. The first and second derivatives of the two curve components, which are needed in the curvature calculation, are estimated as described in Chapter 6.

The method presented in the following differs from previous work on the subject of curvature estimation in seismic data in that it is fully automatic. While most alternative methods address estimation of curvature for manually interpreted horizons, the presented method uses the dip attribute to automatically interpret the horizons before curvature is estimated. One exception is the work by Klein, Richard, and James [11] who present a method for estimation of curvature in three-dimensional seismic volume data. The method uses window-based cross-correlation to propagate a small surface around each data sample. This is followed by fitting of a least squares quadratic surface, and the sample curvature is estimated as the curvature of this surface of this surface. This method is similar to the method presented in the following, but differs in that the automatic interpretation is performed by cross-correlation rather than dip.

8.1. Tracking curves

As described in Chapter 4, the term *dip* refers to the angle a curve makes with the horizontal axis. The dip image, containing the estimated dip for every pixel in the input image, provides valuable information about the structure of the image. In this project, the dip estimation method presented by Randen et al. [19] in Section 4.4, is used to create the dip image. The original saltdome image, inline number 8505 from the North Sea dataset, is displayed in Figure 8.1 on the following page. The resulting dip image can be found in Figure 8.2.

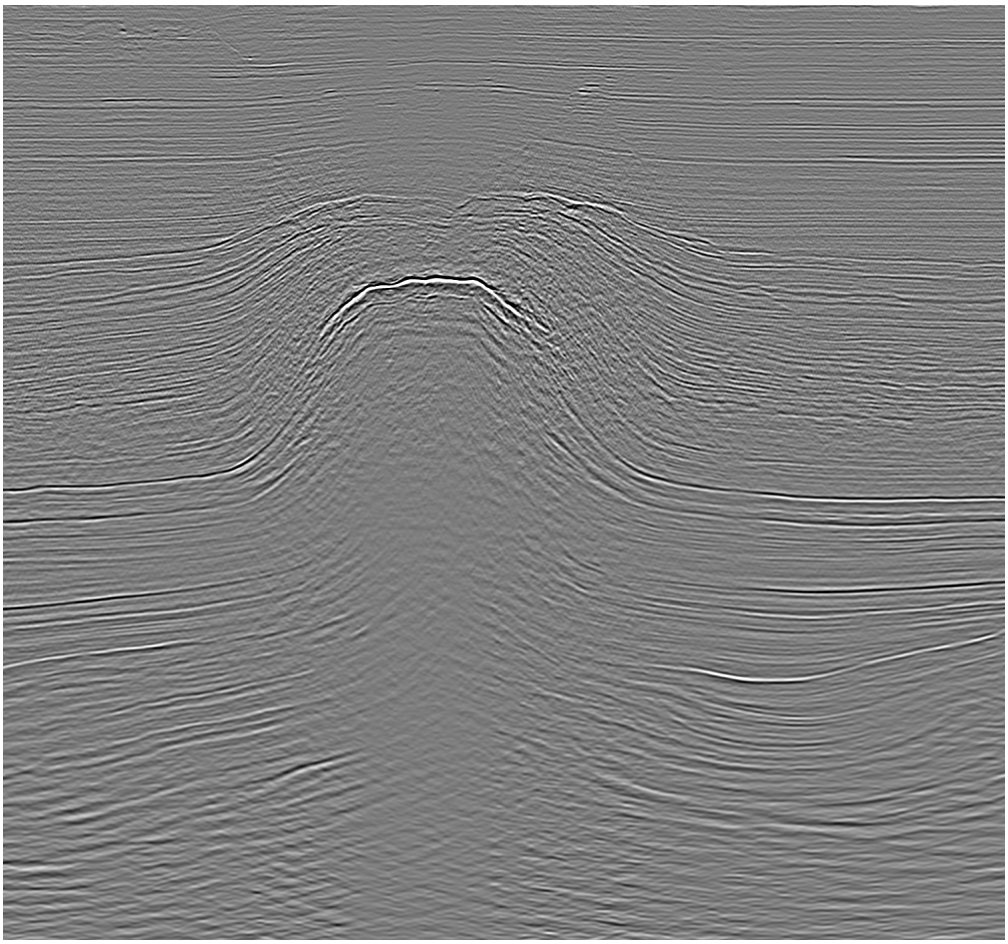


Figure 8.1.: Inline number 8505 from the North Sea dataset.

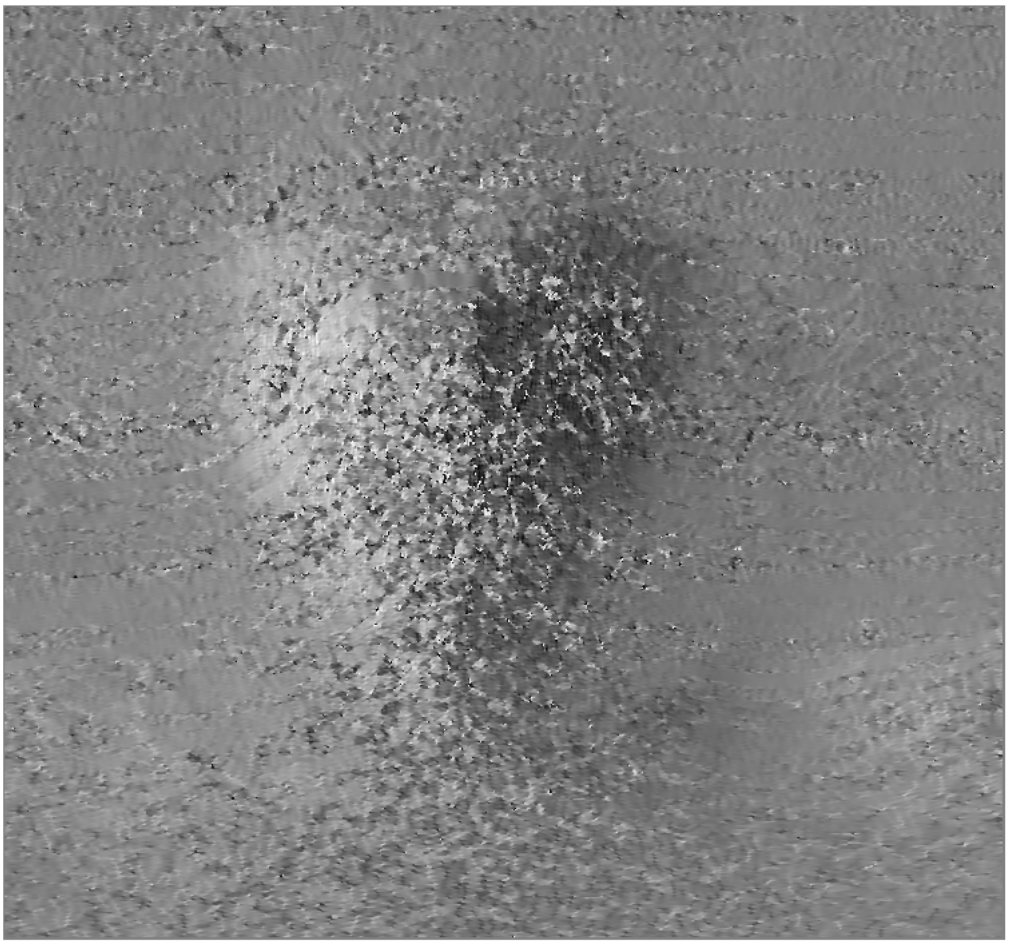


Figure 8.2.: Dip image created using the method presented by Randen et al. [19].

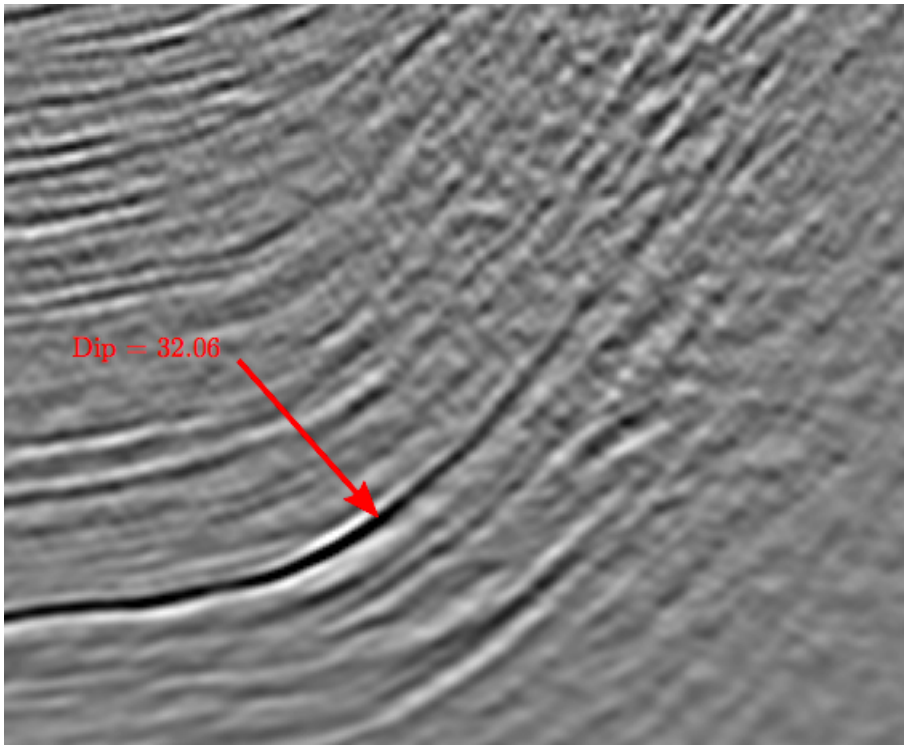


Figure 8.3.: The pixel indicated by the arrow is on a strong negative reflector and has a dip value of 32.06.

The basic idea behind the method proposed here is to view the horizons in an inline seismic image as parametric curves. The saltdome image in Figure 8.1 has a clear horizontally layered structure. The seismic layers, or *reflectors*, curve around the saltdome in the center of the image. This gives them distinctive curved shape. When viewing the reflectors as curves, this shape should result in a pattern in the curvature of the curves.

Recall from Chapter 2 that the sedimentary layers in a seismic image are called *horizons*. In order to represent the horizons as digital curves, the dip is put to use. Each pixel of the dip image holds the angle of the dominant direction for that pixel in the original image. In other words, moving in this direction is likely to be a movement along a horizon. As an example, consider the cropped image in Figure 8.3. The pixel indicated by the red arrow has a dip value of 32.06. In this example we allow the horizontal axis to have zero dip. The indicated pixel is part of a strong negative reflector which has a gentle upward slope in the area around the indicated pixel. Observing the shape of the reflector, it is obvious that the local angle in the indicated pixel, relative to the horizontal axis, must be approximately 30°. This observation corresponds well to the dip value.

So, the angle given by the dip gives the orientation of the horizon the pixel in question belongs to. The aim is to use this information to move from one pixel to its neighbour, in a given direction, on the horizon. Repeating this for a given number of points will result in a digital curve segment, hopefully following a horizon. To summarize, the input to the method will be a starting pixel and the number of pixels to be tracked, and the output should be a discrete curve.

In the following two subsections two different approaches to this problem will be explored. The first approach uses simple trigonometry to find the exact direction of the horizon in the pixel. In the second approach the possible dip angles are divided into a discrete set of intervals, and one neighbouring pixel is assigned to each interval. Then, given the dip, which of the neighbouring pixels is next on the curve can be decided.

8.1.1. Trigonometry

Given a pixel in the dip image, we have the orientation of the horizon the pixel belongs to. One way of using this angle to extract a curve is by fixing the step length in the horizontal direction to one pixel. Then, the dip angle can be used to determine the step length in the vertical direction using the simple trigonometric relation

$$\tan\phi = \frac{\Delta v}{\Delta h}, \tag{8.1}$$

where h and v are the horizontal and vertical step size, respectively, and ϕ is the dip angle of the current pixel. Figure 8.4 on the following page gives an illustration of this relationship. Letting the horizontal step size be one pixel, $h = 1$, Equation (8.1) reduces to

$$v = \tan\phi. \tag{8.2}$$

Using (8.2) iteratively, the path along the horizon can be tracked, pixel by pixel. The pixel coordinates, (x, y), easily translate to digital vector form, $\alpha = (x_i, y_i)_{i=1}^{N}$, where N is the desired length of the curve segment. One may also wish to travel across the image in a backwards, rather than forwards, manner. This is done by setting the step length, h, to -1.

Figure 8.5 on page 61 shows the results of tracking 100 pixels in either direction, starting in pixels $(303, 528)$ and $(631, 408)$. Although the curves do follow the general shape of the horizons in both sub-images, they both trail off in the far right end. Particularly the curve in (b) makes a turn which is not as expected. It may seem that the trigonometric curve tracking method is somewhat unstable. In the next subsection a different approach will be explored.

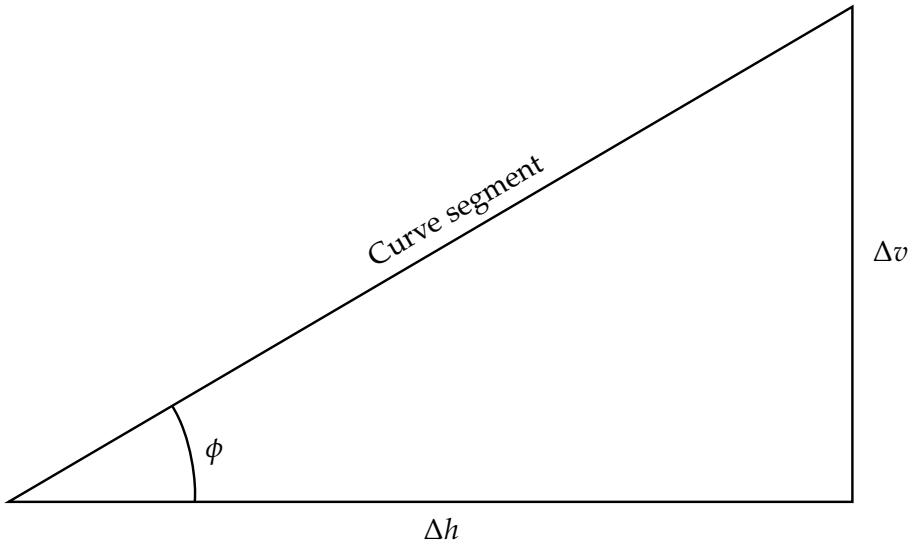


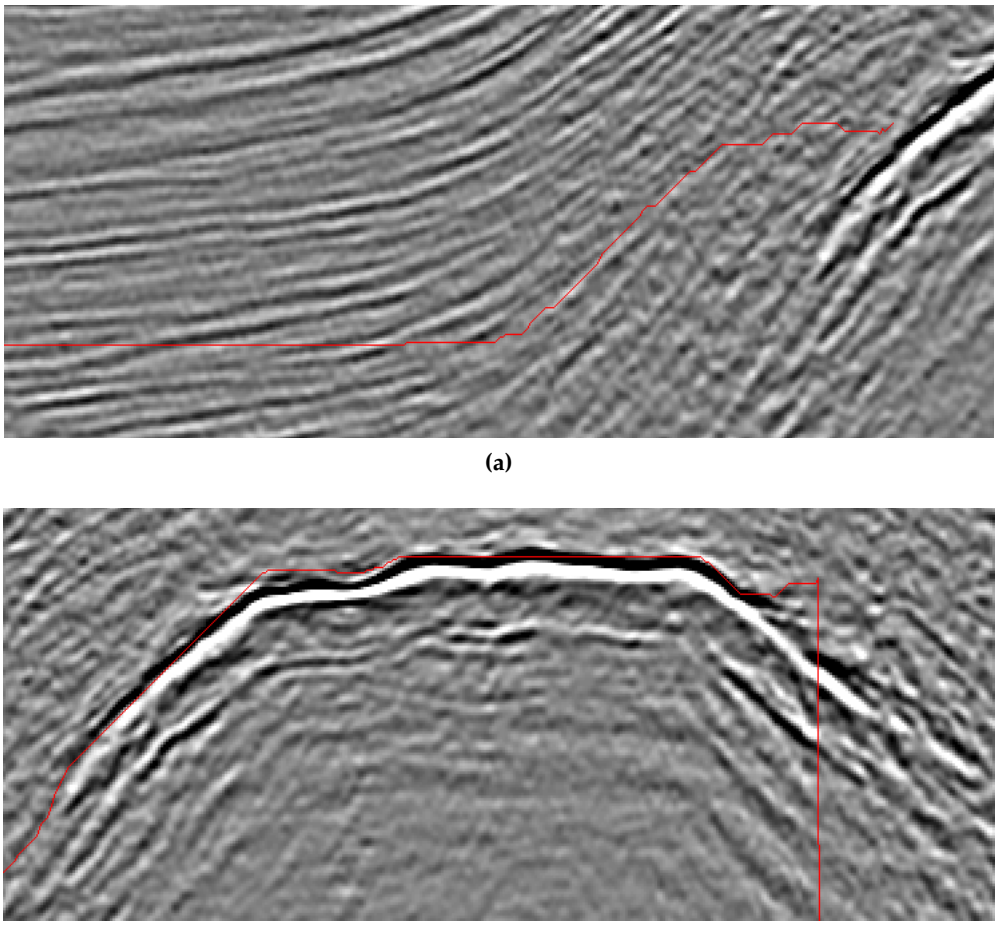
Figure 8.4.: Triangle illustrating a step in tracking a horizon. The step length is set in the horizontal direction, and, given the angle form the calculated dip, the vertical step length is calculated.

8.1.2. A discrete approach

In the above approach to curve tracking, the horizontal step length is assumed to always be one pixel, $h = 1$. This may not always be a sensible assumption. In the case of a curve that is close to vertical, the vertical step length will quickly become very large. In fact, theoretically, if we have a vertical curve and assume $h = 1$, then v will approach infinity. This may be part of the reason why the method is somewhat unstable.

An alternative approach is to base the algorithm on the assumption that we in each step move from the current pixel to one of its eight neighbours. The 8-neighbourhood for a pixel, $x_{i,j}$, is illustrated in Figure 8.6. The possible dip values have a total range of 180°. The dip estimation technique presented in Chapter 4 results in dip values in the interval $[-90^\circ, 90^\circ]$. The discrete curve tracking method involves splitting this interval into smaller intervals, each corresponding to one of the neighbouring pixels.

As with the trigonometric approach presented in the previous subsection, we can decide to move forwards or backwards from the center pixel. The current direction decides which of the neighbouring pixels are possible to travel to. Considering that the range of possible dip values is 180°, it is quite intuitive that the range of possible neighbouring pixels is clockwise and anticlockwise from the pixel vertically above to vertically below the current pixel, when moving forwards and backwards, respectively. This is illustrated in Figure 8.7 on page 63.



(a)

(b)

Figure 8.5.: Centered curve segments tracked using the trigonometric curve tracking method described in Subsection 8.1.1. The curve in (a) is tracked from pixel $(303, 528)$, and (b) is tracked from pixel $(631, 408)$.

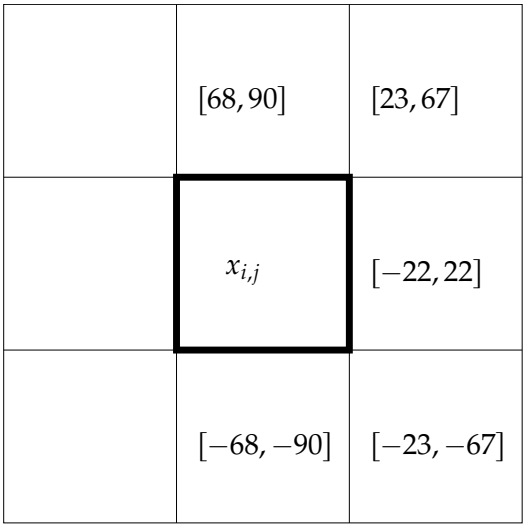
$x_{i-1,j-1}$	$x_{i-1,j}$	$x_{i-1,j+1}$
$x_{i,j-1}$	$x_{i,j}$	$x_{i-1,j-1}$
$x_{i+1,j-1}$	$x_{i+1,j}$	$x_{i+1,j+1}$

Figure 8.6.: The eight-neighbourhood of pixel $x_{i,j}$.

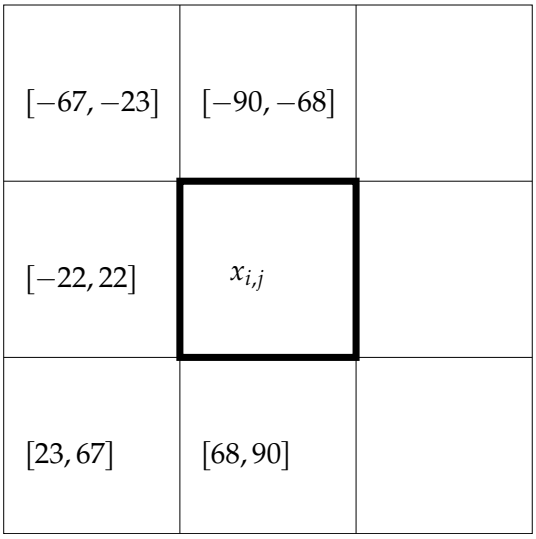
The final question is to decide the *size* of the angle intervals. The most obvious would be to split the total angle range into five equally sized intervals. However, this would result in the pixels vertically above and below the current pixel to have a range double of that of the other pixels, when taking both directions into account. To avoid this, the size of the intervals for these two pixels is made half the size of the other intervals. When the angle range is $[-90^\circ, 90^\circ]$, this results in the angle intervals presented in Figure 8.7, where (a) shows the pixels for forward tracking, and (b) for backwards tracking.

The discrete curve tracking method is tested on the inline saltdome image, using the same two starting points as with the trigonometric method in the previous subsection. The results are displayed in Figure 8.8 on page 64. When comparing these to the results from the previous subsection in Figure 8.5, it seems that the discrete method performs better than the trigonometric method. Neither of the two curves trails off like the curves in Figure 8.5 did. Rather, they follow the saltdome shape in the image closely. Note that they do not follow a single reflector perfectly, but jump between neighbouring reflectors. However, they do follow the general shape of the structure in the image.

Considering the left segment of the curve in Figure 8.8a, it is clear that there is some room for improvement in the curve tracking method. The curve is completely horizontal in this area, although the underlying horizons actually have slight upwards dipping shape. A possible solution to this problem may be to reduce the span of the angle interval corresponding to a horizontal move. An alternative possible improvement could be to determine the size of the intervals analytically, rather than letting them be uniform. However, as discussed in the previous section the curves do follow the general shape in the image and prove to give satisfactory

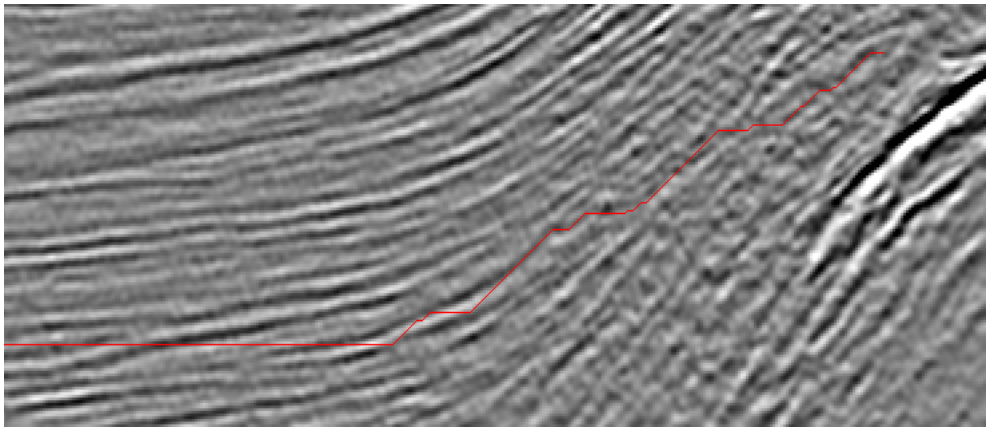


(a)

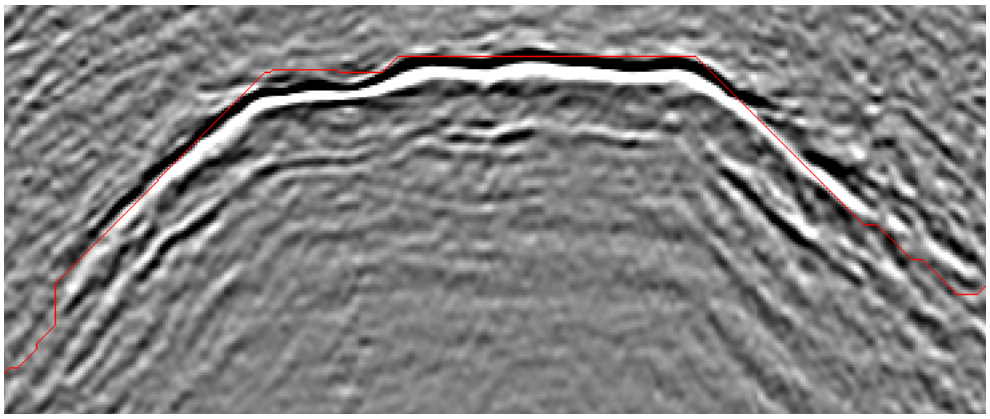


(b)

Figure 8.7.: Subfigure (a) shows the angle intervals for each neighbouring pixel of pixel $x_{i,j}$ that can be reached in forward curve tracking. Subfigure (b) shows the equivalent pixels for backward curve tracking. The possible dip angles are in the interval $[-90°, 90°]$, with the horizontal axis having zero dip.



(a)



(b)

Figure 8.8.: Centered curve segments tracked using the discrete curve tracking method described in Subsection 8.1.2. The curves in (a) and (b) are tracked from pixel $(303, 528)$ and $(631, 408)$, respectively.

results in the later curvature estimation, so this is not further pursued. It would, however, be an interesting topic for further study.

The discrete curve tracking method, with the dip intervals in Figure 8.7, will be used for the curve tracking throughout this project. Having found a successful method for extracting discrete curves from seismic images, the following section will explore how the curvature of these curves can be estimated in practice.

8.2. Curvature

In the previous section, a method for extracting curve segments from a saltdome image using the dip was presented. The curve segments are in discrete form,

$\alpha = (x_i, y_i)_{i=1}^N$, where N is the length of the curve segment. In this section, we will study how the curvature of these segments can be estimated.

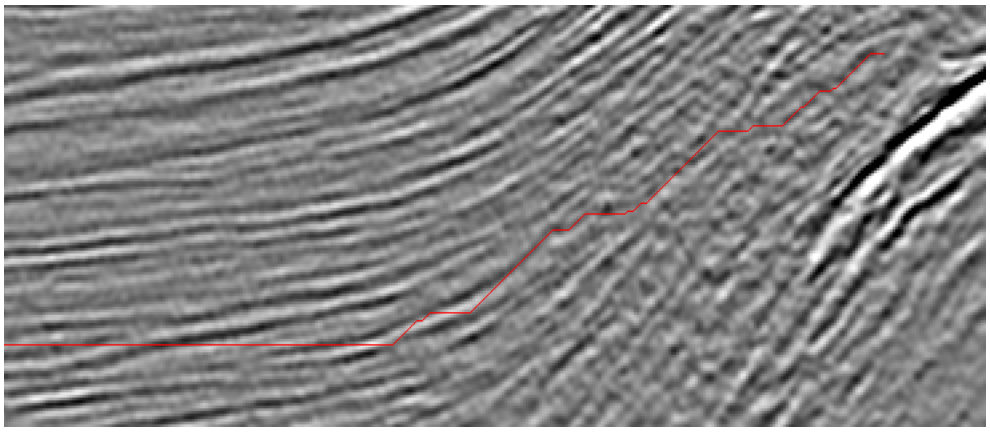
8.2.1. Choosing a method for derivative estimation

The first step in estimating curvature is estimation of derivatives. In Chapter 6, three ways of estimating discrete derivatives were presented. The first method involved finite differences, the second method was based on interpolating the curve segment with parabolas, and the third method fitted a parabola to the full set of curve points using the method of least squares. In this section one of these three methods will be chosen to be used in the curvature estimation.

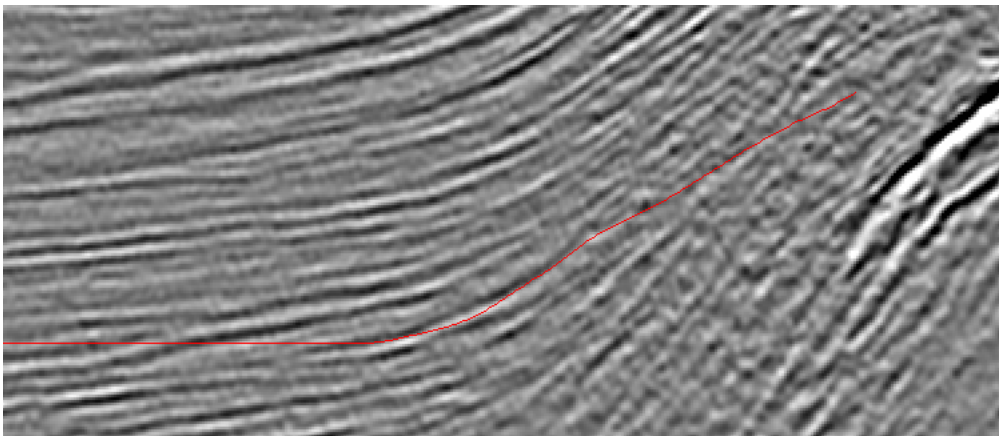
One of the main differences between the three derivative estimation methods is that while the least squares method takes all the available data points into account when fitting the parabola, the finite difference and parabola interpolation methods only use three points in the estimation. This makes these two methods more sensitive to noise in the data, as the line segments or parabolas are forced through these points. To tackle this problem the discrete curves were filtered before any calculations were done. Mean, median and Gaussian filters with sizes varying from 10 to 100 pixels were tested. The experiments showed that a 50 pixel long mean filter gave the best results. The results of filtering the two curves from Figure 8.8 are shown in Figures 8.9 and 8.10. The filtered curves are smoothed out, but still retain the general shape of the horizons. Note that the filtering shortens the curve segments slightly.

The least squares parabola fitting finds the curve that best fits the data points, rather than forcing it through them as in the two interpolation methods described above. It can therefore be considered to have the filtering step integrated in the method. This means that the filtering which is necessary for the two other methods is obsolete for the least squares method.

Because there are no exact derivatives to compare the estimates to, the only way to asses them is by visual inspection. There are four derivatives being estimated in each method. These are the first and second order derivatives for the x- and y-components of the discrete curves, making twelve result images in total, all of which will not be included here. The first order derivative of the y-component is displayed in Figures 8.11, 8.12 and 8.13, respectively. Observe that the results of the two interpolation methods, in Figures 8.11 and 8.12, are practically identical. Because of the shape of the saltdome curves one could intuitively assume that a parabola would be better suited than straight line segments in an approximation, and be surprised by this result. However, the similarity in the results is probably due to the fact that only three data points were involved in the estimation, as discussed in the previous section. The result of the least squares fitting method, displayed in Figure 8.13, is much smoother than those of the two other methods. This may be because all the available data has been included in the estimation,

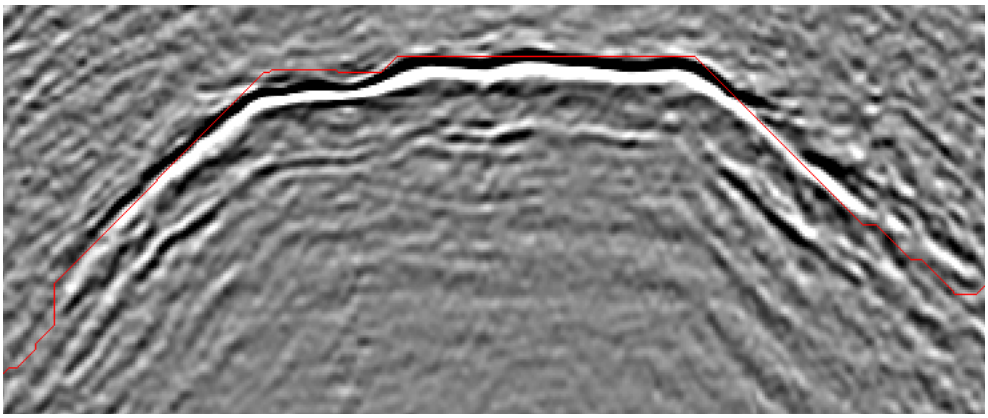


(a)

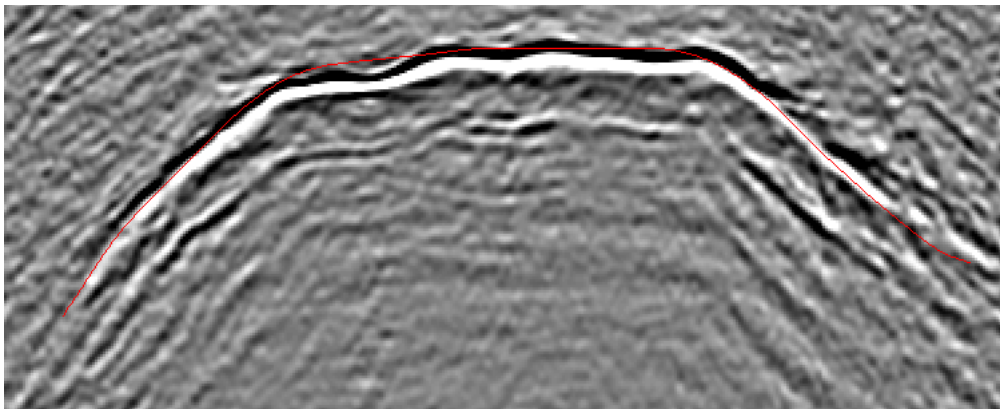


(b)

Figure 8.9.:
(a) Curve tracked from pixel $(303, 528)$ using the discrete curve tracking method from Section 8.1.
(b) The curve in (a) filtered using a 50 point mean filter.



(a)



(b)

Figure 8.10.:
(a) Curve tracked from pixel $(631, 408)$ using the discrete curve tracking method from Section 8.1.
(b) The curve in (a) filtered using a 50 point mean filter.

which means that the risk of outlier data points impacting the result is minimal. The conclusion is that as the least squares method is the most theoretically sound and also gives the smoothest results, this is the method that should be used for the curvature estimation in this project.

8.2.2. Least squares parabola fitting in practice

Based on Section 6.3, the implementation of the method of least squares derivative estimation is simple. First the t-values must be calculated. This is done iteratively with $t_0 = 0$, using the following expression

$$t_i = t_{i-1} + \sqrt{(y_i - y_{i-1})^2 + (x_i - x_{i-1})^2}. \tag{8.3}$$

Once the t-values have been calculated the derivative estimation simply consists of setting up the matrices from Equation (6.27), calculating the coefficient approximations, and estimating the derivatives using Equation (6.31). The derivatives in the x- and y-direction are estimated independently. The curve dataset is of the form $\alpha = (x_i, y_i)_{i=1}^N$, giving two r-vectors which we call r_x and r_y. So we have the matrices

$$r_x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad r_y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad A = \begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ \vdots & \vdots & \vdots \\ 1 & t_n & t_n^2 \end{bmatrix}. \tag{8.4}$$

Once the matrices have been set up the $\hat{a}$ coefficients can be estimated,

$$\hat{a}_x = \begin{bmatrix} \hat{a}_{0x} \\ \hat{a}_{1x} \\ \hat{a}_{2x} \end{bmatrix} = (A^T A)^{-1} A^T r_x, \tag{8.5}$$

$$\hat{a}_y = \begin{bmatrix} \hat{a}_{0y} \\ \hat{a}_{1y} \\ \hat{a}_{2y} \end{bmatrix} = (A^T A)^{-1} A^T r_y. \tag{8.6}$$

The final derivative estimates are

$$\hat{r}'_x(t) = \hat{a}_{x1} + 2\hat{a}_{x2}t, \tag{8.7}$$

$$\hat{r}''_x(t) = 2\hat{a}_{x2} \tag{8.8}$$

in the x-direction, and

$$\hat{r}'_y(t) = \hat{a}_{y1} + 2\hat{a}_{y2}t, \tag{8.9}$$

$$\hat{r}''_y(t) = 2\hat{a}_{y2} \tag{8.10}$$

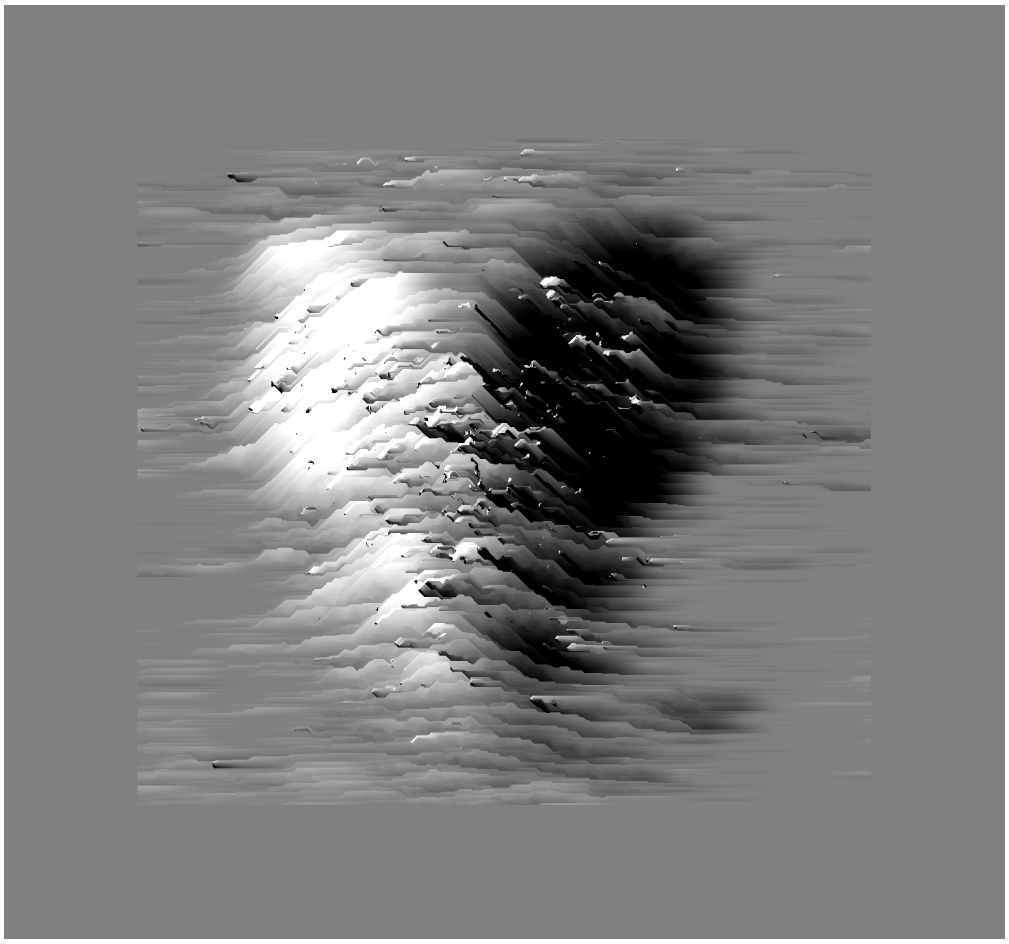


Figure 8.11.: First order derivative of y-component of discrete curve segments, estimated by the finite difference method using curve segments of 200 points in length. To improve readability, the intensities have been inverted and the contrast improved. This means that the light areas correspond to negative values, and the dark areas correspond to positive values.

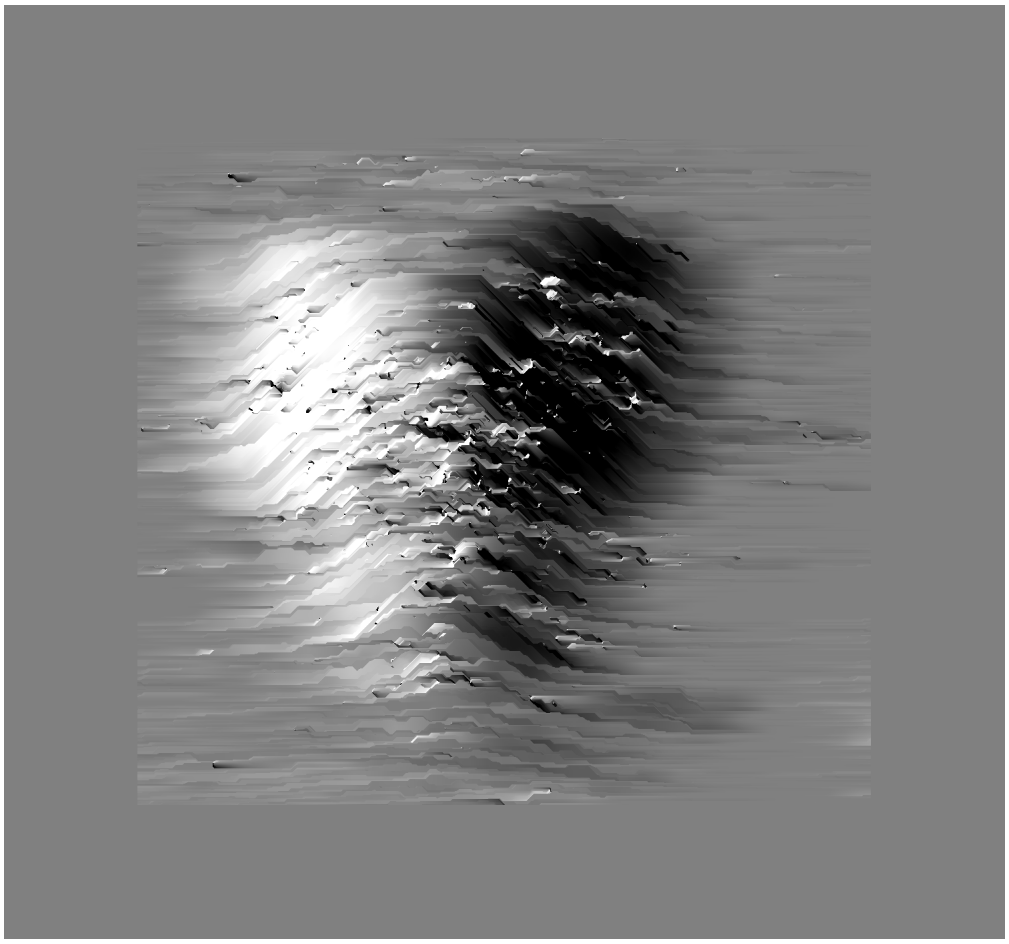


Figure 8.12.: First order derivative of y-component of discrete curve segments, estimated by the parabola interpolation method using curve segments of 200 points in length. To improve readability, the intensities have been inverted and the contrast improved. This means that the light areas correspond to negative values, and the dark areas correspond to positive values.

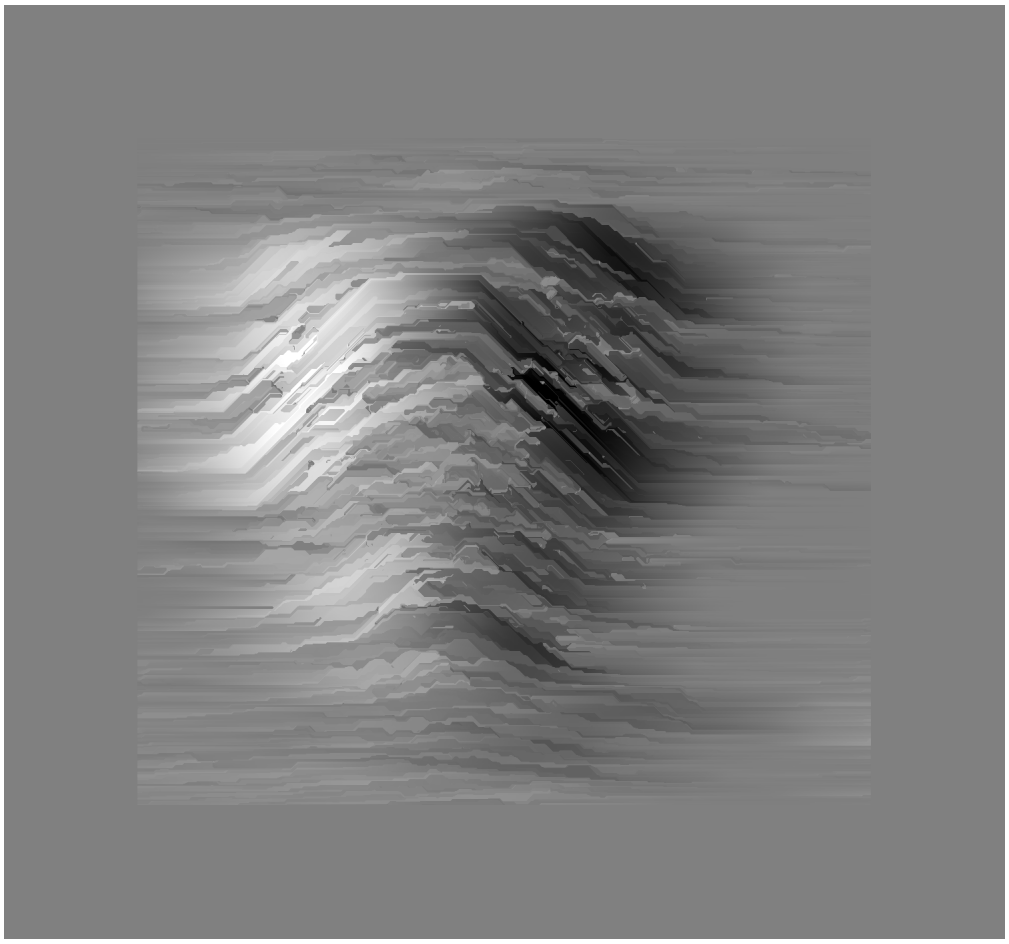


Figure 8.13.: First order derivative of y-component of discrete curve segments, estimated using the method of least squares parabola fitting using curve segments of 200 points in length. To improve readability, the intensities have been inverted and the contrast improved. This means that the light areas correspond to negative values, and the dark areas correspond to positive values.

in the y-direction.

Once the derivatives have been estimated, the curvature can be calculated. This is the topic of the following section.

8.2.3. Curvature

With a method for tracking curve segments and for estimating their derivatives in place, the actual curvature estimation is simply done by filling in the derivatives in the curvature expression found in Secion 5.2,

$$\kappa = \frac{y''x' - y'x''}{\left((x')^2 + (y')^2\right)^{\frac{3}{2}}}. \tag{8.11}$$

The only parameter that needs to be determined is the length of the curve segment, which is equivalent to the number of points used in the estimation. This will be will be discussed in the next section.

In order to create a curvature image of the saltdome image, rather than simply estimate the curvature in a single point, the estimation needs to be performed for every pixel in the image. So, the complete curvature image algorithm is as follows. For each image pixel, given a curve segment length N,

1. Extract a digital curve segment of length N, centered in the current pixel. This is done by tracking along a horizon using the method chosen in Section 8.1, $n = N/2$ points in the forwards and $n = N/2$ points in the backwards direction.

2. Calculate derivative estimates by fitting a parabola to the curve data points using least squares regression.

3. Calculate the curvature in the pixel using Equation 8.11.

The result of performing this algorithm for all pixels will be an image containing the curvature estimate in every point. Note that when creating the curvature image using centered curve segments as described above, the resulting image will be smaller than the original. There will be a border of width $n = N/2$ around the whole image, where the curvature can not be calculated, because it will not be possible to extract a centered curve segment for the outer border pixels.

8.2.4. Determining the length of curve segments

In the curvature estimation method outlined above, the only variable that needs to be determined is N, which is the length of the curve segment, or the number of points to be tracked. The value should be determined based on how long the curve

segment needs to be in order to accurately represent the underlying horizon. There are several considerations to make when choosing a value for this parameter.

The curve segment needs to be long enough to minimize the impact of any unevenness on the curve. If the curve segment is too short, there is a risk that unevenness and noise on the curve will interfere with the curvature estimation. This situation is illustrated in Figure 8.14a on the following page. The markers mark the end points of the curve segment being used in the estimation. The parabola, drawn in red, has been fitted to the points between the markers. Because of the unevenness in the curve the estimated curvature will be quite large, when in fact the general shape of the curve in this area is straight so the curvature should be close to zero. The risk of this situation occurring is minimized if a longer curve segment is used. In Figure 8.14b, the distance between the points is increased, reducing the impact the bend has on the parabola.

On the other hand, if the curve segment is too long, there is a risk of loosing important details in the shape of the reflector.

Determining the best value for N in this project was done experimentally. This will be reviewed in the next chapter.

8.2.5. Critique of method

In Section 5.5 it was noted that Coeurjolly, Miguet, and Tougne [3] consider fitting a continuous function to discrete data in order to apply continuous mathematics to be a bad choice, because it depends on picking the right parameter values and may be computationally heavy. Finding discrete versions of the mathematical expressions is considered a preferable approach. This is a fair point. However, as seen in the previous section, there is only a single parameter that needs to be determined in the presented method. As we will see in the following chapter, determining its value is not problematic. Because we are mainly interested in how the curvature changes in an area, rather than the actual values, the possible lack in accuracy is not an issue. Computational efficiency is, of course, of importance in general. However, this has not been a main concern in the work on this project.

8.2.6. Related work

Some of the work presented in Section 5.5 is similar to the method presented here.

Lewiner et al. [13] present a method for estimating curvature of discrete planar curves by approximating the local arc length in a point and using it to fit a parabola by the method of least squares and estimate its derivatives. This method is similar to the curvature estimation method presented here and could probably be used as a substitute method. However, the method is mathematically more complex. A comparison of the two methods would be interesting, but is unfortunately out of the scope of this project.

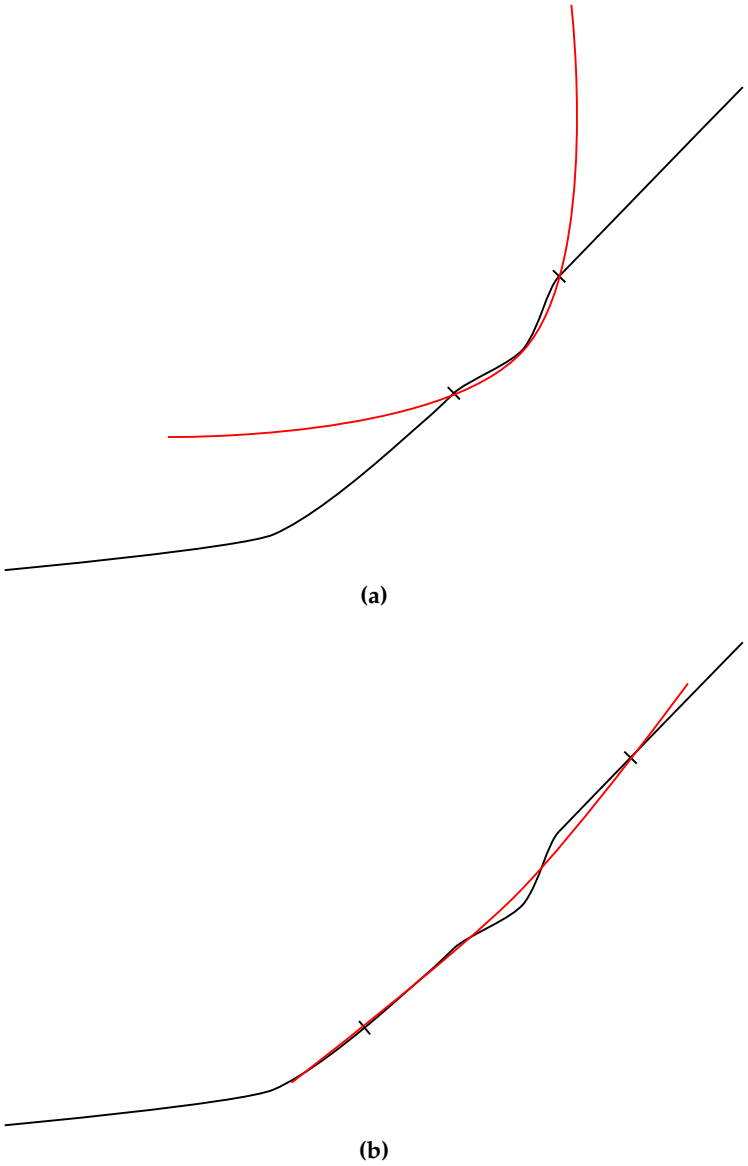


Figure 8.14.: Approximating the curve by fitting a parabola to the points between the two markers. In (a), the length of the curve segment is too short, and the bend on the curve impacts the resulting parabola. In (b), the length is large enough that the resulting parabola follows the general shape of the curve without being disturbed by the bend.

In his article, Roberts [20] studies curvature attributes for seismic data in particular. His work also involves fitting a quadratic surface to the data by the method of least squares. However, this work is done on three-dimensional horizons which have been manually interpreted by a geologist. The method presented in this project aims to interpret the horizons automatically on two-dimensional slices of the 3D data.

The two methods presented above, and some of the other methods presented in Section 5.5, could have been used for the curvature estimation in this project. However, what makes the method presented here unique is the combination of automatically tracking the horizons and curvature estimation, and using this to highlight saltdomes. To my knowledge, in practice this is currently done solely by manual interpretation. The aim has not been to find the best method for estimating curvature, but rather to find a complete method for automatically distinguishing saltdomes from their surroundings.

In the next chapter, the result of running the presented method on several saltdome images will be preseted.

9. Results

In the previous chapter, all the theoretical background from Part I was put together to form a method for estimating the pixel by pixel curvature of a digital seismic image. In this chapter, the results of applying this method to 2D slices of the dataset presented in Chapter 3 will be studied. The method will first be applied to a single curve, before expanding to estimating the curvature of the entire saltdome image.

9.1. Testing on a single curve

Before studying the curvature of the entire saltdome image, it is useful to see how well the method presented performs on a single curve. In an attempt to capture the saltdome shape that we are interested in, a curve is tracked across the entire image using the discrete curve tracking method chosen in the previous chapter. The starting point for the curve was determined experimentally to $(15, 700)$, as this starting point gave a complete saltdome curve. The result, after tracking pixel by pixel across the entire image in a forwards manner from this starting point, is the curve marked in red in Figure 9.1. The curve clearly captures the shape of the saltdome in the image.

9.1.1. Expectations

In order to have a way of assessing the result of curvature estimation for the single curve, we will take a moment to consider what the result could be expected to look like. As seen in Chapter 5, as well as saying how much the curve is bent in a particular point the curvature measure says in what way it is bent, whether it is locally convex or concave. This information is contained in the sign of the curvature. The sign convention for curvature was illustrated in Figure 5.2 on page 37. The concave anticline segments of the curve have positive curvature, while the convex syncline segments have negative curvature. Both the horizontal and the dipping plane curve segments have a curvature equal to zero.

Based on the curvature sign convention described above, the expected curvature pattern for a curve spanning across a saltdome in an inline seismic image is as follows:

1. A horizontal, flat segment with zero curvature.

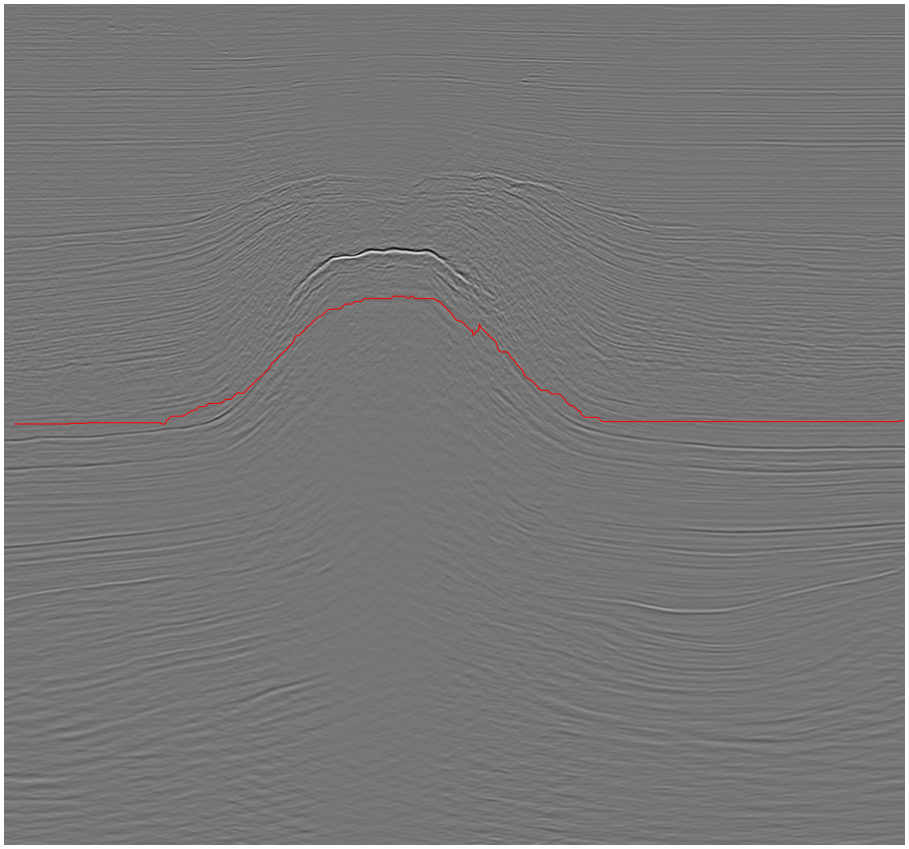


Figure 9.1.: Tracked curve starting in pixel $(15, 700)$.

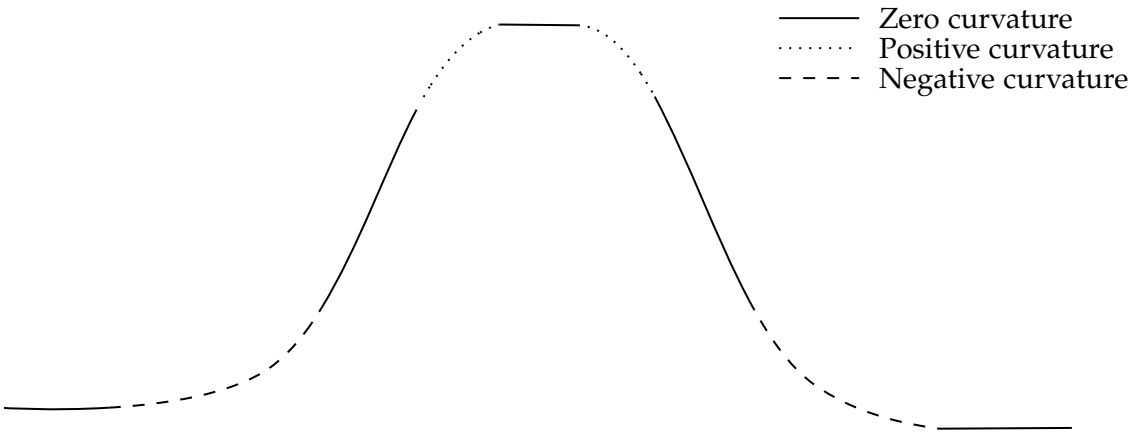


Figure 9.2.: Expectation for curvature of saltdome.

2. A convex segment where the horizon begins to dip upward. Negative curvature.
3. A flat, upward dipping segment with zero curvature.
4. A concave segment with positive curvature.
5. A flat, horizontal segment at the top of the saltdome. Zero curvature.
6. A concave segment with positive curvature.
7. A flat, downward dipping segment with zero curvature.
8. A convex segment with negative curvature.
9. A horizontal, flat segment with zero curvature.

This pattern is illustrated in Figure 9.2. The curvature is expected to be highest in absolute value in the sharpest parts of the bends on the curve, and approach zero curvature as the curve flattens out.

With these expectations in mind, we will study the actual result of calculating the curvature of the tracked saltdome curve.

9.1.2. Result

Applying the curvature estimation method from Chapter 8 to the curve in Figure 9.1 gives the result presented in Figure 9.4 on page 81. Figure 9.5 shows the right half of the curve enlarged, allowing for a more detailed study of the result. The number of points N used in the derivative estimation is 300.

To make the result as readable as possible the following colour scheme of red and green has been employed.

Figure 9.3.: Illustration of the colour scheme defined in Definition 9.20.

Definition 9.1 (Colour scheme for curvature)**.** Red and green represent positive and negative curvature, respectively. The darker the colour, the higher the absolute value of curvature. White represents curvature values close to zero.

The colour scheme is illustrated in Figure 9.3. Observe that the dark, almost black colours represent the highest curvature. The colours range from very dark, through red or green, to white, where the white areas have close to zero curvature. Note that a threshold has been set to determine which pixels belong in the zero curvature group. Because of this, there is not a smooth transition between the low curvature and zero curvature pixels.

Studying Figures 9.4 and 9.5, it is clear that the sign of the curvature is as expected. The convex curve segments in green have negative curvature, and the concave segments in red have positive curvature, perfectly in line with the expectation presented in Figure 9.2 on the previous page. Furthermore, the colours are at their darkest in the parts of the curve where the bend is sharpest, indicating, as expected, highest curvature in these parts. This is particularly clear in the green segment in Figure 9.5. Between the red and green segments, on either side of the saltdome, there are white "breaks" in the curve. These are also present at the far ends of the curve, and are areas with zero, or close to zero, curvature. This also corresponds well with the expectations in Figure 9.2. However, there was expected to be a zero curvature segment at the very top of the curve, which does not seem to be present. The curvature is positive throughout the top of the curve. This is due to the fact that the top of this particular curve is not flat like the top of the stylised, symmetric saltdome shape in Figure 9.2. Naturally, no tracked curve segments will be exactly as expected in shape. However, the idea is that when studying the entire image, rather than single curves, there will be large areas of pixels with the expected curvature values, even though there are some exceptions.

Another possible reason why the there are no zero curvature pixels at the top of the curve is that the number of points used in the derivative estimation is too large. In the particular case of Figure 9.4 the number of data points used in the estimation is 300. This is a fairly large number. However, experiments showed that decreasing this number did not result in a zero curvature area at the top of the

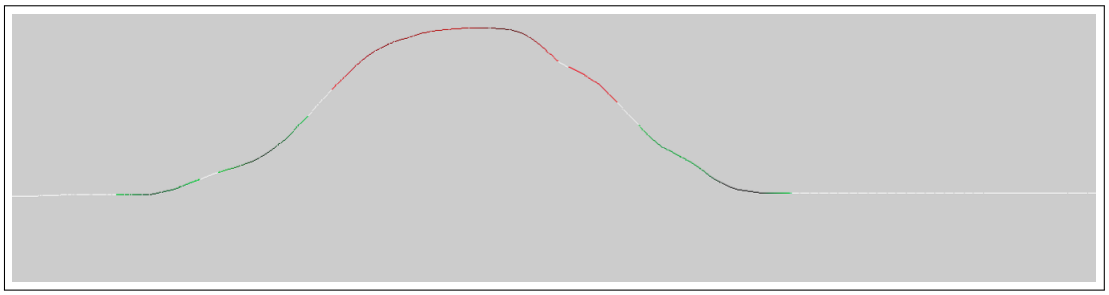


Figure 9.4.: Curvature of the curve from Figure **??**. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

curve. The only result was more noise. This may, however, be relevant for other curves, and is worth keeping in mind when determining the parameter value.

For the purpose of comparison will study another single saltdome curve, with a slightly different shape from the previous curve. This curve is tracked starting in pixel (15,600) and is displayed in Figure 9.6 on page 83. The curve with curvature values represented using the same colour scheme as before can be found in Figure 9.7. The number of points used in the derivative estimation is 300, just as in the above example. This curve has a wider top than the previous curve, and there is a small zero curvature segment present. There are more "breaks" of zero curvature in the sides of this curve than in the previous curve. Overall though, the curvature is well in line with the expectations presented in the previous section and in Figure 9.2.

To summarize, the curvature estimation method presented works as expected. The resulting curvature is highest in the points with visually higher curvature, and approaches zero as the curve flattens out. The expectation of a zero curvature area in the top of the curves turns out to be somewhat unrealistic, as the individual tracked curves will vary in shape. A better way to formulate the expectation is that the saltdome curves will have a concave shape in the top area which may flatten out and approach zero curvature in the center.

In the following section we will estimate and study the curvature of the entire saltdome image.

9.2. A curvature image

Having estimated the curvature for a single curve, the next step is to calculate the curvature for every pixel in the saltdome image. The result will be an image of curvature, hopefully with distinct positive and negative curvature areas which stand out from the rest of the image.

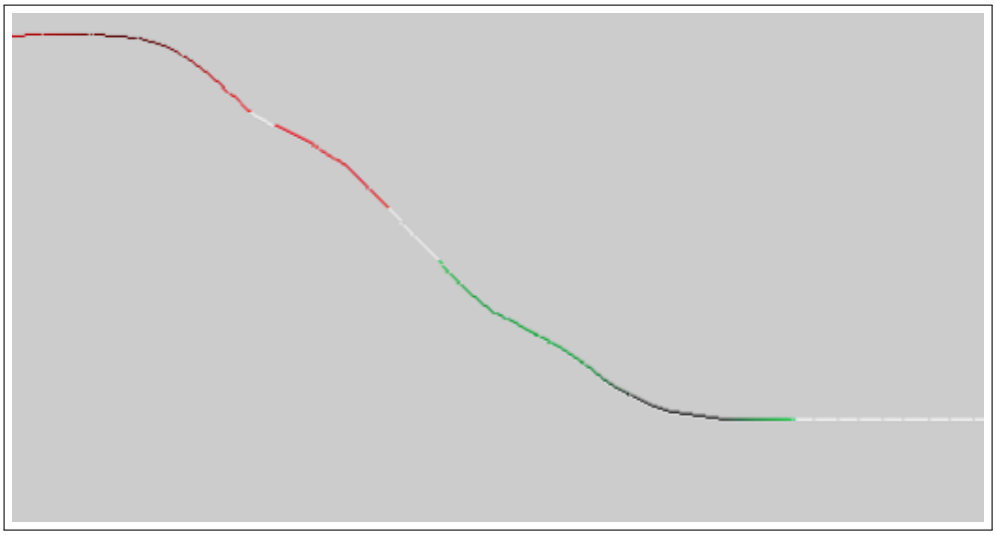


Figure 9.5.: An enlarged section of Figure 9.5. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

9.2.1. Expectation

As with the single curve in the previous section, we will take a moment to consider what results can be expected when calculating the curvature for the entire saltdome image. Hopefully the curvature image will have the same pattern of positive, negative and zero curvature as with the single curves. However, when looking at the entire image rather than a single curve, the areas with similar curvature will be larger, and there is a greater chance of the presence of noise interfering with the result.

The expected result from the curvature calculation in certain areas is illustrated in Figure 9.8. For readability in the figure, only the non-zero curvature areas on the saltdome have been marked. In the rest of the image curvature is expected to be close to zero, although the non-zero areas may be slightly larger than the boxes marked in the figure. The expected curvature pattern is the same as for the single curves, only with larger areas. Based on the observations from the previous section, the top area of the saltdome may be positive everywhere, with no zero curvature area in the center. However, the curvature values are expected to be higher in the corner areas marked with 2 and 3, than in the center.

The area within the saltdome has a chaotic, non-uniform texture. In this area the tracking of curve segments will probably result in random curve segments and many high curvature points, both positive and negative.

With expectations of the curvature image in place, we are ready to study the actual result of applying the curvature estimation method to the entire image.

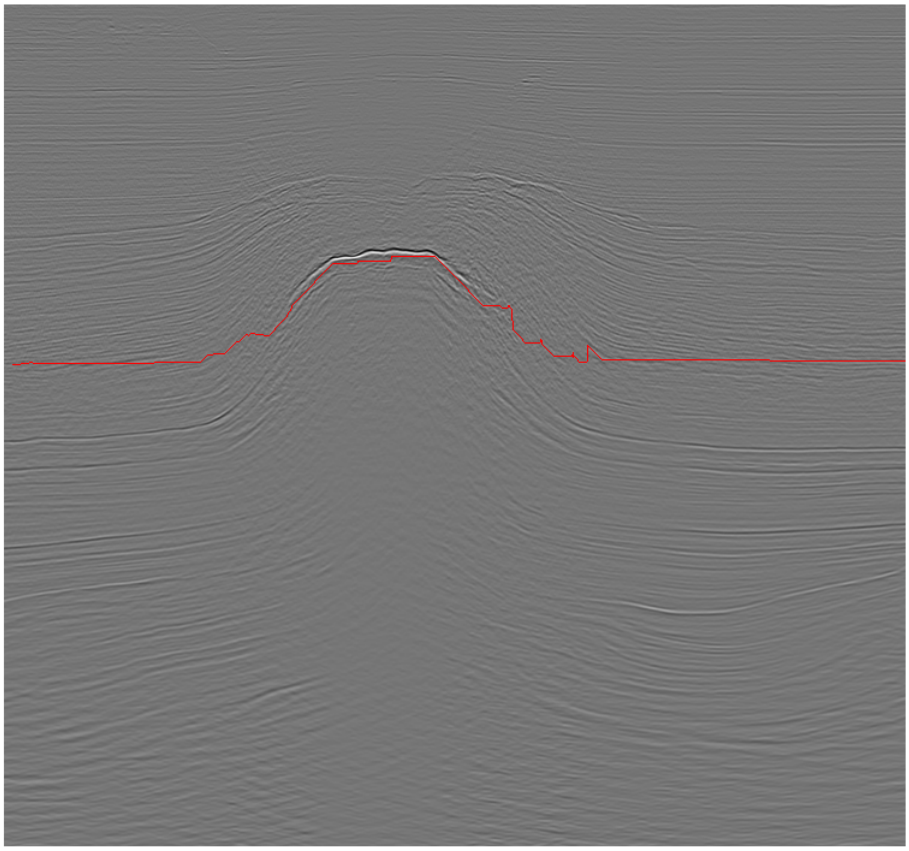


Figure 9.6.: Tracked curve starting in pixel $(15, 600)$.

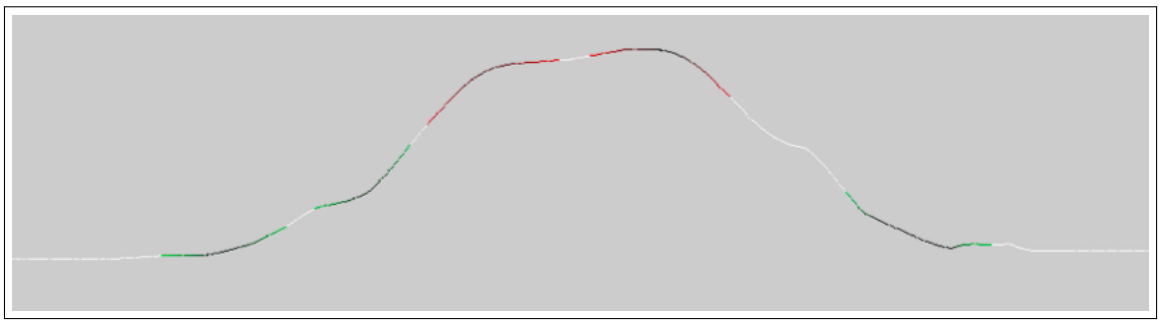


Figure 9.7.: Curvature of the curve in Figure **??**. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

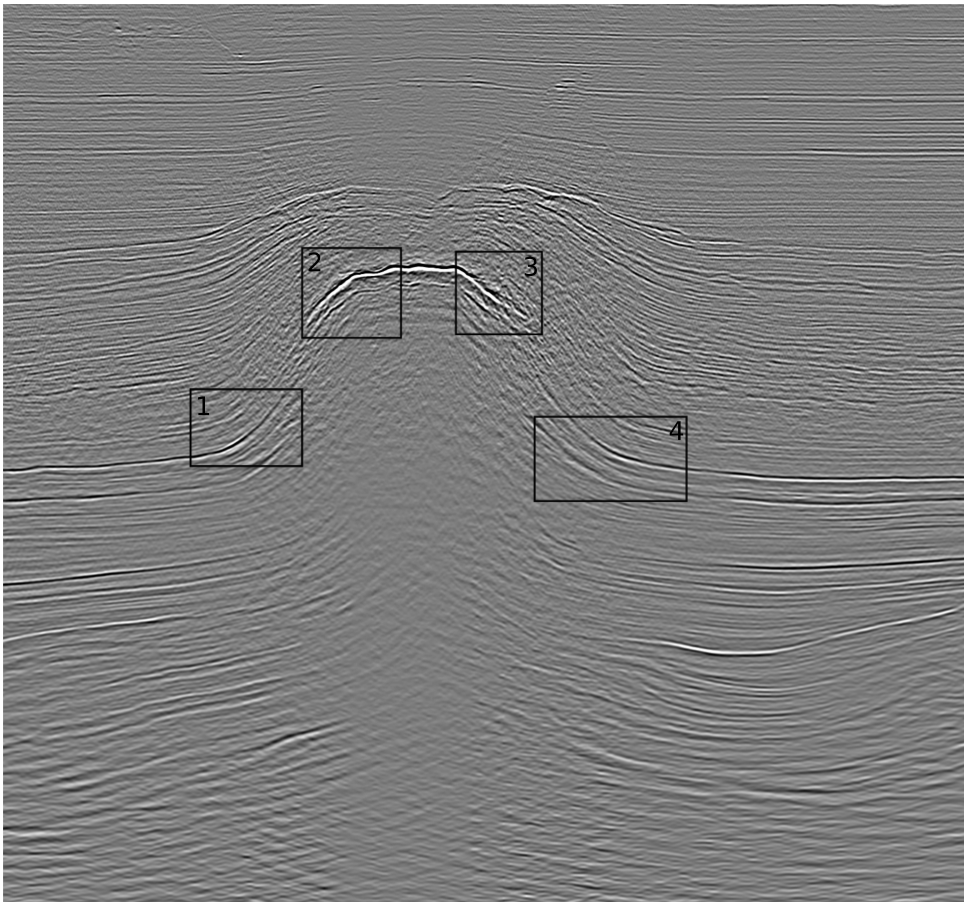


Figure 9.8.: High curvature areas. Expectation for curvature:

1. Negative
2. Positive
3. Positive
4. Negative

9.2.2. Result

The method for estimating the curvature of an entire seismic image was presented in Chapter 8. The result of applying the method is an image of the curvature of the input image. As noted in Chapter 8, because the curve segments are centered, there will be a border along the edge of the image where the curvature can not be estimated.

As discussed in Subsection 8.2.4, the length of the tracked curve segments, N, which translates to the number of points used in the derivative estimation, is the only parameter in this algorithm. We will now study the curvature images resulting from letting the parameter take the values 100, 200, 300 and 400, shown in Figures 9.9, 9.10, 9.11 and 9.12, respectively.

The colour scheme in the curvature images is the same as the scheme presented in Definition 9.20 and Figure 9.3, and used in the presentation of the individual curves in the previous section.

Estimating curvature based on curve segments of length 100 points results in the curvature image in Figure 9.9 on the next page. The image does have areas of coherently positive and negative curvature values. Particularly the green, negative, curvature areas at the beginning and the end of the saltdome are fairly large and uniform. However, there is a lot of noise present in the image, both within the saltdome, and in the surrounding area. Also, there are hardly any areas of white, zero curvature, pixels within the saltdome. Although the desired pattern in curvature is partially present, this image does not highlight the saltdome to a satisfactory extent. This indicates that the number of points used in the estimation should be more than 100.

Increasing the number of points to 200 resulted in the curvature image in Figure 9.10 on page 87. This is a clear improvement from the 100 point image in that there are much larger coherent areas of positive and negative curvature pixels in the expected locations. At the top of the saltdome there is a large area of positive curvature, with high curvature values in the corners and lower values in the center, as expected. However, there is still some noise present in the non-saltdome area, in the form of non-zero curvature pixels. Also, the expected areas of zero curvature in the sides of the saltdome are not present. Rather, these areas have a chaotic pattern of curvature, with a combination of positive and negative curvature values.

Figure 9.11 on page 89 shows the result of using 300 points for curvature estimation. The image is somewhat similar to Figure 9.10. However, one difference is the larger variation in curvature values, particularly in the green, negative curvature areas. Also, there are more zero curvature pixels in the sides of the saltdome than seen previously. Furthermore, there is less non-zero curvature noise present in the surrounding area, compared to in the previous image. The saltdome area clearly stands out from the rest of the image. This can be observed in the lower part of the image in particular. However, increasing the length of the tracked curves has

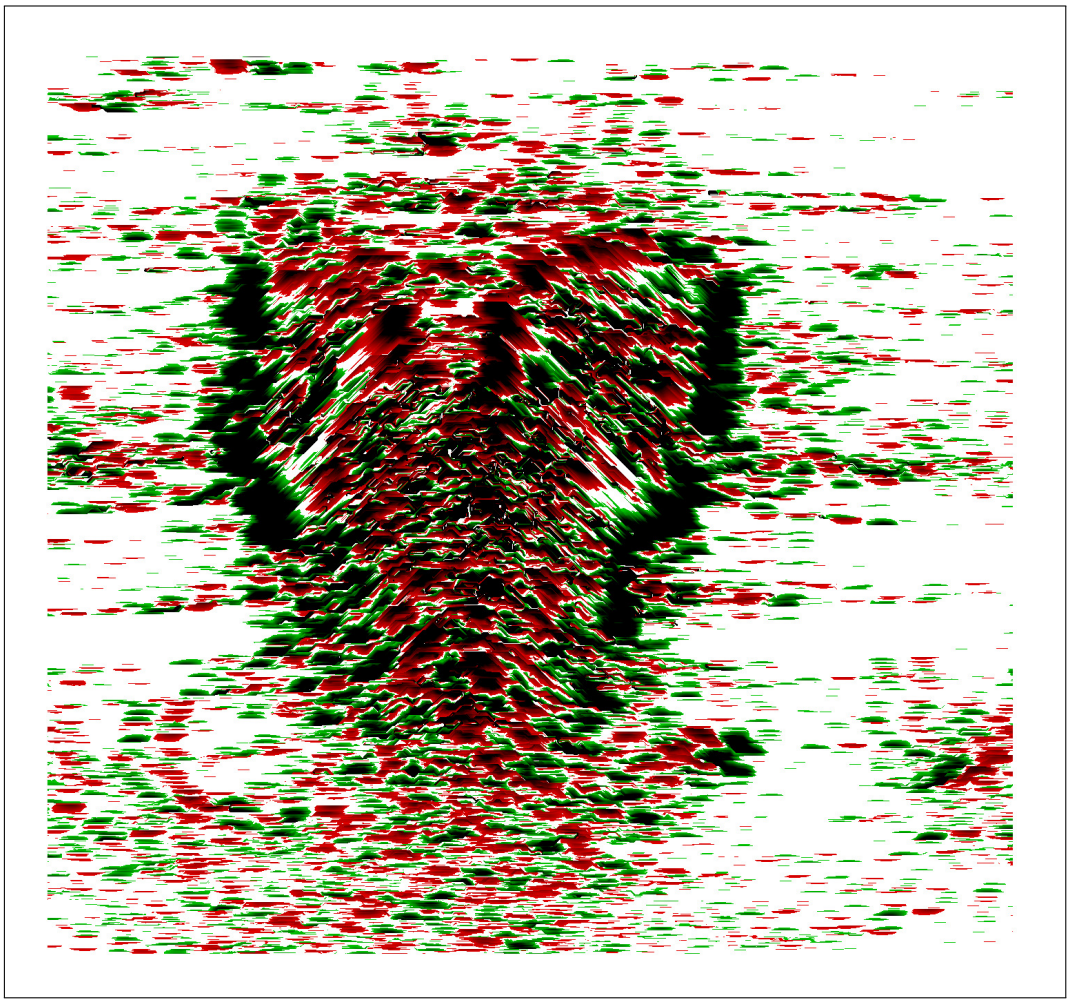


Figure 9.9.: Image of curvature, where the curve segments used for curvature estimation are 100 pixels long. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

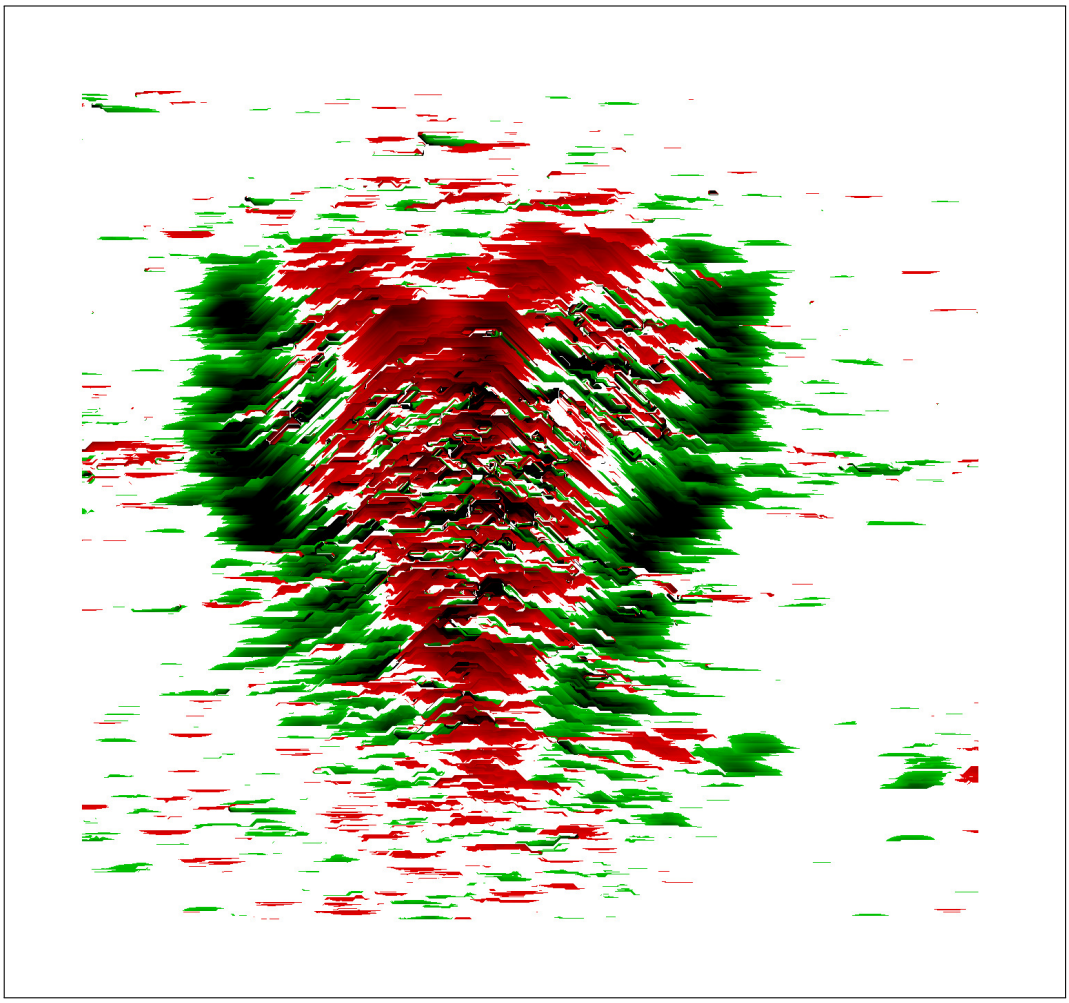


Figure 9.10.: Image of curvature, where the curve segments used for curvature estimation are 300 pixels long. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

increased the width of the border where curvature cannot be calculated.

Finally we will consider the curvature image based on 400 point curve segments, displayed in Figure 9.12 on page 90. The most notable difference between this image and the image in Figure 9.11 is that there are fewer pixels which are very dark in colour, with high curvature. This is particularly clear in the negative curvature areas. Furthermore, in the top of the saltdome the curvature peaks in the center rather than to the sides as in previous images. This is because the curve segments are long enough to capture the entire top of the saltdome in a single curve. The noise in the surrounding area that has not been cropped by the increased border is further reduced.

9.2.3. Testing on other images

All work so far has been based on the same inline image, displayed in Figure 8.1 on page 56. For a more complete analysis of the proposed method, the method is applied to two different inline images. The new images are timeslices 750 and 1150 from the north sea dataset, and can be found in Figures 9.13 and 9.14, respectively. Observe that both images contain a saltdome with roughly the same shape as in the original image. However, in Figure 9.13 the top of the saltdome is quite flat, and also slightly tilted to the right. In Figure 9.14 the saltdome appears more symmetric.

The results of estimating the curvature for the image in Figure 9.13 using curve segments of lengths 200, 300 and 400 points are displayed in Figures 9.15, 9.16 and 9.17, respectively. The corresponding results for Figure 9.14 can be found in Figures 9.18, 9.19 and 9.20. Images based on curve segments of length 100 have not been included, due to the poor result obtained using this curve length for the original image, see Figure 9.9.

First consider the curvature images based on 200 point curves, in Figures 9.15 and 9.18. There are clear coherent areas with positive and negative curvature present in the images. The amount of noise present in the area surrounding the saltdome is similar to that in the corresponding result for the original image. However, note that in the inline 1150 image in Figure 9.18 the zero curvature areas in the sides of the saltdome are much more distinct than previously.

Next, consider the 300 point based curvature images in Figures 9.16 and 9.19. The noise in the surrounding areas has been greatly reduced. Furthermore, as observed with the original image, there is more variation in the curvature values, and the curvature reaches its highest values in the expected locations. Again, the inline 1150 image has very clear zero curvature areas in the sides of the saltdome. These coherency of these areas is improved compared to in the 200 point image.

Finally, Figures 9.17 and 9.20 show the results of using curve segments of 400 points in the curvature estimation. As expected, the surrounding noise has been further reduced, particularly for the inline 1150 image. The variation in curvature

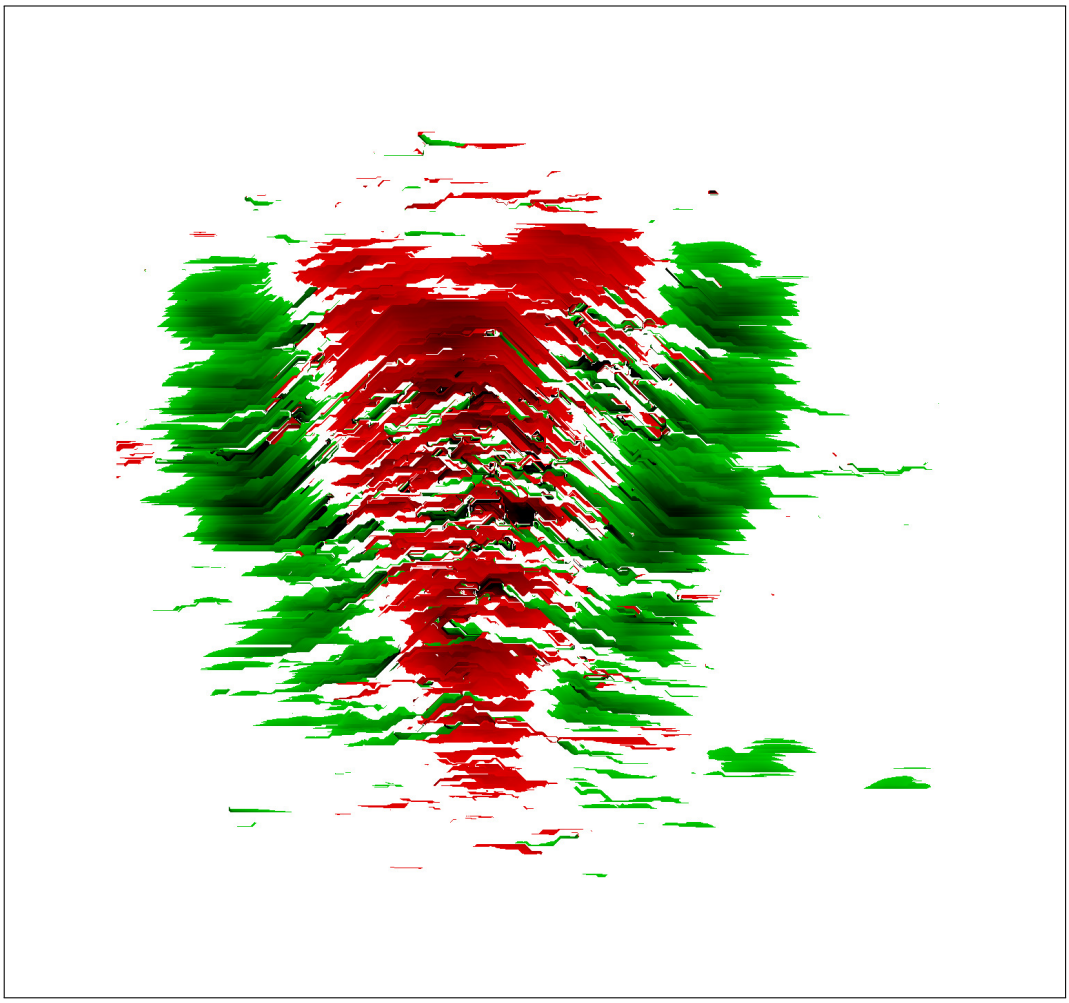


Figure 9.11.: Image of curvature, where the curve segments used for curvature estimation are 300 pixels long. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

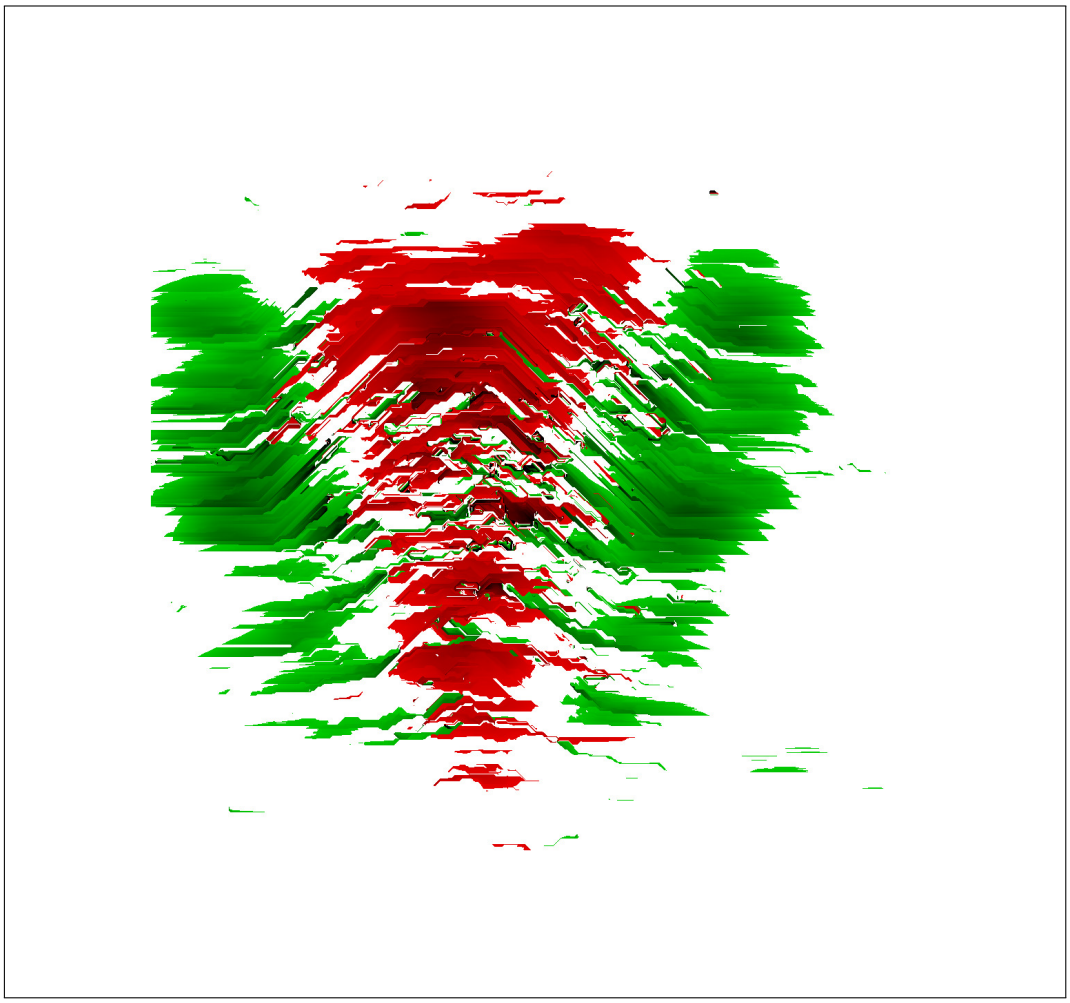


Figure 9.12.: Image of curvature, where the curve segments used for curvature estimation are 400 pixels long. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

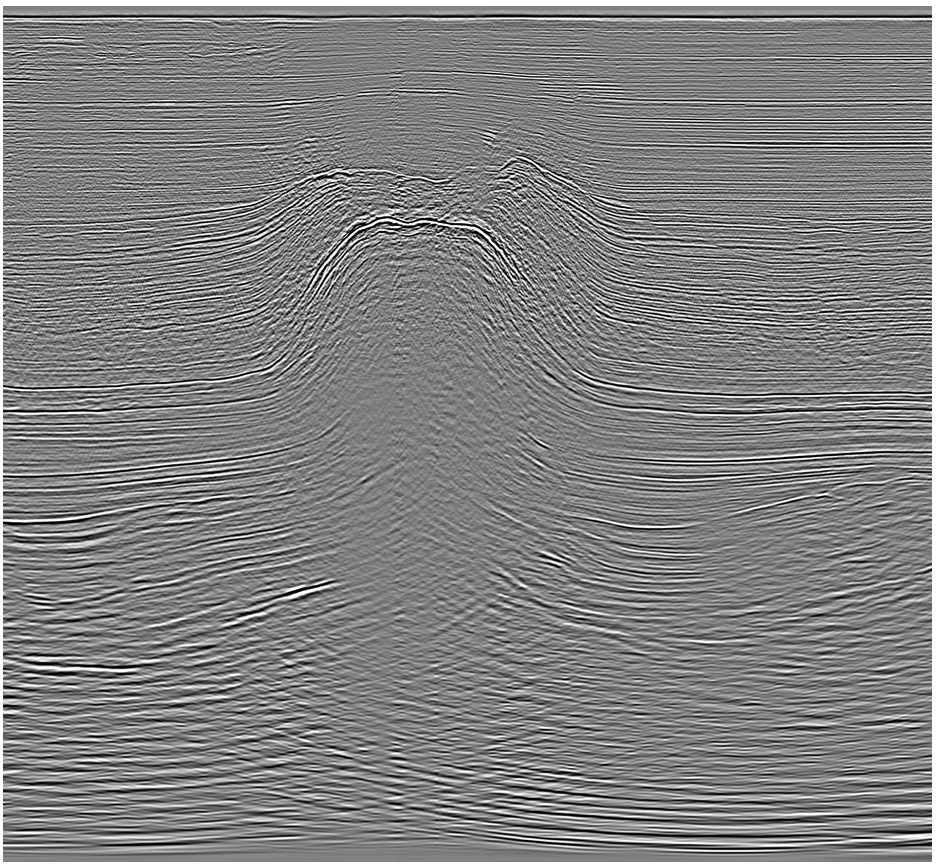


Figure 9.13.: Inline slice 750 from the north sea dataset.

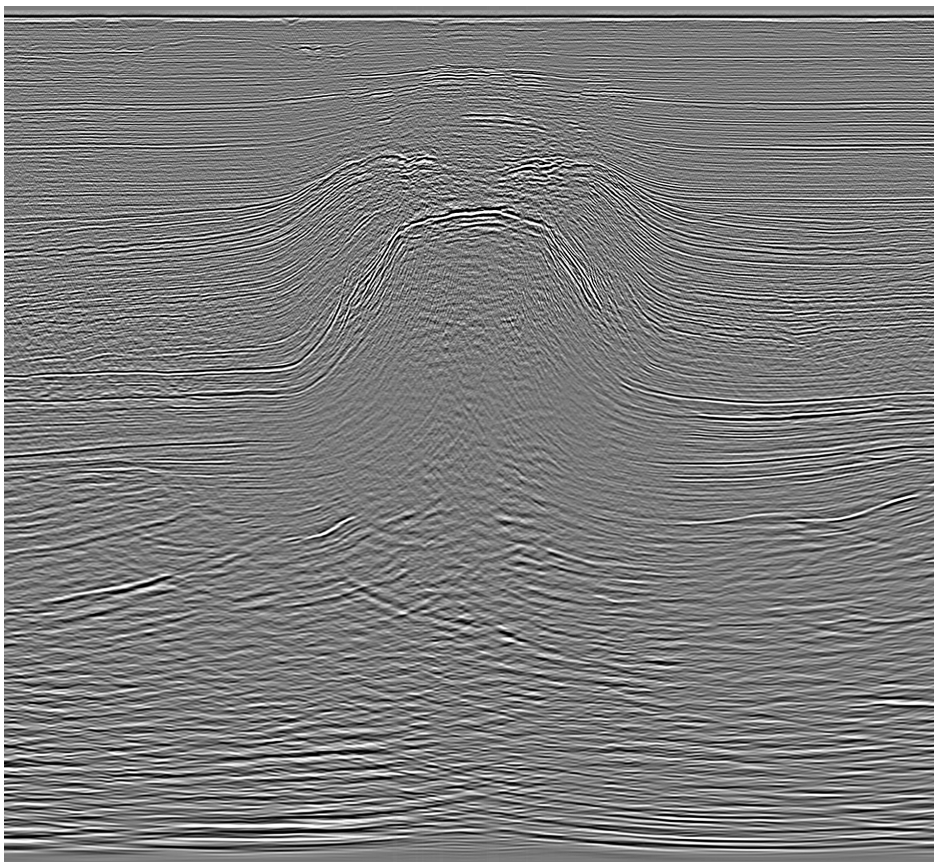


Figure 9.14.: Inline slice 1150 from the north sea dataset.

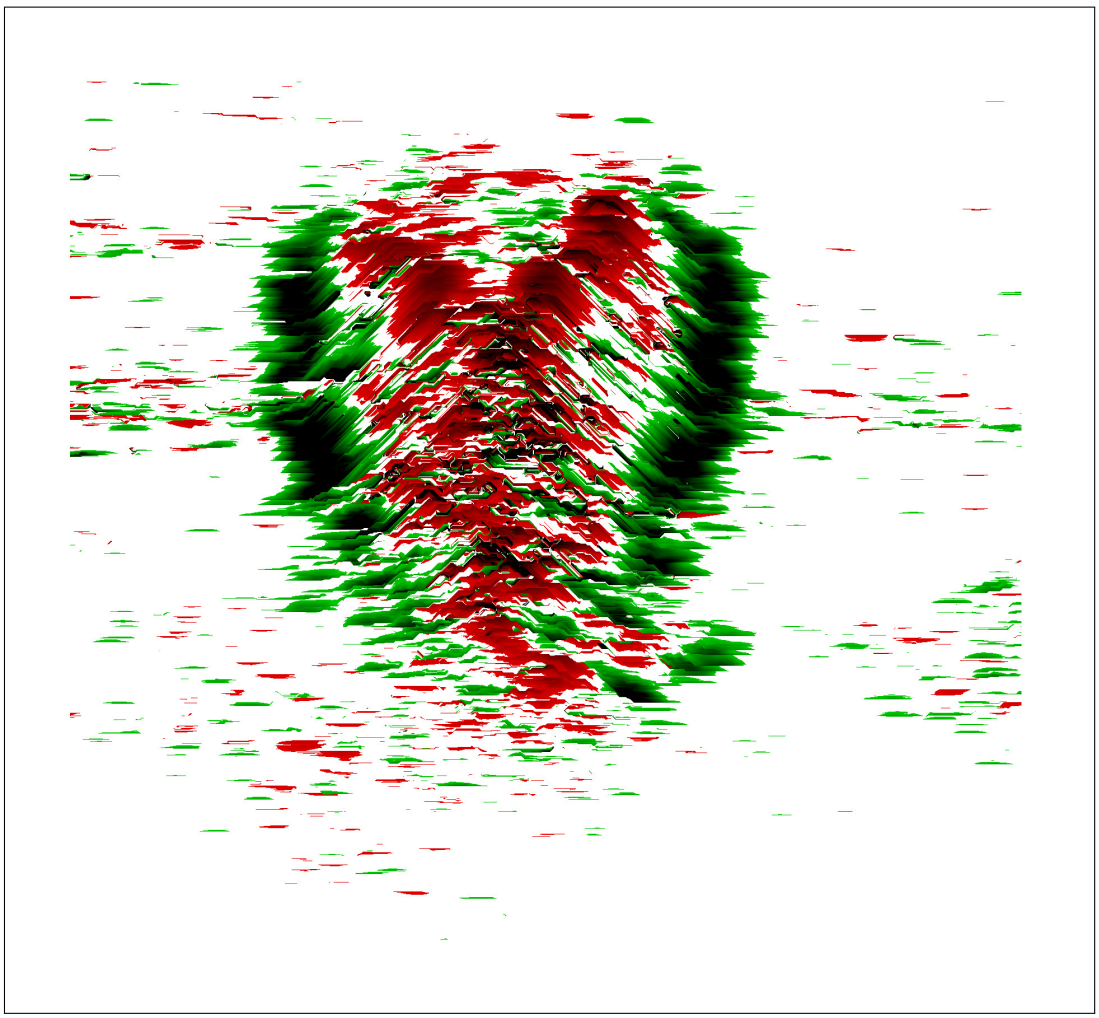


Figure 9.15.: Curvature image of inline 750 from Figure 9.13 on page 91, obtained using curve segments of length 200. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

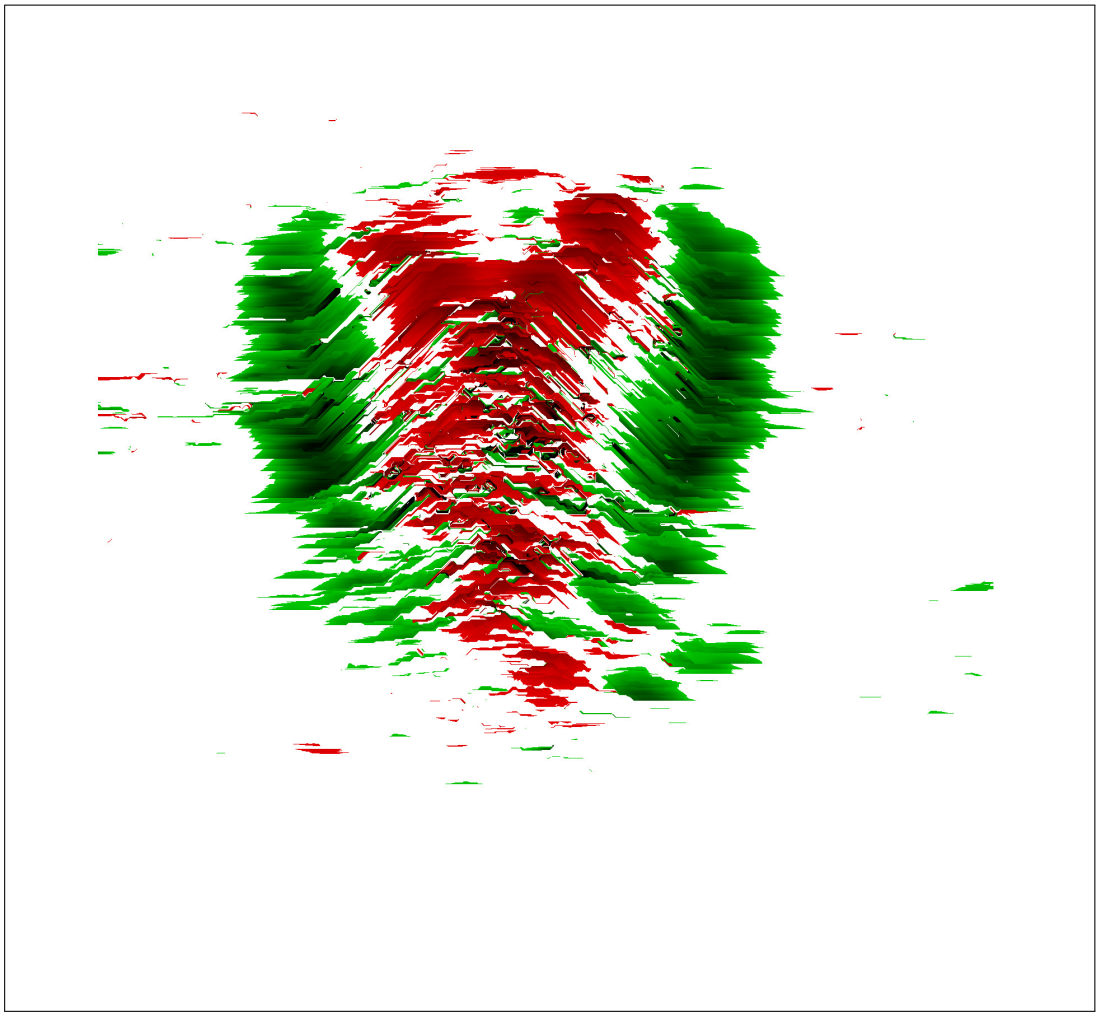


Figure 9.16.: Curvature image of inline 750 from Figure 9.13 on page 91, obtained using curve segments of length 300. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

values is less than in the 300 point based results, just as was observed in the original. The extent of the coherent positive and negative curvature areas is equivalent to that in the 300 point based images.

To summarise, the curvature image estimation method presented in the previous chapter works consistently well for three different inline saltdome images. This gives reason to believe that it would work for other data sets as well. Continuous areas of positive and negative curvature are present in the expected locations, as presented in Figure 9.8, in all the images. However, the images based on 100 and 200 point curve segments contain a considerable amount of noise in the area surrounding the saltdome. The images created using segments of length 300 and 400 points, on the other hand, contain considerably less noise. The images based on 300 point segments have a greater variation in curvature value in the high curvature areas than the 400 point images, and obtain higher maximum curvature.

Figure 9.17.: Curvature image of inline 750 from Figure 9.13 on page 91, obtained using curve segments of length 400. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

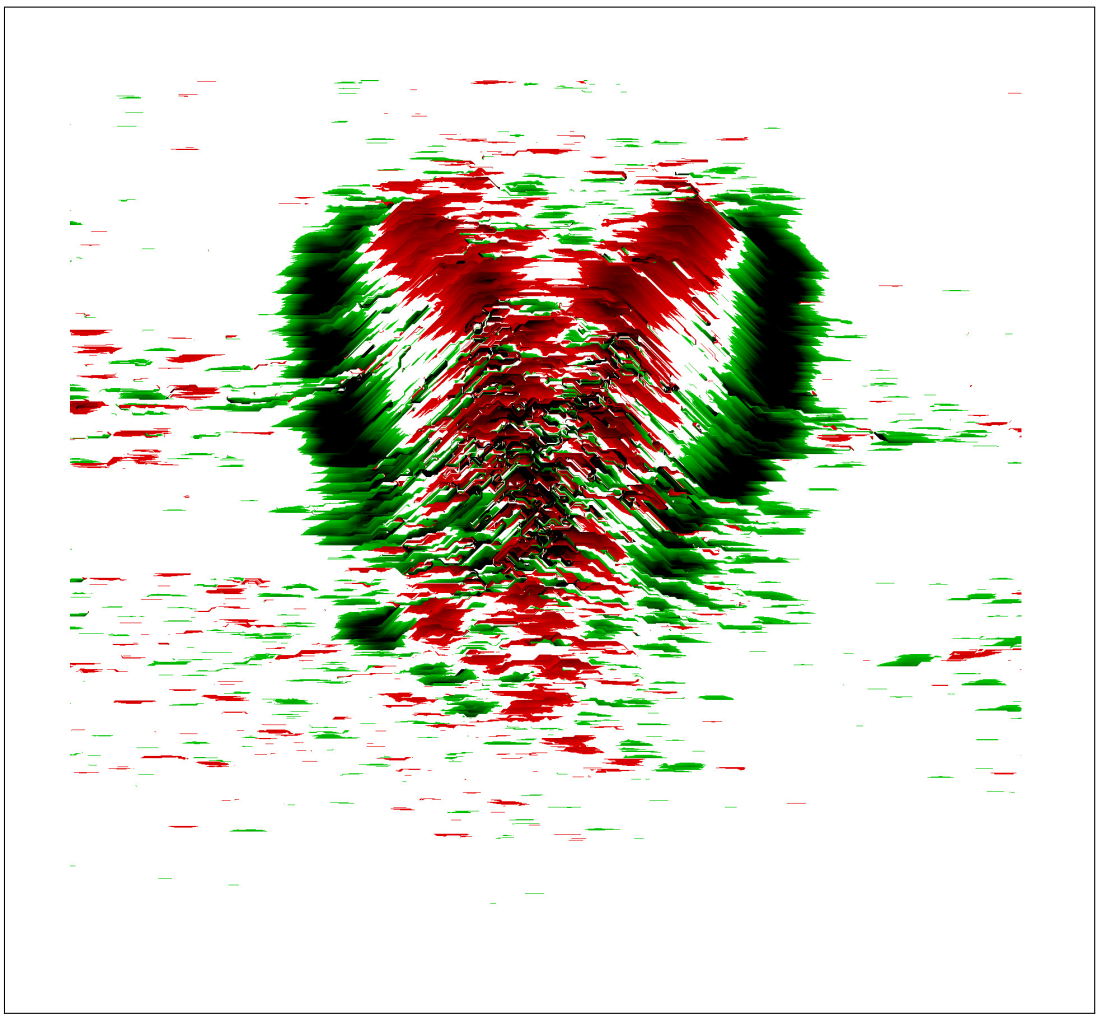


Figure 9.18.: Curvature image of inline 1150 from Figure 9.14 on page 92, obtained using curve segments of length 200. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

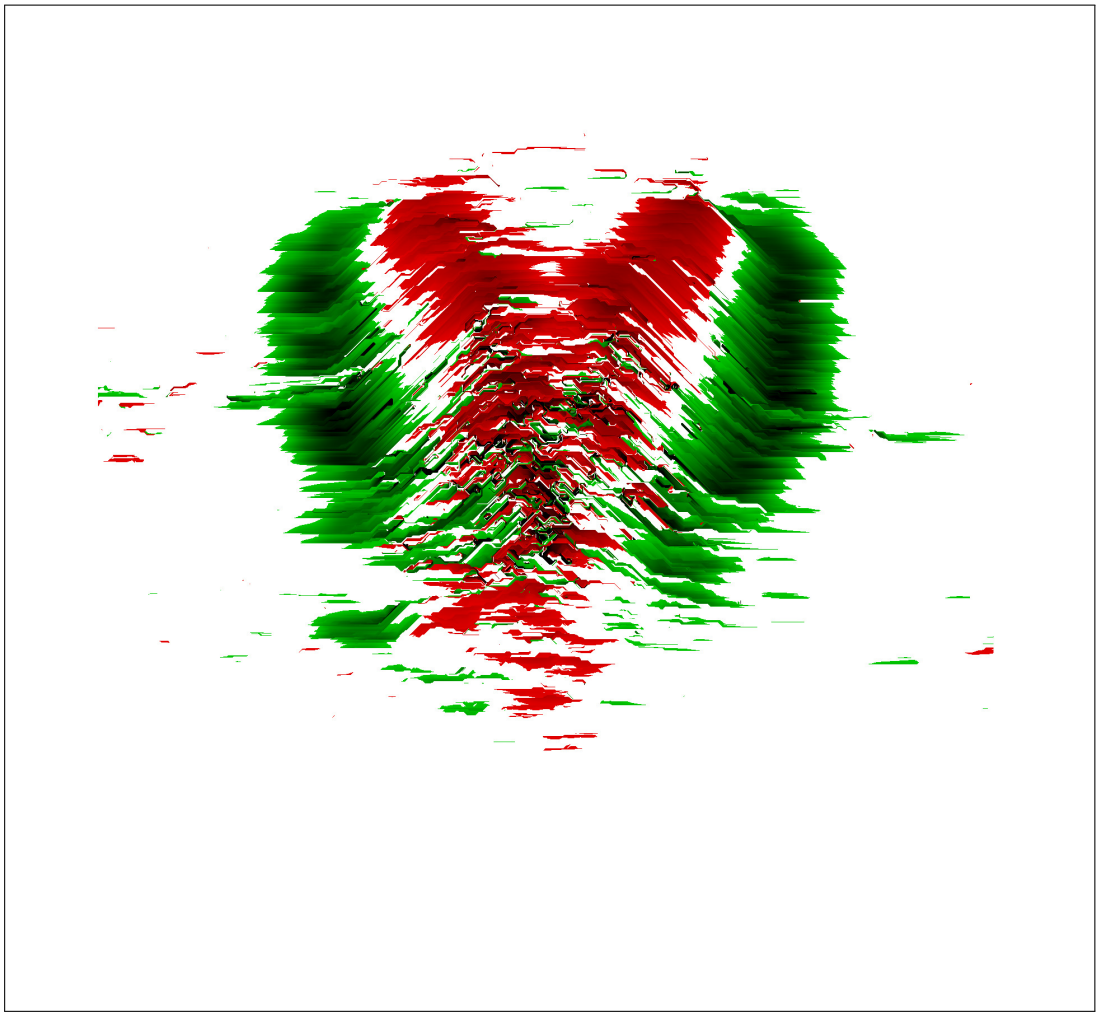


Figure 9.19.: Curvature image of inline 1150 from Figure 9.14 on page 92, obtained using curve segments of length 300. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

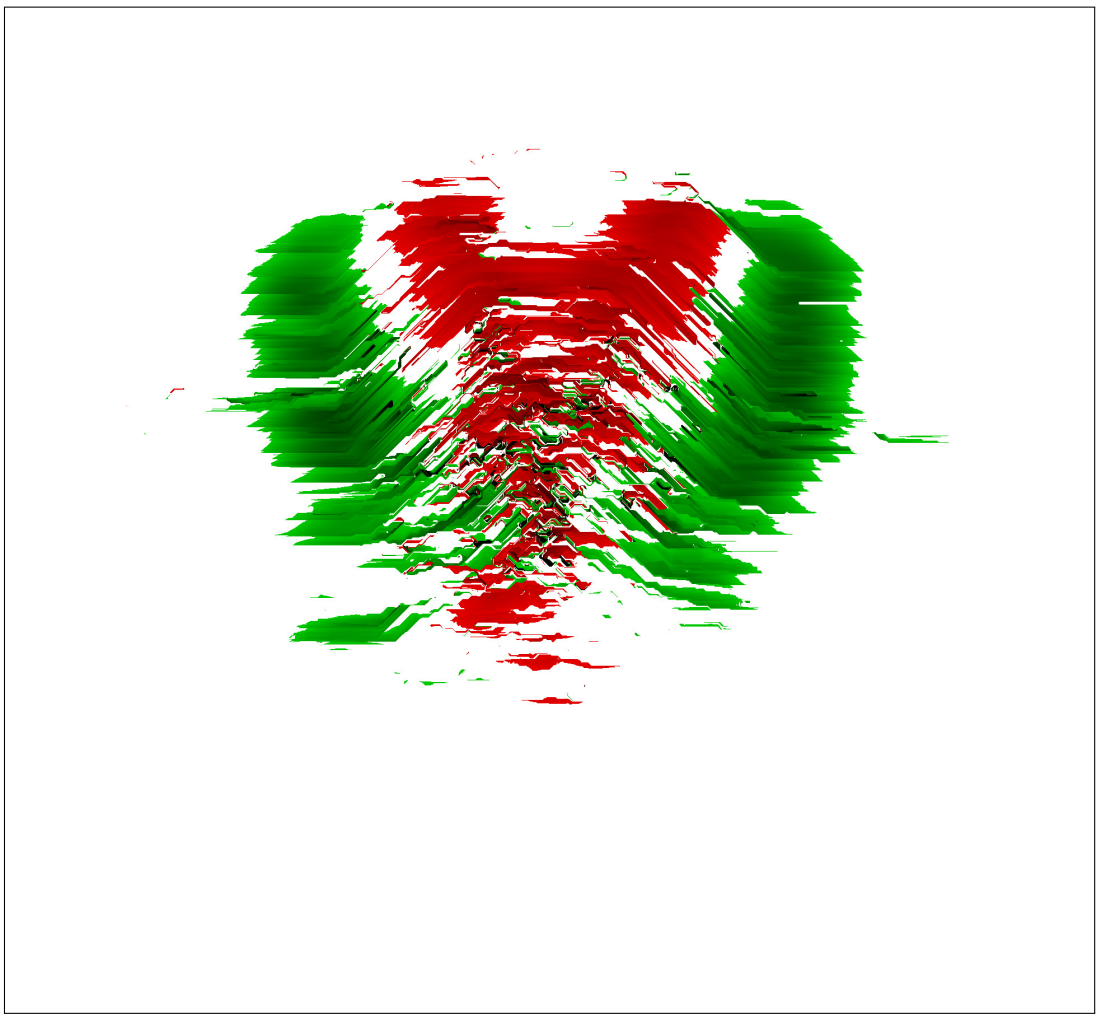


Figure 9.20.: Curvature image of inline 1150 from Figure 9.14 on page 92, obtained using curve segments of length 400. The colour scheme used is defined in Definition and illustrated in Figure 9.3.

10. Concluding remarks

This chapter starts by giving a brief summary of the method and results presented in this project. This is followed by a few thoughts on possible improvements and further work.

10.1. Summary of method and results

The aim of this project has been to develop a method for highlighting curved structures, saltdomes in particular, in two-dimensional sections of seismic data. This has been done using dip and curvature attributes. Given the dip image and the length of the curve segments, N, to be used in the curvature estimation, the method presented traverses the image and performs the following operations for every image pixel:

1. Using the dip image, track $n = N/2$ pixels forwards and backwards across the image, extract a discrete curve segment representing the horizon the pixel belongs to.

2. Based on the extracted curve segment, estimate the first and second derivatives in the pixel by fitting a parabola to the curve using the method of least squares, and calculating the derivatives of this parabola.

3. Calculate the pixel curvature using the derivative estimates.

The result of performing these steps for every image pixel is an image where every pixel holds the curvature for that point, except for a border of width $N/2$ where the curvature cannot be estimated.

The number of points N used in the least squares parabola fitting is the only parameter that needs to be determined. As seen in Chapter 9, the choice of value for N depends somewhat on the desired properties of the result. One may prioritise a high variation in curvature values in which case, for a given example image, setting $N = 300$ was shown to give the best result. If, on the other hand, the highest priority is to minimize the number of non zero curvature pixels outside the saltdome area, setting $N = 400$ gives the best results. Parameter values lower than 300 resulted in a considerable amount of noise both within the saltdome and in the surrounding area.

The saltdome highlighting method outlined above was tested on three different inline cross section images of the saltdome, using a variety of parameter values. The method proved to perform well, and gave results well in line with the expectations. In the result images presented in Chapter 9, the saltdome area clearly stands out from its surroundings in all three inline sections, and there are clear areas of positive and negative curvature values in the expected locations.

The presented results indicate that curvature based on automatically tracked curve segments would be useful as part of a set of features for saltdome classification. However, note that in order to uniquely identify the curved shape of the saltdomes, the curvature feature would benefit from being combined with the dip. This would mean that, for example, the leading convex segment of a saltdome would be identified with negative curvature and positive dip. Such an extension is easy to implement, but was not included in this project as the focus has been saltdome highlighting rather than classification.

10.2. Limitations and further work

Although the developed method for curve detection and saltdome highlighting is considered successful it does have some limitations, and improvements could certainly be made. This section will study a few of these and consider possible extensions and further work.

10.2.1. Extending to three dimensions

As discussed previously in the project, the seismic data is originally three dimensional. Having highlighted saltdomes in the two-dimensional data sections, a natural next step would be to unite these to form three dimensional result data. This would consist of running the saltdome highlighting method on a large set of both inline and crossline cross sections, and combining the results. For a review of the difference between inlines and crosslines, refer to Section 2.4. Such a three dimensional visualisation has not been a part of this project, as the focus has been on the mathematical and computational elements rather than visualisation of results. However, a three-dimensional visualisation would be an interesting and necessary element in further work on the subject.

10.2.2. Classification

One of the most relevant applications of the saltdome highlighting feature developed in this project is as part of a set of features in a classification algorithm for seismic structures. In such a classification the saltdome highlighting curvature feature would be combined with several other features based on, for example, shape

and texture. Texture would be of particular importance because of the characteristic random texture that is present within saltdomes. The saltdome highlighting curvature feature could be used in the pixel-based form presented here. However, a better alternative may be to create a window based feature using the curvature attribute. A few examples of possible window-based curvature features are the variation in curvature values, total summed curvature, or the total number of non zero curvature pixels within the window, to name a few examples.

A possible alternative approach could be to traverse the data in a certain direction, and consider the data to be in one of a set of different states. The states could be zero curvature, positive curvature and negative curvature, much as illustrated in Figure 9.8 on page 84. Combined with the sign of the dip, the sequence of states could be used to uniquely classify the data, as described in the previous section. The classification step could be performed using a hidden Markov model (HMM), much in the same way as done for traffic surveillance by Eikvil and Huseby [5].

10.2.3. Non-uniform dip intervals

As mentioned in Section 8.1, a weakness in the horizon tracking method presented is that the dip angle intervals which decide which neighbouring pixel to move to are uniformly distributed. This may not be the best approach because certain details in the horizons may not be captured. For an example of this, refer to the tracked curve in Figure 8.8a on page 64. The left part of the curve is completely horizontal and does not seem to follow the structure in the horizons which are slightly upward dipping. This is due to the low dip values in this area, and may indicate that the size of the interval of dip values resulting in a horizontal move should be reduced. Finding the optimal division of dip values would have been an interesting extension to this project.

10.2.4. Other seismic shapes

Another subject that would be interesting to explore is the use of the presented method for highlighting other seismic structures of interest besides saltdomes. An example of such a structure is *channels*. Channels are convex depressions in the sedimentary layers which have a shape similar to an upside-down saltdome. This shape indicates that the method presented in this project may indeed perform well in channel detection. During the work with this project the presented method was briefly tested on inline images of channels, but with little success. Positive, negative and zero curvature areas were present in the results, but did not display a clear pattern as in the saltdome images. The main challenge was that the dip values were not strong enough to give a successful horizon tracking. In order to work for channels, the horizon tracking method would have to be adjusted in some way, either by improving the dip estimation or replacing it with some other

pixel relation measure. Changing the size of the angle intervals, as described in the previous section, may also improve the results. As the focus in this project has been on saltdomes, such adjustments have not been a priority. However, this would certainly be an interesting area for further research.

10.2.5. Comparison with previous work

A comparison of the performance of several curvature estimation methods on the automatically tracked seismic curves would have been an interesting extension to this project. Such a comparison was not performed mainly because the work here treats automatically tracked horizons, and as such differs in nature from other methods. However, this would be an interesting topic for further work.

Bibliography

Printed sources

[1] Angélique Berthelot, Anne H. Solberg, and Leiv J. Gelius. "Texture attributes for detection of salt".

[2] Angélique Berthelot et al. "Salt diapirs without well defined boundaries - a feasibility study of semi-automatic detection". In: *Geophysical Prospecting* 59 (2011), pp. 682–696.

[3] David Coeurjolly, Serge Miguet, and Laure Tougne. "Discrete Curvature Based on Osculating Circle Estimation". In: *Lecture Notes in Computer Science* 2059 (2001), pp. 303–312. URL: http://link.springer.com/chapter/10.1007/3-540-45129-3_27.

[4] Manfredo P. Do Carmo. *Differential Grometry of Curves and Surfaces*. Prentice-Hall, Inc., 1976.

[5] L. Eikvil and R. B. Huseby. "Traffic Surveillance in Real-time using Hidden Markov Models". In: 2001.

[6] Patric J. Flynn and Anil K. Jain. "On Reliable Curvature Estimation". In: *Computer Vision and Pattern Recognition, 1989. Proceedings CVPR '89., IEEE Computer Society Conference on* (June 1989), pp. 110–116.

[7] Karl Friedrich Gauss. *General Investigations of Curved Surfaces of 1827 and 1825*. The Princeton University Library, 1902.

[8] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. 3rd ed. Pearson Education, Inc., 2008.

[9] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision Volume I*. Addison-Wesley Publishing Company, Inc., 1992, pp. 575–577.

[10] Richard Hoffman and Anil K. Jain. "Segmentation and classification of range images". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 9 (1987), pp. 608–620.

[11] Pascal Klein, Loic Richard, and Huw James. "3D curvature attributes: a new approach for seismic interpretation". In: *First Break* 26 (Apr. 2008).

[12] David C. Lay. *Linear algebra and its applications*. 3rd ed. Pearson Education, 2006.

[13] Thomas Lewiner et al. "Arc-length Based Curvature Estimator". In: *Sibgrapi 2004 (XVII Brazilian Symposium on Computer Graphics and Image Processing)*. IEEE, 2004, pp. 250–257. URL: http://thomas.lewiner.org/pdfs/curvature_sibgrapi.pdf.

[14] Tom Lindstrøm. *Kalkulus*. 3rd ed. Universitetsforlaget AS, 2006, p. 377.

[15] Richard J. Lisle. "Detection of Zones of Abnormal Strains in Structures Using Gaussian Curvature Analysis". In: *AAPG Bulletin* 78.12 (1994), pp. 1811–1819.

[16] John Mathewson and Dave Hale. "Detection of channels in seismic images using the steerable pyramid". In: *SEG Technical Program Expanded Abstracts*. Vol. 27. 1. 2008, p. 859.

[17] Knut Mørken. "Numerical Algorithms and Digital Representation". URL: www.uio.no/studier/emner/matnat/math/MAT-INF1100/h11/kompendiet/komp.html (visited on 05/15/2012).

[18] Shyamosree Pal and Partha Bhowmick. "Estimation of Discrete Curvature Based on Chain-Code Pairing and Digital Straightness". In: IEEE, 2009, pp. 1097–1100.

[19] Trygve Randen et al. *Three-dimensional Texture Attributes for Seismic Data Analysis*. 2000.

[20] Andy Roberts. "Curvature attributes and their application to 3D interpreted horizons". In: *First Break* 19.2 (2001), pp. 85–100.

[21] Edward J. Tarbuck and Frederick K. Lutgens. *Earth: an introduction to physical geology*. 10th ed. Pearson Prentics Hall, 2011.

[22] Marcel Worring and Arnold W. M. Smeulders. "Digital Curvature Estimation". In: *Computer Vision, Graphics, and Image Processing: Image Understanding* 58.3 (1993), pp. 366–382.

Online sources

[23] *Curvature at Encyclopedia of Mathematics*. Springer and The European Mathematical Society. URL: http://www.encyclopediaofmath.org/index.php/Curvature (visited on 02/14/2012).

[24] *Curvature at Wolfram MathWorld*. Wolfram Research. URL: http://mathworld.wolfram.com/Curvature.html (visited on 04/05/2012).

[25] *GeoCLASS*. UniGEO. URL: http://www.unigeo.no (visited on 05/01/2011).

[26] *Schlumberger Oilfield Glossary*. Schlumberger. URL: http://www.glossary.oilfield.slb.com (visited on 02/06/2012).

[27] *Veritas Caspian*. URL: www.veritas-caspian.kz (visited on 10/09/2012).