CWP-643
May 2010

Seismic image flattening as a linear inverse problem

Derek Parks

— Master of Science Thesis —
Mathematical & Computer Sciences

Defended January 19, 2010

Committee Chair: Dr. Dinesh Mehta
Advisor: Dr. Dave Hale
Co-Advisor: Dr. Andrzej Szymczak

Committee member:
Dr. Anthony Petrella

Center for Wave Phenomena
Colorado School of Mines
Golden, Colorado 80401
(1) 303 273-3557

Abstract

Seismic image flattening is a common task in geophysical interpretation and is typically used to identify stratigraphic features within a seismic image. 3D seismic images are normally interpreted by viewing 2D slices of the image on a desktop computer. Unfortunately, It can be difficult for an interpreter to identify features of the subsurface by viewing horizontal slices or axis-aligned probes of an image because geologic features are rarely aligned with the axes of a seismic survey.

Seismic image flattening attempts to reverse the effects of geologic processes by removing all of the geologic structure present in the image and thereby transforming the image into layers as they were deposited in geologic time; i.e., flattening transforms a seismic image to make the structural features in the image flat. Stratigraphic features such as channels are easier to recognize after flattening because an interpreter is able to view an entire geologic interface at once. Typically, a flattened seismic image is created by shifting samples in the original image up or down. This means that parts of the original image are stretched in some areas and squeezed in others to flatten the features in the image.

Traditionally, flattening is performed by having an interpreter manually pick events in a seismic image. This picking process requires specialized training and can be time-consuming. I present a method for seismic image flattening that is completely automatic. This new flattening algorithm uses the structure tensor to estimate the local dip of every sample in a seismic image and a linear least squares inversion to solve for the shifts that will flatten the image.

Table of Contents

Abstract i

Acknowledgments v

Chapter 1 Introduction 1

1.1 Manual Horizon Flattening Versus Full-Volume Flattening 2
1.2 Slope-Based Full-Volume Flattening 5

Chapter 2 Nonlinear equations for flattening shifts 7

2.1 Mapping From an Input Image to a Flattened Image 7
2.2 Vertical Shifts 9
2.3 Computing Shifts from Slopes 10
2.4 Nonlinearity 14

Chapter 3 Linear equations for flattening shifts 17

3.1 Mapping in Terms of τ with Inverse Interpolation 17
3.2 A Linear System of PDEs 18
3.3 Geometric Derivation of the Linear Equations 20
3.4 Change of Variables 24
3.5 Results and Conclusions 29

References 35

Acknowledgments

First, I would like to thank my parents Gary Parks, Phyllis Schmit, and Mike Schmit for supporting me in all of my endeavors. Next, I would like to thank and acknowledge my advisors Andrzej Szymczak and Dave Hale. This work would not have been possible without their insight and guidance. I thank Bill Harlan for sharing his expertise in inversion problems with me. I thank all of the Center for Wave Phenomena students and faculty for providing a great environment in which to work and learn. I acknowledge and thank Landmark Graphics Corporation for funding this research. I would like to thank Diane Witters and John Jackson for their help proof reading. Finally, I thank the Rocky Mountain Oilfield Testing Center and the United States Department of Energy for the use of the Teapot Dome seismic image.

Chapter 1

Introduction

As the worldwide demand for hydrocarbons grows, seismic imaging has been a valuable tool for detecting and mapping features of the subsurface. Rising demand has led to prospecting for hydrocarbons in remote locations, such as in deep water offshore where the cost of a failed prospect is extremely high. Geophysical interpreters must better map and visualize subsurface structures so that such failures can be avoided.

Many image processing and seismic attribute algorithms have been developed to help interpreters analyze the subsurface. One such technique is the process of seismic image flattening. Various forms of seismic image flattening have been used in practice to aid in the interpretation of seismic images (e.g., Stark (1996); Zeng *et al.* (1998); Gao (2009); Lomask *et al.* (2006)).

It can be difficult for an interpreter to identify subsurface features by viewing horizontal time (or depth) slices or any axis-aligned surfaces of an image because geologic features are rarely aligned with the axes of a seismic survey. Thus, a simple constant-time slice may intersect many different geologic layers. Therefore, to study a single geologic layer an interpreter may have to view several different slices and attempt to remember relevant pieces from each of them. Seismic image flattening attempts to reverse the effects of geologic processes by removing all of the geologic structure present in the image, i.e., flattening transforms a seismic image such that the axes used to display the data are more geologically relevant.

Once an image has been flattened, all of the geologic interfaces in the image are transformed to be flat (planar and horizontal). Therefore, after flattening, a horizontal time slice of a flattened image corresponds to a single geologic interface. Flattened events can be seen as an image of how the layers of the earth were deposited in geologic time. While flattening is not a replacement for an advanced stratigraphic interpretation (such as palinspastic reconstruction), flattening can be performed automatically and does not require advanced

knowledge of sequence stratigraphy. After flattening, even a novice interpreter can quickly recognize stratigraphic features (e.g. buried channels) which may have been obscured by geologic structures present in the original image.

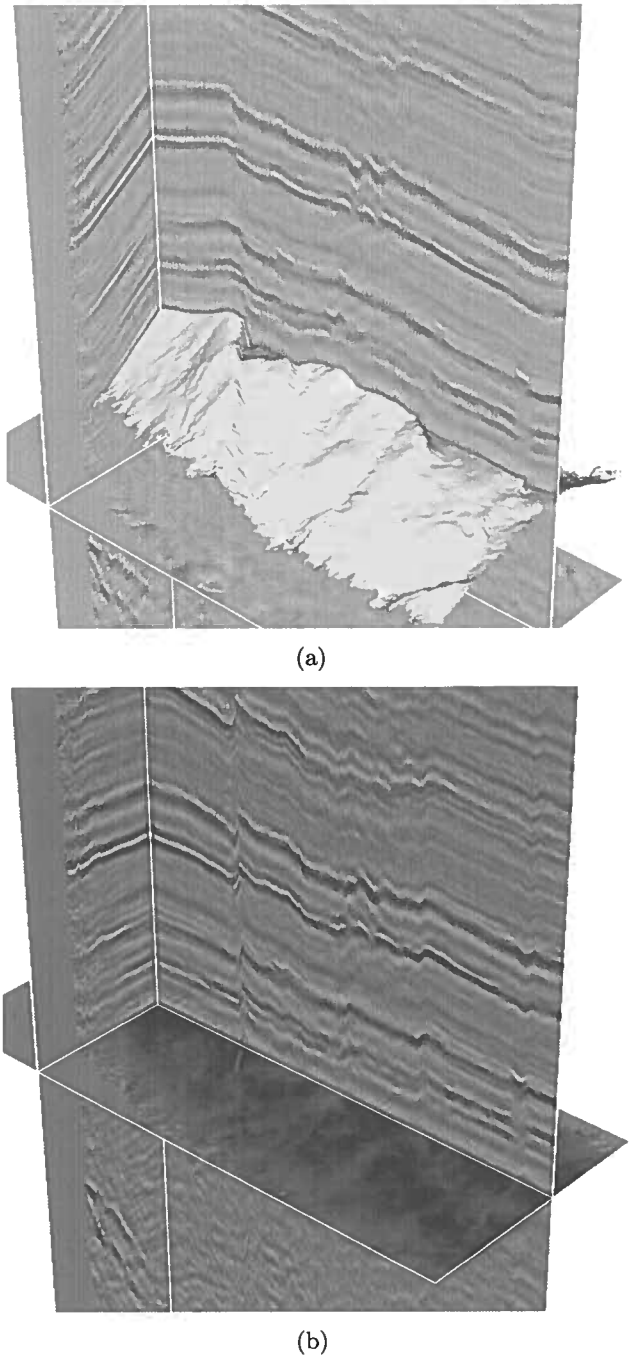
Seismic data are flattened by removing the slope (or dip) from events within a seismic image. Typically, a flattened seismic image is created by shifting samples in the original image up or down such that coherent events in the image are flattened. Therefore, each trace (a column of samples) in the original image is stretched in some areas and squeezed in others. Hence, all flattening methods can be reduced to finding how the input image must be warped (or mapped) to flatten events.

1.1 Manual Horizon Flattening Versus Full-Volume Flattening

In many commercial seismic interpretation applications, flattening is performed by manually interpreting horizons (geologic interfaces). Horizon interpretation requires a user to map an entire event in a 3D seismic image. The yellow surface in Figure 1.1a is an example of an interpreted horizon. The aforementioned horizon mapping is done by visually identifying and picking (digitizing) an event throughout the entire seismic image. A user typically digitizes an event by clicking points on a 2D vertical slice of the seismic image to create a polyline that follows the event. Then, a 3D surface of the event can be created by interpolating between the user's picks. Figure 1.2 shows a simplified example of this picking process. An interpreted surface can be used to calculate how the samples of the input image must be shifted to flatten the interpreted horizon. Figure 1.1b shows the seismic image from Figure 1.1a after flattening based on the interpreted yellow horizon.

Manual horizon interpretation may be facilitated by using an automatic horizon tracking algorithm. Unfortunately, automated horizon tracking is rarely completely accurate when events are of low amplitude and, in many cases, must be guided by a user. Thus, horizon tracking, whether done manually or using automatic picking algorithms, can be a time-consuming process.

Single horizon flattening can easily be extended to flatten more than one horizon, but each new horizon to be flattened must also be picked or tracked. Labrunye *et al.* (2009) present a method for performing horizon-based flattening which can flatten every layer in an image, even across geologic discontinuities; but one must interpret many horizons and, possibly, faults to get an accurate flattening. Horizon-interpretation-based flattening has two distinct disadvantages. First, in order to flatten horizons, one must perform manual



(a)

(b)

Figure 1.1: An example of single horizon flattening seen in a typical seismic interpretation workflow. The horizon in (a) has been interpreted by manually picking points within the seismic image. Then, the horizon has been used to flatten the seismic image seen in (b). Notice that only one horizon (the one interpreted in (a)) has been flattened.

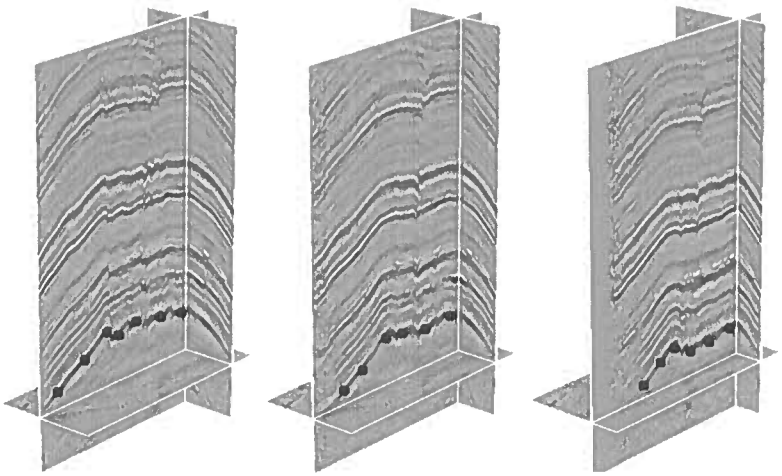


Figure 1.2: An example of manual horizon picking on three different slices of a seismic image. An interpreter has identified the bottom horizon by clicking at the locations of the red dots in (a), (b), and (c). The surface shown in Figure 1.1 was created by preforming this picking process at a much more dense interval of slices.

interpretation. Second, horizons that contain interesting stratigraphic features must be known beforehand, so that they can be interpreted and flattened.

Often, it is preferable to perform full-volume flattening (Stark, 2004) which flattens all of the events within a seismic image. Full-volume flattening leads to a simple stratigraphic reconnaissance procedure because an interpreter does not need to know beforehand which geologic layers contain useful information. In a full-volume flattened image, one can simply pan through horizontal-time slices (which are now flattened geologic layers) and identify layers which contain interesting stratigraphic features.

In recent years, several methods for performing automatic full-volume flattening have been developed. The main challenge of automatic flattening is how one should track the geologic interfaces of the input image. The work presented in Stark (2003, 2005) applies the method of instantaneous phase unwrapping in 3D to track geologic interfaces in the image. Pauget *et al.* (2009) correlate neighboring traces to find probabilistic links that follow events between traces and then track global events by minimizing a cost function of the probabilistic links. De Groot *et al.* (2006) first calculate the local slopes of these geologic interfaces within a seismic image. Then, they use the slopes to track geologic interfaces from

a starting seed point. In the following section, I will discuss a flattening method that uses the local slopes of geologic interfaces to perform a global inversion and flatten the image.

1.2 Slope-Based Full-Volume Flattening

In this thesis, I will build upon the full-volume flattening algorithm presented in Lomask (2006); Lomask *et al.* (2006); Lomask & Guitton (2007). Both Lomask *et al.*'s (2006) method and the new method presented here compute a flattened image directly from an input image and do not require a user to perform any manual interpretation.

Interestingly, the Lomask *et al.* (2006) flattening method starts by computing an estimate of the local slope at every sample in the seismic image, then flattens the input image using only these local slope estimates. The local slopes give an estimate of the predominate direction of events in the seismic image and can be computed using a variety of methods, such as plane-wave destruction filters (Claerbout, 1992; Fomel, 2002) or structure tensors (van Vliet & Verbeek, 1995; Fehmers & Höcker, 2003). An extended discussion of slope calculation methods applied to seismic images can be found in Chopra & Marfurt (2007). Figure 1.3 shows an example of slopes estimated from structure tensors for inline and crossline slices of a 3D seismic image.

The local slopes, calculated from the input image, are then used as the data in an inversion problem which attempts to find shifts for the events in the input image that will make the slopes go to zero. In a typical inversion problem, one seeks to find a model $\mathbf{m}$ given data $\mathbf{d}$ such that

$$\mathbf{d} \approx \mathbf{F}(\mathbf{m}), \tag{1.1}$$

where $\mathbf{F}$ is a forward operator that relates the model $\mathbf{m}$ and data $\mathbf{d}$. In this case, the data are the local slopes and the model is the amount by which each sample of the input image must be shifted to make the slopes zero. I will discuss the nonlinear forward operator presented in Lomask *et al.* (2006) in the following chapter. In the third chapter, I present a new linear forward operator and a method for reparameterizing this inverse problem. Both the new linear forward operator and the reparameterization technique allow one to compute a solution to the flattening inverse problem in less time than the previously presented method.

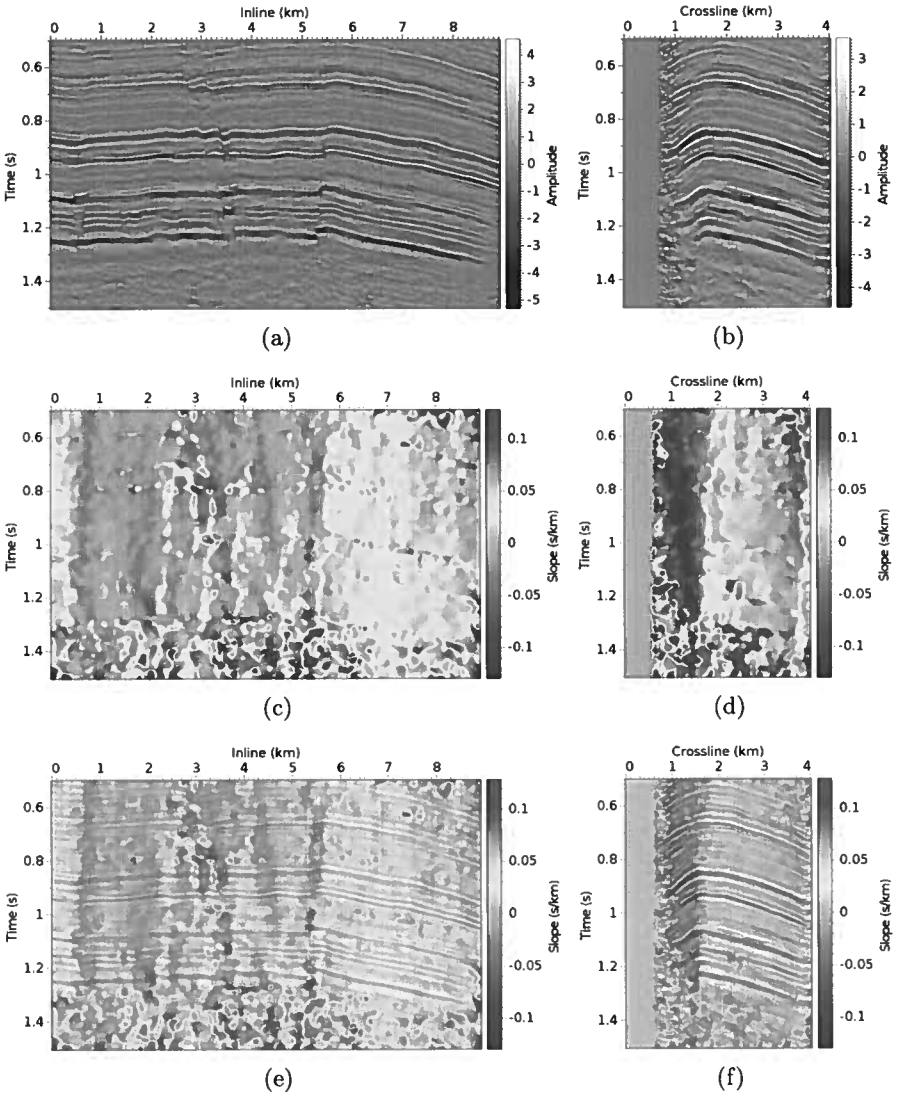


Figure 1.3: Inline and crossline slices of a 3D seismic image are shown in (a) and (b), respectively. The computed local slopes for the slices shown in (a) and (b) are shown in (c) and (d), respectively. The hotter colors indicate that events are sloping down to the right, while cooler colors indicate that the event are sloping down to the left. Corresponding image slices and slopes plotted on top of each other are shown in (e) and (f).

Chapter 2

Nonlinear equations for flattening shifts

In this chapter, I present full-volume flattening as the solution of a nonlinear inverse problem. In a typical inversion problem, one seeks to find a model that best explains some data. In this case, the data are slopes derived from an input image, and I solve for a model, shifts that describe how to flatten the image. Inversion is routinely used in other geophysical problems (such as seismic tomography and impedance inversion), and can also be used for the interpretation of seismic data. The method for flattening that I present in this chapter is based on the work of Lomask *et al.* (2006). An inversion-based method has several distinct advantages over other methods. It is automatic; it does not require labor-intensive horizon picking. Also, this method ensures that all of the geologic interfaces are flattened and can give a reasonable flattening even when parts of the image are noisy or of low quality.

In the following section, I introduce a function that specifically defines how the flattened image is found. Next, I explain the shifts that are used to flatten the image. Then, I show the forward operator that relates the model to the input image, and finally, I conclude by discussing the nonlinearity of the inverse problem.

2.1 Mapping From an Input Image to a Flattened Image

A flattened image is made up entirely of data derived from an original input image; thus, a flattened image is an input image rearranged such that all of the events are flat. This flattening method simplifies the problem even further by assuming that the samples of the input image only need to be moved vertically up or down to create flattened events. Figure 2.1 is a cartoon, in which curves represent events in a seismic image, that illustrates how this image rearrangement is performed. Notice that some samples must be moved vertically away from each other (stretching), and other samples must be moved closer together

(squeezing).

The flattening process is analogous to finding a function $t(x, y, \tau)$ that stretches and squeezes the events in the input image. Specifically, given an input image $f(x, y, t)$ and a mapping function $t(x, y, \tau)$, the flattened output image is computed by

$$g(x, y, \tau) = f(x, y, t(x, y, \tau)), \tag{2.1}$$

where x and y denote the inline and crossline directions. Notice that the vertical axes are different in the input and output images. The input image is a function of time t. (For consistency, I am assuming the input image is a function of time but flattening works equally well for input images that have been converted or migrated to depth.) Then, I define the flattened image as a function of a new variable τ that spans geologic time. This geologic time τ does not correspond to a specific number of millions of years ago but instead, each individual value of τ corresponds to an individual hypothetical geologic time. The samples that make up a horizontal slice in the flattened image correspond to a similar geologic event.

Notice that given the function $t(x, y, \tau)$, one can find a time surface for every value of τ, and a given τ defines a t value for every point (x, y). Thus, a single-value of τ corresponds to a single geologic time in both the input and flattened images. The flattening process finds the function $t(x, y, \tau)$ that maps between t and τ and then uses this mapping to interpolate the flattened image from the input image using equation 2.1.

There are several inherent assumptions in equation 2.1 that should be considered. The first, as stated above, is that samples only need to be moved vertically to flatten an image. The second is that events in the input image can be approximated by surfaces of constant τ; i.e., geologic interfaces in the input image are single-valued 3D surfaces.

Neither of these assumptions is completely valid for all seismic images. Layers of the subsurface can be faulted and moved laterally by geologic processes, thus breaking the first assumption. In this case, my automatic flattening will flatten the layers on both sides of the fault, but the layers on either side of the fault may not be connected in the output image. Also, there are geologic phenomena that cannot be approximated by single-valued surfaces; e.g., salt bodies.

As stated previously, this automatic flattening method may not produce a perfect palinspastic reconstruction but despite these assumptions, automatic flattening is useful in many cases because important stratigraphic features (such as channels) that one would like to find in a flattened image do meet these assumptions.

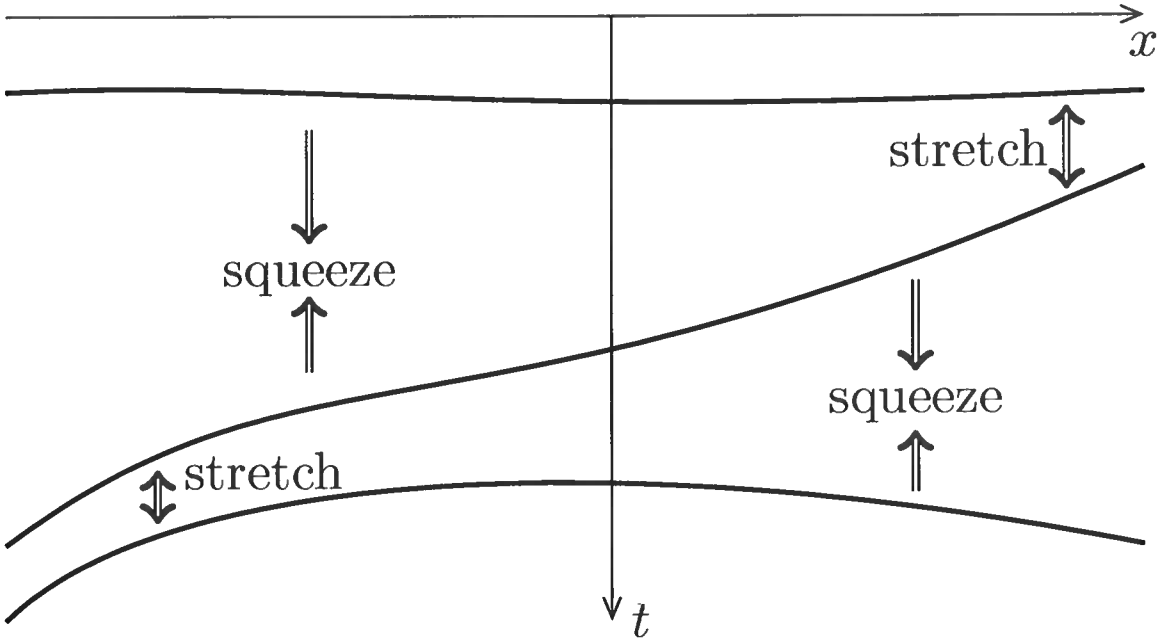


Figure 2.1: Flattening is performed by shifting events in the image; thus, in this cartooned seismic image, some areas are stretched and others are squeezed to flatten the image.

2.2 Vertical Shifts

The mapping function $t(x, y, \tau)$ can be related to the image by considering how events must be shifted to make them flat. Let the function $s(x, y, \tau)$ denote the vertical time shifts that must be applied to each sample in the input image to flatten. Figure 2.2a shows a cartoon of a 2D (or a slice of a 3D) input image. For simplicity the image only has one event. This event corresponds to a geologic interface which has been labeled τ_0. Thus, the mapping function $t(x, \tau_0)$ gives the time t of the event as a function of x. The shifts required to flatten event τ_0 are shown as vertical vectors in Figure 2.2a. Notice that the shift varies with the shape of the event and is positive on the right and negative on the left.

The same event is shown after flattening in Figure 2.2b. The event in the flattened image is made by interpolating samples from the input image at every time τ_0, and in general, a flattened image is created by interpolating samples from times given by the mapping function $t(x, \tau)$. Using this same logic, the 3D mapping function is:

$$t(x, y, \tau) = \tau - s(x, y, \tau). \tag{2.2}$$

The above equation implies that if the shift function $s(x, y, \tau)$ is known, then the mapping function $t(x, y, \tau)$ is also known. The shift function is the model in the flattening inverse problem. In the next section, I show how these shifts are related to the input image.

2.3 Computing Shifts from Slopes

A transform between the shift function $s(x, y, \tau)$ and the local slopes measured for an input image can be found by taking derivatives of both sides of equation 2.1 with respect to both x and y. The derivative with respect to x of both sides of equation 2.1 is

$$\frac{\partial g}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial x} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial x} + \frac{\partial f}{\partial t}\frac{\partial t}{\partial x} \tag{2.3}$$

$$= \frac{\partial f}{\partial x} + \frac{\partial f}{\partial t}\frac{\partial t}{\partial x}. \tag{2.4}$$

If the flattened image $g(x, y, \tau)$ only contains flattened geologic interfaces as in Figure 2.2b, $\partial g/\partial x$ is zero. Thus, equation 2.4 becomes

$$0 = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial t}\frac{\partial t}{\partial x} \tag{2.5}$$

$$-\frac{\frac{\partial f}{\partial x}}{\frac{\partial f}{\partial t}} = \frac{\partial t}{\partial x} \tag{2.6}$$

$$-\frac{\frac{\partial f}{\partial x}}{\frac{\partial f}{\partial t}} = -\frac{\partial s}{\partial x}. \tag{2.7}$$

The left-hand side of the above equation can be computed from an input image and is the slope of events in the image.

Unfortunately, computing the slopes from the partial derivatives $\partial t/\partial x$ and $\partial f/\partial t$ may result in numerical problems. These errors are due to several causes. First, seismic images are often noisy; thus, derivatives of these images will also contain noise. Next, notice that the shifts $s(x, y, \tau)$ can be found by integrating both sides of equation 2.7. Therefore, the shifts are a summation of the slopes, and small errors in the derivatives may quickly accumulate. Finally, when $\partial f/\partial t$ is zero, $-(\partial f/\partial x)/(\partial f/\partial t)$ is undefined. Hence, it is preferable to use a more robust method for computing slopes and to smooth input slopes before computing shifts.

The structure tensor (van Vliet & Verbeek, 1995) provides a simple and robust method for computing smoothed slopes. Using the structure tensor one can compute the direction

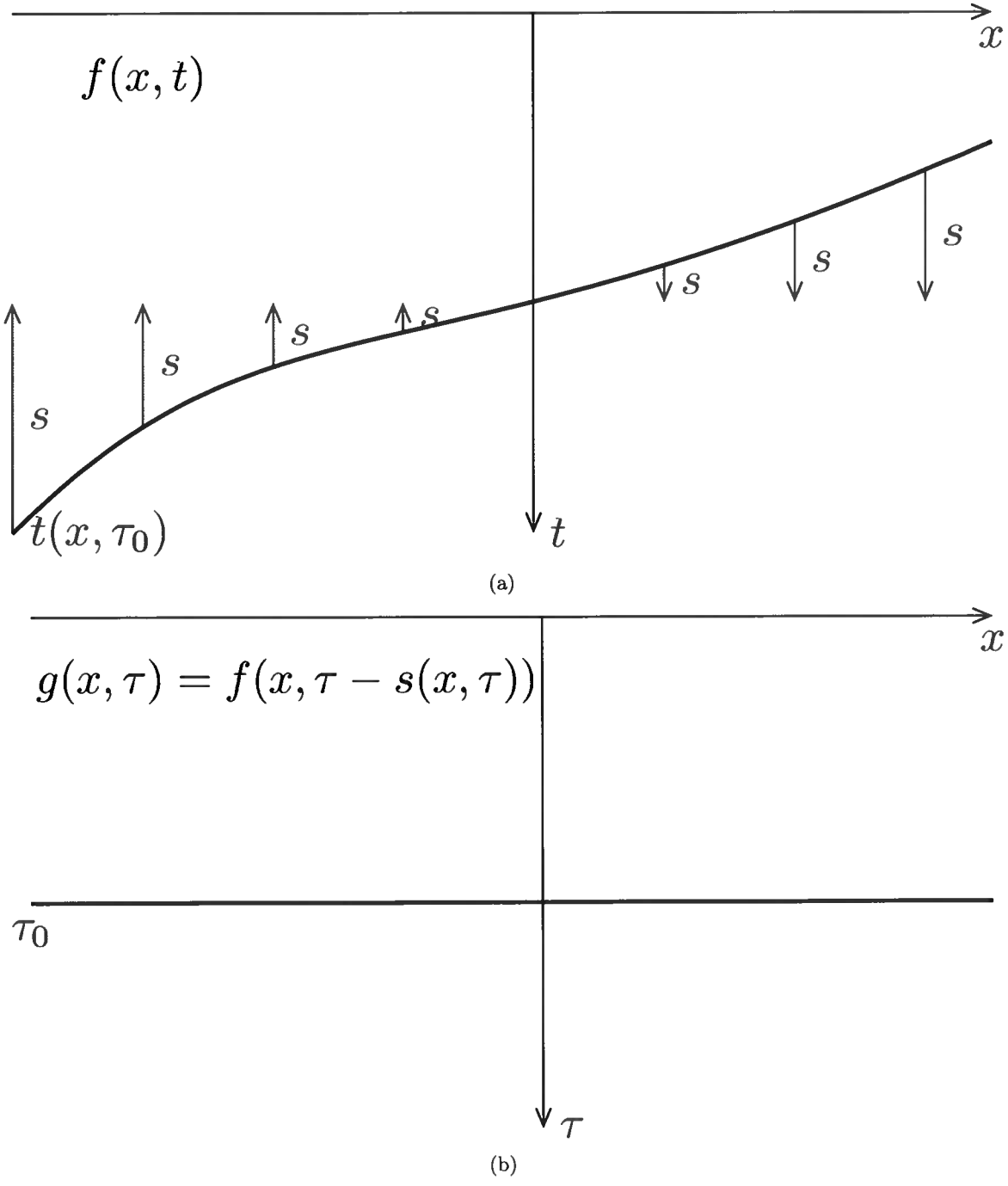


Figure 2.2: A 2D cartoon of a seismic event before and after flattening is shown in (a) and (b), respectively. The blue arrows in (a) show how the event must be shifted to flatten the geologic interface. The length and direction of these arrows is the value of the shift function $s(x, \tau_0)$. From (a) it can be seen that the flattened event in (b) is given by $g(x, \tau_0) = f(x, \tau_0 - s(x, \tau_0))$.

of greatest change in the image over a local window for every sample in the image. The direction of greatest change in the image is normal to the events in the image. Figures 2.3a and 2.3b, show, for inline and crossline slices of a 3D seismic image, a subset of the local normal vectors computed using structure tensors. Figure 2.3c shows the relationship between these local normal vectors and the slope of a seismic event. Finally, Figures 2.3d and 2.3e show the computed slopes plotted on top of their corresponding seismic slices.

To simplify notation, the left-hand side of equation 2.7 is defined to be a function:

$$p(x,y,t) \equiv \frac{\Delta t}{\Delta x} = -\frac{\frac{\partial f}{\partial x}}{\frac{\partial f}{\partial t}} = -\frac{n_x(x,y,t)}{n_t(x,y,t)}, \tag{2.8}$$

so that equation 2.7 becomes

$$p(x,y,t) = -\frac{\partial s}{\partial x}(x,y,\tau). \tag{2.9}$$

A similar derivation can be made for the y derivative of equation 2.1. I define the slope in the y direction to be

$$q(x,y,t) \equiv \frac{\Delta t}{\Delta y} = -\frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial t}} = -\frac{n_y(x,y,t)}{n_t(x,y,t)} \tag{2.10}$$

and consequently, the relation for the shift in the y direction is

$$q(x,y,t) = -\frac{\partial s}{\partial y}(x,y,\tau). \tag{2.11}$$

Notice that there are two relations for the shift function $s(x,y,\tau)$, equation 2.9 for the inline x direction and equation 2.11 for the crossline y direction. Lomask *et al.* (2006) suggest that one should attempt to honor both equations by using a least-squares solution fit to both equations. They also suggest that one should include a third equation to enforce smoothness in the shift function to "ensure a monotonic and continuous result". The third line in equation 2.12 sets $\partial s/\partial \tau$ to zero and includes a tuning parameter ϵ which presents a tradeoff between smoothness and the need to honor the input slopes. A larger ϵ puts more emphasis on having little vertical change in the shift function, while a smaller ϵ allows the shift function to vary more vertically. The value of ϵ is dependent on the amount of noise in the input image.

Using the notation introduced above, the following equations are the partial differential

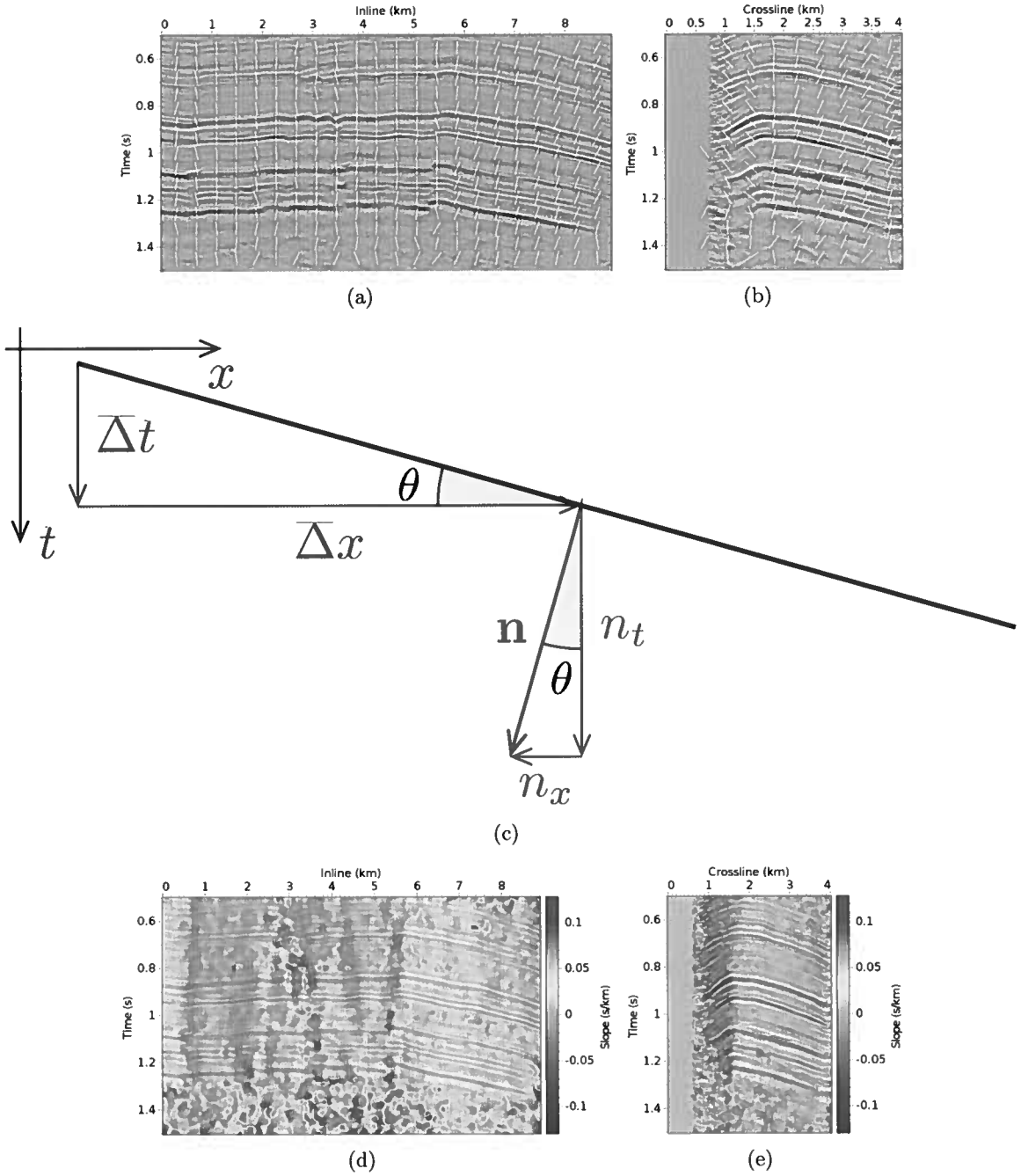


Figure 2.3: (a) and (b) show inline and crossline slices of a seismic image where a subset of the local normal vectors have been plotted on top of the image. (c) is an idealized cartoon of a local window around one of the vectors plotted in (a) and (b). The local slopes computed using the normal vectors from (a) and (b) are plotted in color on top of the seismic data in (d) and (e).

equations presented by Lomask *et al.* (2006):

$$\begin{bmatrix} -\frac{\partial s}{\partial x}(x,y,\tau) \\ -\frac{\partial s}{\partial y}(x,y,\tau) \\ \epsilon\frac{\partial s}{\partial \tau}(x,y,\tau) \end{bmatrix} \approx \begin{bmatrix} p(x,y,t) \\ q(x,y,t) \\ 0 \end{bmatrix}. \tag{2.12}$$

2.4 Nonlinearity

Notice that left-hand side of equation 2.12 is a function of τ, while the right-hand side is a function of t. The local slopes $p(x,y,t)$ and $q(x,y,t)$ are functions of t because they are estimated from the input image. However, the shift function (that is being solved for) is a function of τ and because the τ variable is based on the geology present in the image there is no constraint that it must be linear. The need to have a shift function in terms of τ can be seen in equation 2.1 and is illustrated in Figure 2.2. The output image (Figure 2.2b) is created by interpolating samples from the input image (Figure 2.2a) at times $t(x,\tau_0)$; however, in order to know where samples are located in the input image, one must know the shifts $s(x,\tau_0)$.

Lomask *et al.* (2006) explain that a least-squares solution to the nonlinear partial differential equations in equation 2.12 can be found by linearizing them approximately and then, iteratively solving for the shifts. In iteration k, the shift function from the previous iteration $s^{\{k-1\}}(x,y,\tau)$ is used to relate the measured slopes to the shift function and solve for a new shift function $s^{\{k\}}(x,y,\tau)$. For the first iteration, it is assumed that t and τ are equal, so that

$$s^{\{0\}}(x,y,\tau) = 0. \tag{2.13}$$

Then, for each iteration k, least squares fitting is used to solve the following equations

$$\begin{bmatrix} -\frac{\partial s^{\{k\}}}{\partial x}(x,y,\tau) \\ -\frac{\partial s^{\{k\}}}{\partial y}(x,y,\tau) \\ \epsilon\frac{\partial s^{\{k+1\}}}{\partial \tau}(x,y,\tau) \end{bmatrix} \approx \begin{bmatrix} p(x,y,\tau - s^{\{k-1\}}(x,y,\tau)) \\ q(x,y,\tau - s^{\{k-1\}}(x,y,\tau)) \\ 0 \end{bmatrix}. \tag{2.14}$$

A solution to the above relation can be found using an iterative solver such as the conjugate gradient method. Therefore, there are two different sets of iterations preformed when solving for a shift function $s(x,y,\tau)$, there are successive iterations of k which are repeated until the difference between $s^{\{k-1\}}(x,y,\tau)$ and $s^{\{k\}}(x,y,\tau)$ is small, and there are iterations within the iterative solver used to compute each new shift function $s^{\{k\}}(x,y,\tau)$. Figure 2.4 shows a slice from a 3D seismic image where the shifts $s^{\{k\}}(x,y,\tau)$ which were solved for at each iteration k have been used to compute a flattened image.

When solving for one particular version of the shift function $s^{\{k\}}(x,y,\tau)$, it is difficult to know how many iterations of the conjugate gradient method to perform. This is because at each iteration k a linear approximation of the true problem is being solved, and thus, one may waste computations by taking the iterative solver to full convergence. Unfortunately, it is challenging to know beforehand how long to iterate on the incorrect linearized problem. This uncertainty motivates the need for a linear relationship between local slopes and the shift function. Such a linear system can be solved once with the conjugate gradient method taken to full convergence.

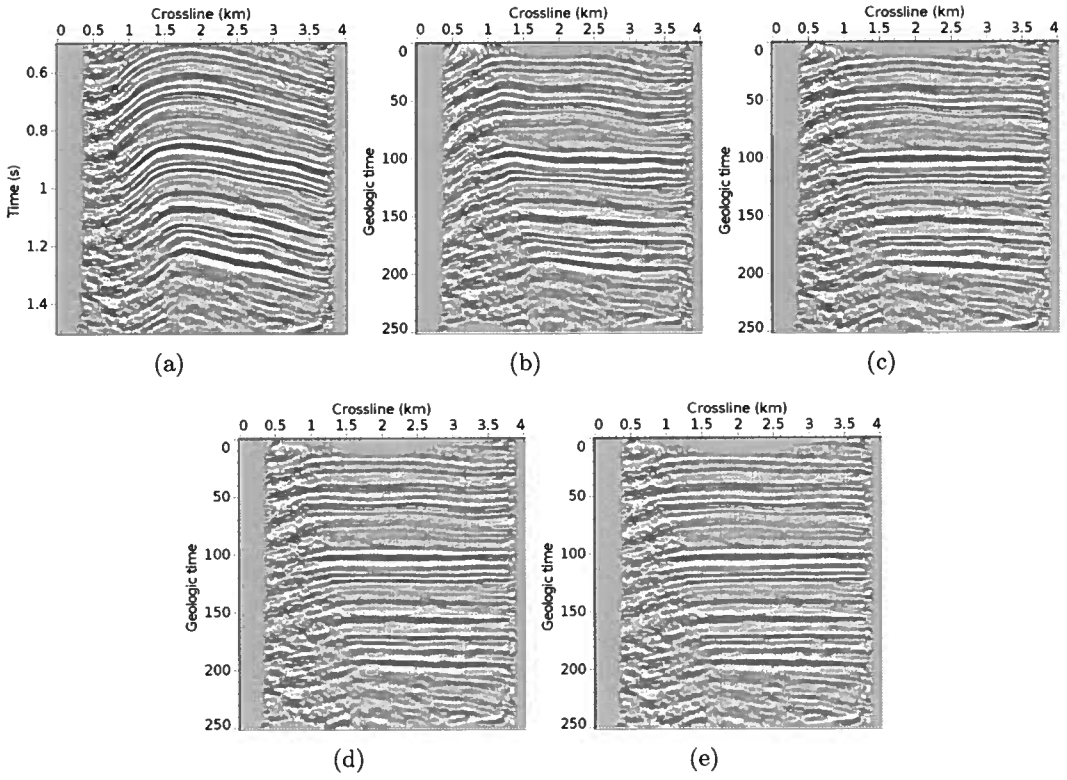


Figure 2.4: A crossline slice from a 3D seismic image shown after each iteration of flattening. The original input image is shown in (a). Then, each subsequent image shows the same slice after applying the shift function found at each of four iterations.

Chapter 3

Linear equations for flattening shifts

In this chapter I present a new method for solving for the shift function used to flatten a seismic image. In the previous chapter, the slopes of events in an input image were calculated as functions of time t, but to flatten the image, a shift function of τ was required. This difference of functions in terms of t and τ led to a set of nonlinear partial differential equations (PDEs) for shifts. I propose that instead of directly calculating a shift function in terms of τ, one should use a two-step process to find the flattening shifts:

1. Solve a set of linear PDEs to a find shift function $\sigma(x, y, t)$ of t.
2. Use inverse interpolation to find a shift function $s(x, y, \tau)$ of τ that will flatten the image.

In the following section, I will explain this new inverse interpolation step in further detail. Then, I will introduce a new set of PDEs that can be used to calculate the new shift function $s(x, y, \tau)$. Next, I will introduce a change of variables that yields faster convergence when solving for the shift function in practice. Finally, I will show examples of this flattening method applied to a 3D seismic image and compare runtime results with the nonlinear method.

3.1 Mapping in Terms of τ with Inverse Interpolation

In the previous chapter, the function $t(x, y, \tau)$ that maps the input image to the flattened image was discussed:

$$g(x, y, \tau) = f(x, y, t(x, y, \tau)), \tag{3.1}$$

and it was shown that this mapping function is directly related to the shifts by

$$t(x, y, \tau) = \tau - s(x, y, \tau), \tag{3.2}$$

where $s(x, y, \tau)$ is the shift function. The key to finding a set of linear PDEs for shifts is to realize that it is simpler to find a shift function in terms of time t. Then, the shift function of t can be used to find a shift function in terms of τ. Instead of inverting for a shift function $s(x, y, \tau)$ and calculating a mapping function $t(x, y, \tau)$, one finds a shift function $\sigma(x, y, t)$ and a mapping function $\tau(x, y, t)$:

$$\tau(x, y, t) = t + \sigma(x, y, t). \tag{3.3}$$

Given a mapping function in terms of t, one can easily use inverse interpolation to find the mapping in terms of τ if it is assumed that t monotonically increases as τ increases. Using inverse interpolation, one can go from τ as a function of t to t as a function of τ:

$$\tau(x, y, t) \rightarrow t(x, y, \tau). \tag{3.4}$$

Note that the function $t(x, y, \tau)$ was used in the previous chapter, and each value of τ corresponds to a geologic time. Therefore, the assumption that t monotonically increases is equivalent to the assumption that there is a top to bottom ordering of geologic time, i.e., the layers of the earth were created in a chronological order.

3.2 A Linear System of PDEs

Next, I use a derivation similar to the previous chapter to introduce a new linear system of equations that relates slopes in the input image to shifts as a function of time t. Start by rewriting equation 2.1 using the new mapping function $\tau(x, y, t)$:

$$g(x, y, \tau(x, y, t)) = f(x, y, t). \tag{3.5}$$

Then, as before, take the derivative of the above equation with respect to x:

$$\frac{\partial g}{\partial x}\frac{\partial x}{\partial x} + \frac{\partial g}{\partial y}\frac{\partial y}{\partial x} + \frac{\partial g}{\partial \tau}\frac{\partial \tau}{\partial x} = \frac{\partial f}{\partial x} \tag{3.6}$$

and simplify:

$$\frac{\partial g}{\partial x} + \frac{\partial g}{\partial \tau}\frac{\partial \tau}{\partial x} = \frac{\partial f}{\partial x}. \tag{3.7}$$

Note that, again, the derivative of the output image with respect to x is zero because the output image only contains flat layers. Thus, the above equation simplifies to

$$\frac{\partial g}{\partial \tau}\frac{\partial \tau}{\partial x} = \frac{\partial f}{\partial x}. \tag{3.8}$$

Next, expand the above equation by computing the derivative of both sides of equation 3.5 with respect to t:

$$\frac{\partial g}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial g}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial g}{\partial \tau}\frac{\partial \tau}{\partial t} = \frac{\partial f}{\partial t}, \tag{3.9}$$

which simplifies to

$$\frac{\partial g}{\partial \tau}\frac{\partial \tau}{\partial t} = \frac{\partial f}{\partial t} \tag{3.10}$$

or

$$\frac{\partial g}{\partial \tau} = \frac{\frac{\partial f}{\partial t}}{\frac{\partial \tau}{\partial t}}. \tag{3.11}$$

Using equation 3.11, equation 3.8 becomes

$$\frac{\frac{\partial f}{\partial t}}{\frac{\partial \tau}{\partial t}}\frac{\partial \tau}{\partial x} = \frac{\partial f}{\partial x}. \tag{3.12}$$

Then, using the derivatives of equation 3.3 with respect to x and t equation 3.12 yields

$$\frac{\frac{\partial f}{\partial t}}{1 + \frac{\partial \sigma}{\partial t}}\frac{\partial \sigma}{\partial x} = \frac{\partial f}{\partial x} \tag{3.13}$$

Finally, using the definition in equation 2.8 for the slopes $p(x, y, t)$, the above equation 3.13 becomes

$$-\frac{\partial \sigma(x, y, t)}{\partial x} - p(x, y, t)\frac{\partial \sigma(x, y, t)}{\partial t} = p(x, y, t). \tag{3.14}$$

A similar derivation can be performed for the slopes $q(x, y, t)$ in the y direction:

$$-\frac{\partial\sigma(x,y,t)}{\partial y} - q(x,y,t)\frac{\partial\sigma(x,y,t)}{\partial t} = q(x,y,t). \tag{3.15}$$

Note that both the slopes $p(x, y, t)$ and $q(x, y, t)$ and the shifts $\sigma(x, y, t)$ are functions of time t. Equations 3.14 and 3.15 do not include the unknown function $\tau(x, y, t)$. They are linear PDEs.

Table 3.1 shows the above equations and the equations from the previous chapter side by side. Note that the right-hand sides are the same, but that the left-hand sides are very different.

Table 3.1: Shift PDEs comparison

Nonlinear	$\begin{bmatrix} -\frac{\partial s}{\partial x}(x,y,\tau) \\ -\frac{\partial s}{\partial y}(x,y,\tau) \end{bmatrix} \approx \begin{bmatrix} p(x,y,t) \\ q(x,y,t) \end{bmatrix}$
Linear	$\begin{bmatrix} -\frac{\partial\sigma}{\partial x}(x,y,t) - p(x,y,t)\frac{\partial\sigma}{\partial t}(x,y,t) \\ -\frac{\partial\sigma}{\partial y}(x,y,t) - q(x,y,t)\frac{\partial\sigma}{\partial t}(x,y,t) \end{bmatrix} \approx \begin{bmatrix} p(x,y,t) \\ q(x,y,t) \end{bmatrix}$

3.3 Geometric Derivation of the Linear Equations

Equations 3.14 and 3.15 can be derived in a slightly different manner. This derivation is included because it may help one better understand flattening but this derivation is equivalent to the derivation presented in the previous section. Flattening can be seen as the process of mapping between coordinate systems or, equivalently, the process of mapping between tangent spaces (Lee, 2009). Figure 3.1 illustrates this mapping between coordinate systems where points in the left most coordinate system have been mapped to the points in the coordinate system on the right.

Specifically, define a point in the input image (the domain) to be

$$\mathbf{x} = (x, y, t) \tag{3.16}$$

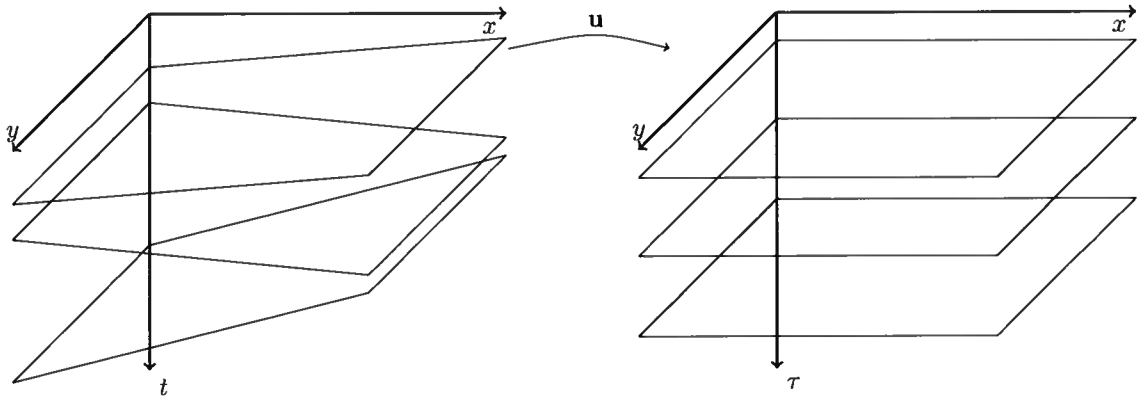


Figure 3.1: A cartoon of the flattening transform where the points in (x, y, t) space (on the left) are mapped to the points in (x, y, τ) space (on the right). The mapping $\mathbf{u}$ is constructed with the constraint that all events must be flat after transformation.

and then a point in the flattened coordinate system (the range) to be

$$\mathbf{u} = (x, y, \tau). \tag{3.17}$$

Flattening requires a mapping $\mathbf{u}(\mathbf{x})$ between these two coordinate spaces. Given a point $\mathbf{x}$ in the input space, $\mathbf{u}(\mathbf{x})$ is the corresponding point in the flattened space.

The mapping $\mathbf{u}(\mathbf{x})$ is found by considering how the events should be shifted such that all events are flattened. Unfortunately, finding a direct mapping of points between coordinate systems is difficult; one does not know if a point belongs to a flattened geologic interface without considering other points that belong to the same interface. It is much easier to find a mapping between vectors that are normal to the events of a seismic image because a normal vector that points vertically down is easily identified.

Fortunately, the relationship between transformed points and normal vectors is well known and commonly used in computer graphics, where vertices and normal vectors of a surface must be transformed simultaneously. Using a similar process, one can find a transform that maps normal vectors between the two coordinate systems, then work backwards to find a mapping of points between the coordinate systems. Note that in the previous chapter, I presented a method for measuring the slopes and equivalently the vectors normal to events of the seismic image using the structure tensor.

A derivation of the correspondence between vertex and normal transforms is presented in Shreiner *et al.* (2007). Consider a vector $\mathbf{n}$ that is normal to some surface and a vector

$\mathbf{w}$ which is perpendicular to $\mathbf{n}$ (tangent to the surface). Then by definition:

$$\mathbf{n}^T\mathbf{w} = 0. \tag{3.18}$$

If $\mathbf{M}$ is a non-singular matrix, then, the following is also true:

$$\mathbf{n}^T\mathbf{M}^{-1}\mathbf{M}\mathbf{w} = 0. \tag{3.19}$$

The above equations imply that the vector $(\mathbf{M}^{-1})^T\mathbf{n}$ is perpendicular to the vector $\mathbf{M}\mathbf{w}$ which is tangent to the surface after transformation. Shreiner *et al.* (2007) succinctly summarizes this property by stating that:

> Normal vectors are transformed by the inverse transpose of the transformation that transforms points.

Next, I use this property of transformations to derive the linear equations for flattening shifts. I first assume that the normal vectors (or slopes) estimated using the structure tensors correspond to smooth manifold surfaces that follow the events of the input image. Then, let $\mathbf{x}_0$ be an arbitrary point on a smooth manifold surface in the domain, and $\mathbf{u}_0$ be the corresponding point on the surface in the range:

$$\mathbf{u}(\mathbf{x}_0) = \mathbf{u}_0. \tag{3.20}$$

If the manifold surface in the domain is smooth and $\mathbf{x}$ is a point in an infinitesimally small neighborhood around $\mathbf{x}_0$, then

$$\mathbf{u}(\mathbf{x}) = \mathbf{u}_0 + \mathbf{J}(\mathbf{x} - \mathbf{x}_0), \tag{3.21}$$

where $\mathbf{J}$ denotes the Jacobian matrix of $\mathbf{u}(\mathbf{x})$. Note that the choice of the point $\mathbf{x}_0$ is arbitrary; the above equation could be written for any point in the input image. Now, define vectors normal $\mathbf{n}$ and $\mathbf{v}$ to their respective manifolds at points $\mathbf{x}_0$ and $\mathbf{u}_0$. By definition, for a point $\mathbf{x}$ on the surface in the domain and in the neighborhood of $\mathbf{x}_0$

$$\mathbf{n}^T(\mathbf{x} - \mathbf{x}_0) = 0 \tag{3.22}$$

and for a point $\mathbf{u}$ on the surface in the range and in the neighborhood of $\mathbf{u}_0$

$$\mathbf{v}^T(\mathbf{u} - \mathbf{u}_0) = 0. \tag{3.23}$$

Because vectors $\mathbf{n}$ and $\mathbf{v}$ are normal to the surface at their respective points they must be orthogonal to any vector along the surface in the infinitesimal neighborhood of $\mathbf{x}_0$ and $\mathbf{u}_0$. Then, using equation 3.21,

$$\mathbf{v}^T\mathbf{J}(\mathbf{x} - \mathbf{x}_0) = 0. \tag{3.24}$$

Comparing the above relation with equation 3.22 and simplifying yields:

$$\mathbf{v}^T\mathbf{J}(\mathbf{x} - \mathbf{x}_0) = \mathbf{n}^T(\mathbf{x} - \mathbf{x}_0) = 0. \tag{3.25}$$

The above equation must be satisfied for all points x on the surface and in the neighborhood of $\mathbf{x}_0$; therefore, $(\mathbf{J}^{-1})^T\mathbf{n}$ is a vector that is normal to the surface after transformation at $\mathbf{u}(\mathbf{x})$ and is parallel to $\mathbf{v}$. Thus, the flattened normals $\mathbf{v}$ are found by taking "the inverse transpose of the transformation that transforms points" and in this case, the transformation that transforms points is the Jacobian of $\mathbf{u}(\mathbf{x})$.

Finally, one must honor the constraint that the output image should be flattened; any event mapped into this space should be horizontal and planar. Figure 3.2 shows an example of surfaces before and after flattening. In order to enforce this constraint, the x and y components of normal vectors $\mathbf{v}$ should be zero. Therefore,

$$(\mathbf{J}^{-1})^T\mathbf{n} = \begin{bmatrix} 0 \\ 0 \\ c \end{bmatrix}, \tag{3.26}$$

where c is the length of $(\mathbf{J}^{-1})^T\mathbf{n}$. The above equation is simplified to the relations from the previous section by expanding the Jacobian of $\mathbf{u}(\mathbf{x})$. Specifically, the transform is

$$\mathbf{u}(x, y, t) = (x, y, t + \sigma(x, y, t)). \tag{3.27}$$

Expanding out the full inverse transpose of the Jacobian of $\mathbf{u}(\mathbf{x})$ and plugging into equa-

tion 3.26 yields

$$\begin{bmatrix} 1+\frac{\partial\sigma}{\partial t} & 0 & -\frac{\partial\sigma}{\partial x} \\ 0 & 1+\frac{\partial\sigma}{\partial t} & -\frac{\partial\sigma}{\partial y} \\ 0 & 0 & 1 \end{bmatrix} \mathbf{n} = \begin{bmatrix} 0 \\ 0 \\ c \end{bmatrix}. \tag{3.28}$$

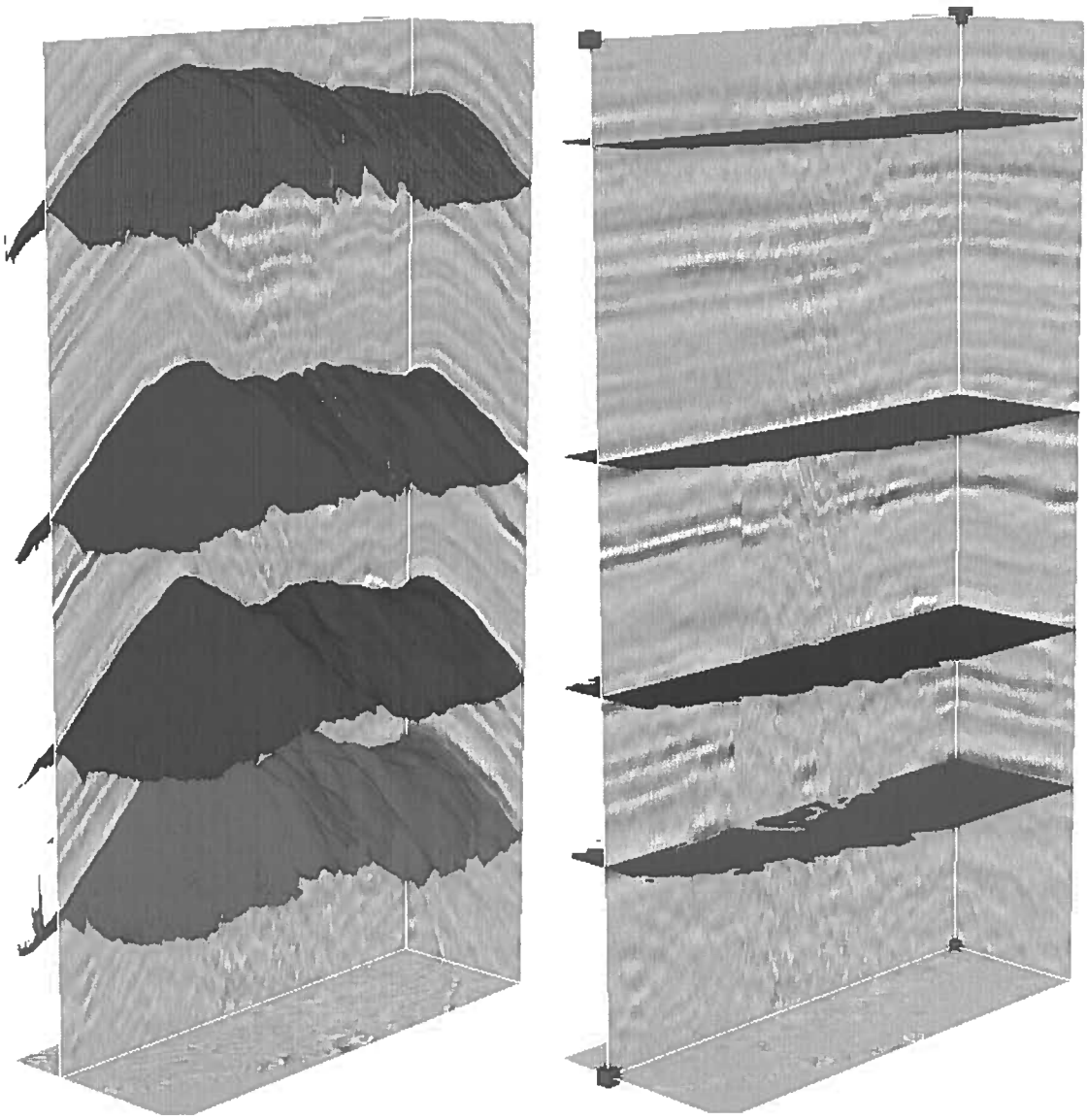
Further simplification of the above equation yields equations 3.14 and 3.15. While this derivation is slightly longer than the one presented in the previous section, it uses concepts from differential geometry and computer graphics to produce an equivalent result. The surfaces plotted in Figure 3.2, when combined with the derivation presented in this chapter, also provide a useful insight into how the flattening transform is applied to a seismic image.

3.4 Change of Variables

In the previous sections, I derived the linear partial differential equations (Table 3.1) that are used to compute the shift function $\sigma(x, y, t)$. A least-squares best solution to these equations can easily be found by using an iterative optimization method such as conjugate gradients. Unfortunately, directly solving for the shifts may be problematic. There are many possible ways to flatten a given set of slopes and some solutions are preferable to others. In this section, I outline why some shift functions may be preferable to others and show how to solve for them.

Figure 3.3 illustrates two possible problems one can encounter when flattening a seismic image. In Figure 3.3a, all of the flattened reflectors have been shifted down when compared to the flattened reflectors shown in Figure 3.3c. This shifting means that a constant value has been added to the shift function $\sigma(x, y, t)$. A second possible problem is illustrated in Figure 3.3b; a linear trend has been added to the shift function which causes the flattened reflectors to squeeze together. Note that in both cases (Figures 3.3a and 3.3b), the reflectors are still perfectly flat. The shift functions used to compute both images satisfy a least-squares solution to the flattening PDEs. Specifically, given a shift function $\sigma(x, y, t)$ that is a least-squares solution to the flattening PDEs for a given set of slopes $p(x, y, t)$ and $q(x, y, t)$, another valid solution is

$$\sigma(x, y, t) + C_0 + C_1 t \tag{3.29}$$



(a) (b)

Figure 3.2: The surfaces in (a) have been mapped to the correspondingly colored surfaces in (b). While only four surfaces are shown here, the actual number of transformed surfaces corresponds to the number of samples in the τ direction of the output image.

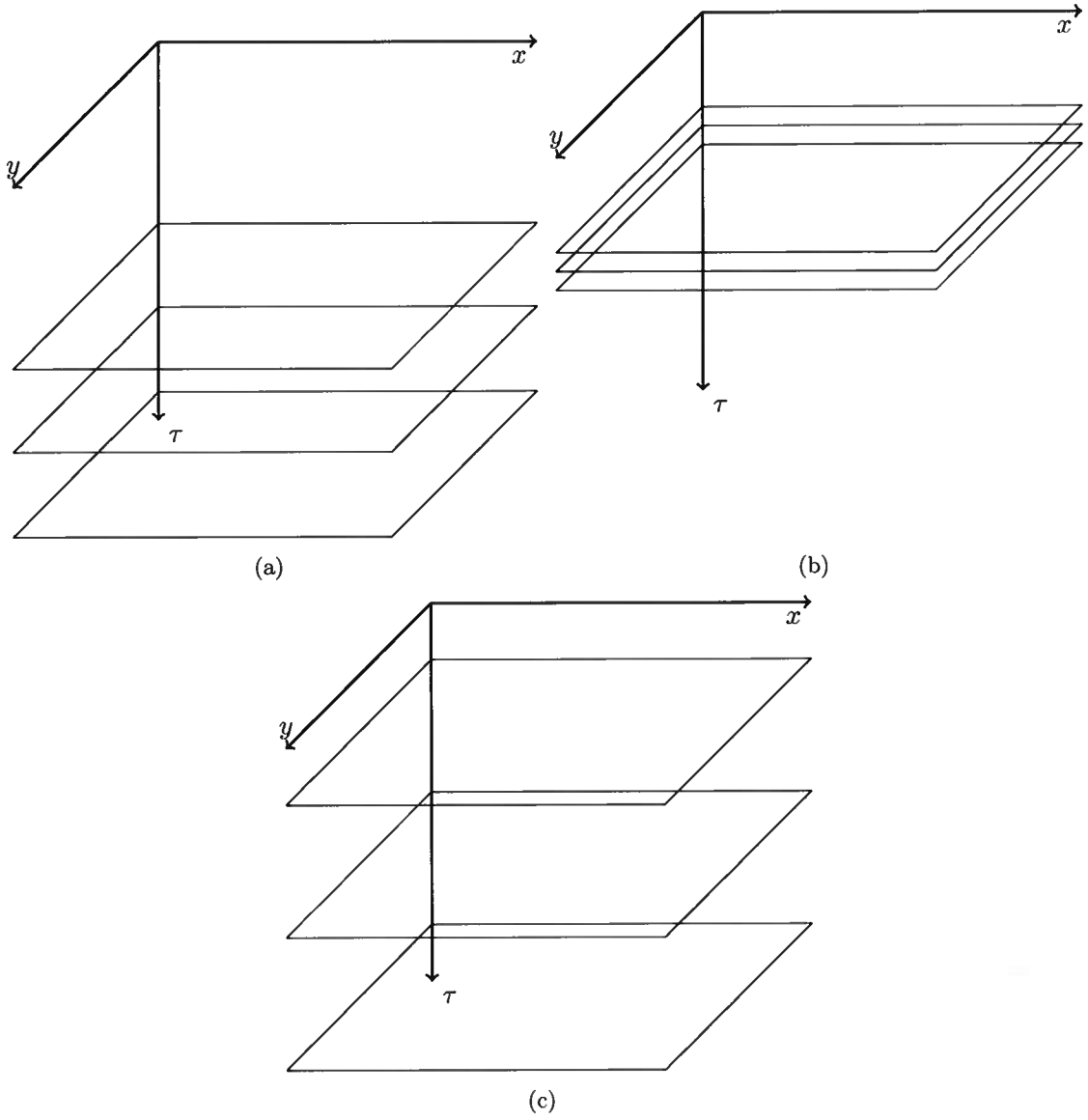


Figure 3.3: A cartoon showing three possible ways to flatten three reflectors. In (a) the reflectors have been shifted down; In (b) the reflectors have been squeezed together; and in (c) they have been spaced equally throughout the flattened image.

for any nonzero constants C_0 and C_1. A nonzero C_0 adds a constant shift up or down to all of the reflectors in the flattened image (Figure 3.3a), and a nonzero C_1 corresponds to a vertical stretching or squeezing of the flattened reflectors (Figure 3.3b). To understand why these constant and linear shifts are unresolved, one must consider how this inversion problem is solved. In a typical inversion problem, one seeks to find a model vector $\mathbf{m}$ that fits a given data vector $\mathbf{d}$ after applying a forward operator $\mathbf{F}$:

$$\mathbf{d} \approx \mathbf{F}(\mathbf{m}). \tag{3.30}$$

In the case of flattening:

- the data vector $\mathbf{d}$ contains the samples of the local slopes $p(x, y, t)$ and $q(x, y, t)$
- the model vector $\mathbf{m}$ contains the samples of the shift function $\sigma(x, y, t)$
- the forward operator $\mathbf{F}$ is constructed such that $\mathbf{F}(\mathbf{m})$ contains the samples from the left-hand sides of the linear PDEs in equations 3.14 and 3.15.

Since, in this case, $\mathbf{F}$ is a linear operator the least-squares objective function for such an inversion problem is written as

$$||\mathbf{Fm} - \mathbf{d}|| \tag{3.31}$$

or

$$(\mathbf{m}^T\mathbf{F}^T - \mathbf{d}^T)(\mathbf{Fm} - \mathbf{d}). \tag{3.32}$$

Note that equation 3.32 applies the forward operator twice (once as $\mathbf{F}$ and again as $\mathbf{F}^T$) and that the forward operator $\mathbf{F}$ contains derivatives with respect to all coordinate directions. Consequently, when computing a solution to equation 3.32, all of the constant and linear components of the model $\mathbf{m}$ will be removed.

A third problem is common in many inversion problems and arises when there are many possible solutions to a set of equations, but one would prefer the solution that has the least unneeded complexity. This type of regularization is performed in the previous chapter when I introduced the smoothing of the shift function in the third row of equation 2.12.

All of the aforementioned problems can be overcome by using the model reparameterization technique shown in Harlan (1995). This is done by introducing a new operator $\mathbf{S}$

which is defined to suppress the parts of the model $\mathbf{m}$ that are unwanted. Harlan (1995) states that

> This operator [$\mathbf{S}$] should be designed to preserve simplicity and suppress complexity, although without destroying complexity entirely.

Now, instead of solving directly for the model $\mathbf{m}$, I solve for a new variable $\tilde{\mathbf{m}}$ which is a reparameterized version of the model, where $\mathbf{m} = \mathbf{S}\tilde{\mathbf{m}}$, by minimizing

$$(\tilde{\mathbf{m}}^T\mathbf{S}^T\mathbf{F}^T - \mathbf{d}^T)(\mathbf{F}\mathbf{S}\tilde{\mathbf{m}} - \mathbf{d}). \tag{3.33}$$

Note that, after solving for a reparameterized model $\tilde{\mathbf{m}}$, one can quickly find the original model $\mathbf{m}$ by applying the operator $\mathbf{S}$.

The next step is to define the operator $\mathbf{S}$. Typically, this reparameterization technique is used to resolve the problem of over-complex models but in the case of flattening, one still has the problem of unwanted constant or linear shifts, which are the simplest components of the model. The key realization is that one can use the model reparameterization trick to suppress any parts of the model that are unwanted (shifts, squeezes, and unnecessary complexity) while enhancing the parts of the model that are desirable (smoothness). In order to accomplish this, I define $\mathbf{S}$ to be

$$\mathbf{S} \equiv (\mathbf{I} - \mathbf{e_1}\mathbf{e_1}^T)(\mathbf{I} - \mathbf{e_0}\mathbf{e_0}^T)\mathbf{S}_x\mathbf{S}_y\mathbf{S}_t, \tag{3.34}$$

where

- $\mathbf{S}_t$ is a symmetric smoothing in time
- $\mathbf{S}_y$ is a symmetric smoothing in the crossline direction
- $\mathbf{S}_x$ is a symmetric smoothing in the inline direction
- $\mathbf{e_0}$ is a unit vector with each element equal to $1/\sqrt{N}$, where N is the number of samples in the image
- $\mathbf{e_1}$ is a vector linear ramp in time such that $\mathbf{e_1}^T\mathbf{e_0} = 0$

This reparameterization seeks to remove the parts of the model that are undesirable:

- $\mathbf{S}_x$, $\mathbf{S}_y$, and $\mathbf{S}_t$ remove unnecessary roughness from the shift function

- the $\mathbf{I} - \mathbf{e_0}\mathbf{e_0}^T$ factor removes constant shifts by subtracting out the mean shift
- the $\mathbf{I} - \mathbf{e_1}\mathbf{e_1}^T$ factor removes trends in the shifts that are linear in t.

Any of the above symmetric smoothing filters ($\mathbf{S}_x$, $\mathbf{S}_y$, and $\mathbf{S}_t$) can be implemented as a generic symmetric filter but this can be improved upon by choosing a filter that conforms to the guideline of enhancing the parts of the model that are desirable. Smoothness in the shifts along the layers of the input image is desirable; thus, it is preferable to implement the symmetric smoothing filters as structure preserving filters (Hale, 2009). Structure preserving filters smooth along and not across the events of a seismic image, which allows the resulting shift function to also be smooth along the layers. Finally, note that

$$(\mathbf{I} - \mathbf{e_1}\mathbf{e_1}^T)(\mathbf{I} - \mathbf{e_0}\mathbf{e_0}^T) = (\mathbf{I} - \mathbf{e_1}\mathbf{e_1}^T - \mathbf{e_0}\mathbf{e_0}^T) \tag{3.35}$$

which means that the above equation can be applied in two passes over a 3D image. The first pass computes $m_0 = \mathbf{e_1}^T\tilde{\mathbf{m}}$ and $m_1 = \mathbf{e_0}^T\tilde{\mathbf{m}}$ where m_0 and m_1 are temporary variables. Then, the second pass computes $\tilde{\mathbf{m}}_{out} = \tilde{\mathbf{m}} - \mathbf{e_1}m_0 - \mathbf{e_0}m_1$, where $\tilde{\mathbf{m}}_{out}$ is result of applying the operator shown in equation 3.35. Finally, each factor of $\mathbf{S}$ is symmetric; therefore, the transpose of $\mathbf{S}$, which is required by equation 3.33, is just the same factors applied in reverse order.

This reparameterization technique could be replaced by adding multiple regularization terms but this reparameterization technique is preferable. This is because it models simplicity directly and does not add extra complexities to the objective function. To paraphrase Harlan (1995), minimizing an objective function that only suppresses complexity through a regularization term should be avoided because the model becomes complex quickly and later iterations remove this complexity. The reparameterization technique is preferable because it models the desired simplicity directly and goes directly to the desired solution.

3.5 Results and Conclusions

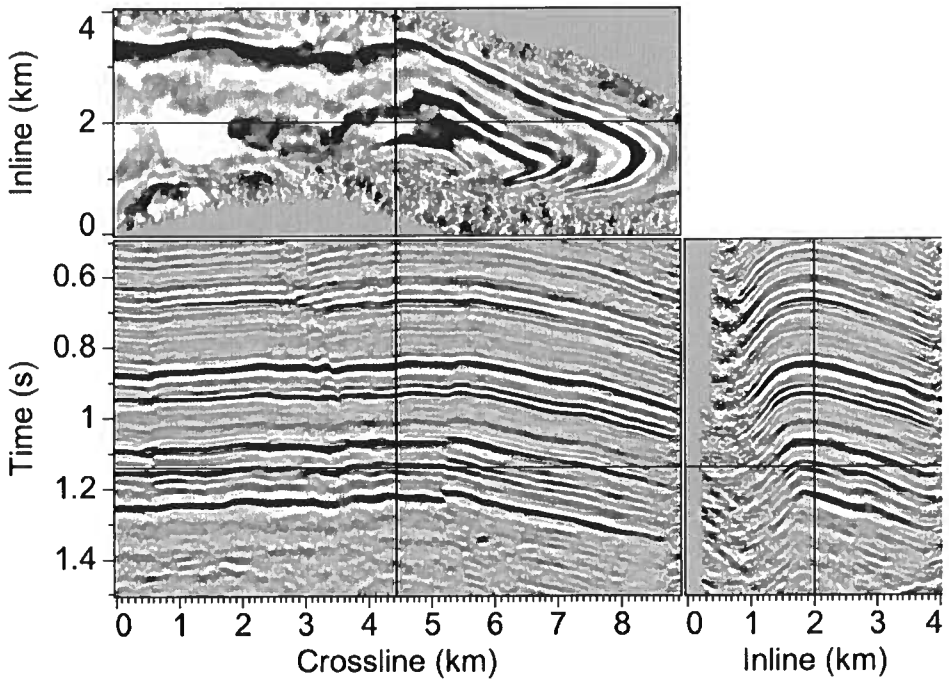
I have applied the linear PDEs to a 3D image from the Teapot Dome. Teapot Dome is located in Natrona County, Wyoming and currently has over 600 active wells (Personal Communication, Brian Black, 2009). The dome structure present in the seismic image provides an ideal example for flattening, as stratigraphic features within the image may be obscured by the dome shaped layers.

The images before and after flattening are shown in Figure 3.4. The arrows in Figure 3.4b indicate two possible channels in the flattened image. These channels are difficult to detect in the input image, but are quickly recognized when scanning through the horizontal (constant-τ) slices of the flattened image. The flattening used to find these channels was completely automated and did not require any manual horizon picking. In order for an interpreter to detect these features in a traditional picking based manner, they would have had to know beforehand to flatten on the horizon located at 1.42 seconds in the input image.

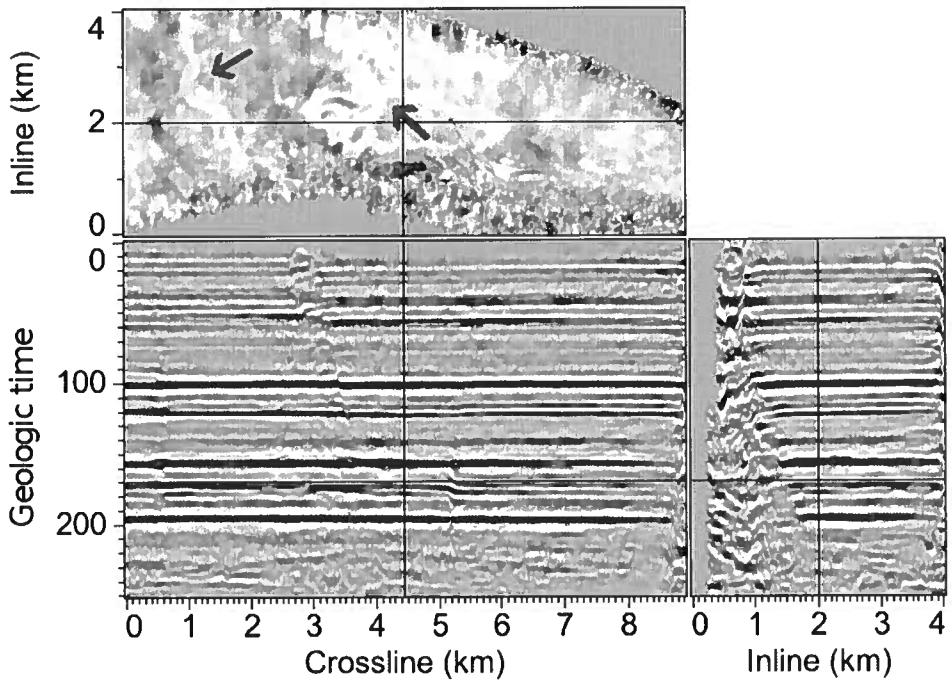
Figure 3.5 shows two additional horizontal slices from the flattened image. A possible astrobleme (Personal Communication, Tom Davis, 2009) can be seen in Figure 3.5a and another possible channel can be seen in Figure 3.5b. In general, it is much easier for one to see stratigraphic features in the flattened domain, and an automatic flattening allows interpreters to spend less time picking horizons and more time identifying such stratigraphic features. Table 3.2 shows a comparison of the runtimes of various ways of solving for a shift function. All of the algorithms were tested on the Teapot Dome seismic image and were run on a computer with dual quad-core 3 gigahertz processors and enough memory to avoid swapping. The average absolute value of the inline and crossline slopes *after flattening* was computed to give a quantitative comparison of the flattening results. A zero value of absolute average slope corresponds to perfectly flat events in the output image but some residual slope is to be expected in practice. Qualitatively, all of the tested flattening methods provided similar flattened output images. The total number of conjugate gradient iterations required is also reported for each test. For the nonlinear tests, the total number of conjugate gradient iterations required is the sum of the conjugate gradient iterations required at each of the linearized k iterations. The number of conjugate gradient iterations required does not exactly correspond with the runtime for the nonlinear tests because each linearized k iteration incurs additional overhead.

The first thing to be noted from these results is that the Harlan (1995) reparameterization method provides a significant speedup in the computation of the shift function.

Two sets of results are provided for the nonlinear methods. The results marked as (FC) have been taken to full convergence at each of the linearized k iterations and the nonlinear results not marked as (FC) have been optimized to avoid unnecessary computations. In these tests, full convergence was reached when the norm of the residuals for an iteration of the conjugate gradient method was less than 0.01 times the norm of the data

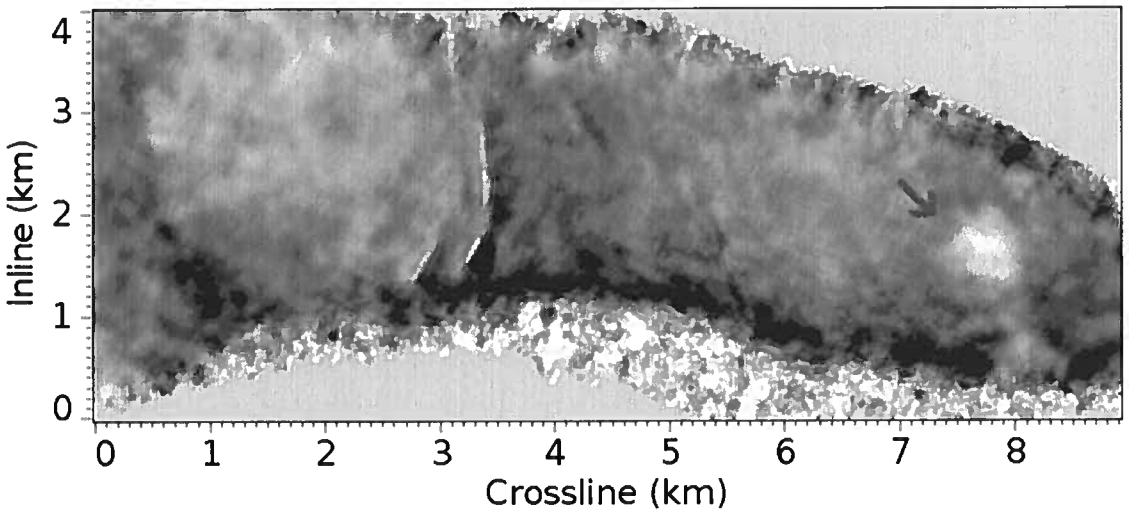


(a)

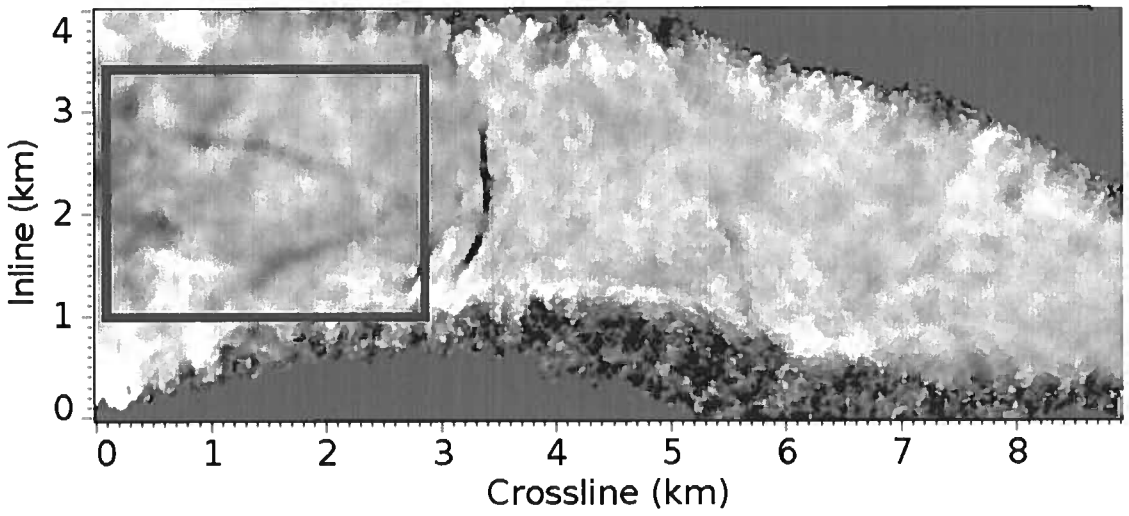


(b)

Figure 3.4: Each of these figures ((a) and (b)) contain three axis-aligned slices from a 3D seismic image. The black lines indicate the location of the slices in the other directions. (a) shows the image before flattening and (b) shows the corresponding slices after flattening. Two possible channels are indicated by the red arrows in (b).



(a)



(b)

Figure 3.5: Two horizontal constant-τ slices after flattening. In (a) a possible astrobleme is indicated by the red arrow and (b) shows another possible channel formation which is indicated by the red box.

vector **d**. Note that the optimized tests are much faster. In the optimized tests, the next linearized k iteration was started after 10 conjugate gradient iterations in the nonlinear with reparameterization test and 200 conjugate gradient iterations in the nonlinear without reparameterization test. The optimal number of conjugate gradient iterations for the optimized nonlinear tests was found experimentally and is dependent on the seismic image that is being flattened. This highlights the advantage of the linear method because with the linear method no such experimentation is needed.

In conclusion, I have shown that flattening can be posed as a linear inverse problem, flattening is related to the mapping of tangent spaces, and a change of variables speeds convergence.

Table 3.2: Comparison of methods

Method	Runtime (s)	Total conjugate gradient iterations	Linearized k iterations	Absolute average slope (ms/km)
Linear reparameterization	41.2	35	N/A	56.1
Linear no reparameterization	357.5	766	N/A	56.4
Nonlinear reparameterization	58.1	40	4	55.6
Nonlinear no reparameterization	343.2	800	4	56.7
Nonlinear reparameterization (FC)	725.9	959	5	55.5
Nonlinear no reparameterization (FC)	815.3	1976	3	56.3

References

Chopra, Satinder, & Marfurt, Kurt J. 2007. *Seismic Attributes for Prospect ID and Reservoir Characterization.* Society of Exploration Geophysicists.

Claerbout, Jon F. 1992. *Earth Soundings Analysis: Processing Versus Inversion.* Blackwell Science.

de Groot, Paul, de Bruin, Geert, & Hemstra, Nanne. 2006. How to create and use 3D Wheeler transformed seismic volumes. *SEG Technical Program Expanded Abstracts*, **25**(1), 1038–1042.

Fehmers, Gijs C., & Höcker, Christian F. W. 2003. Fast structural interpretation with structure-oriented filtering. *Geophysics*, **68**(4), 1286–1293.

Fomel, Sergey. 2002. Applications of plane-wave destruction filters. *Geophysics*, **67**(6), 1946–1960.

Gao, Dengliang. 2009. 3D seismic volume visualization and interpretation: An integrated workflow with case studies. *Geophysics*, **74**(1), W1–W12.

Hale, Dave. 2009. Image-guided blended neighbor interpolation of scattered data. *SEG Technical Program Expanded Abstracts*, **28**(1), 1127–1131.

Harlan, William S. 1995. *Regularization by Model Reparameterization.* `http://billharlan.com/pub/papers/regularization.html`.

Labrunye, Emmanuel, Winkler, Christophe, Borgese, Cédric, Mallet, Jean-Laurent, & Jayr, Stanislas. 2009. New 3D flattened space for seismic interpretation. *SEG Technical Program Expanded Abstracts*, **28**(1), 1132–1136.

Lee, Jeffrey M. 2009. *Manifolds and Differential Geometry.* American Mathematical Society.

Lomask, Jesse. 2006. *Seismic volumetric flattening and segmentation.* Ph.D. thesis, Stanford University.

Lomask, Jesse, & Guitton, Antoine. 2007. Volumetric flattening: an interpretation tool. *The Leading Edge*, **26**(7), 888–897.

Lomask, Jesse, Guitton, Antoine, Fomel, Sergey, Claerbout, Jon, & Valenciano, Alejandro A. 2006. Flattening without picking. *Geophysics*, **71**(4), P13–P20.

Pauget, Fabien, Lacaze, Sébastien, & Valding, Thomas. 2009. A global approach in seismic interpretation based on cost function minimization. *SEG Technical Program Expanded Abstracts*, **28**(1), 2592–2596.

Shreiner, Dave, Woo, Mason, Neider, Jackie, Davis, Tom, & Board, OpenGL Architecture Review. 2007. *OpenGL(R) Programming Guide: The Official Guide to Learning OpenGL(R), Version 2.1.* Addison-Wesley Professional.

Stark, Tracy J. 1996. Surface slice generation and interpretation: A review. *The Leading Edge*, **15**(7), 818–819.

Stark, Tracy J. 2003. Unwrapping instantaneous phase to generate a relative geologic time volume. *SEG Technical Program Expanded Abstracts*, **22**(1), 1707–1710.

Stark, Tracy J. 2004. Relative geologic time (age) volumes—Relating every seismic sample to a geologically reasonable horizon. *The Leading Edge*, **23**(9), 928–932.

Stark, Tracy J. 2005. Generation of a 3D seismic "Wheeler Diagram" from a high resolution Age Volume. *SEG Technical Program Expanded Abstracts*, **24**(1), 782–785.

van Vliet, Lucas J., & Verbeek, Piet W. 1995. Estimators for Orientation and Anisotropy in Digitized Images. *Pages 442–450 of: Proceedings of the first annual conference of the Advanced School for Computing and Imaging ASCI95, Heijen (The Netherlands).* Advanced School for Computing and Imaging.

Zeng, Hongliu, Backus, Milo M., Barrow, Kenneth T., & Tyler, Noel. 1998. Stratal slicing, Part I: Realistic 3-D seismic model. *Geophysics*, **63**(2), 502–513.