Chapter 1

Zero-offset migration

In chapter **??** we discussed methods of imaging horizontal reflectors and of estimating velocity $v(z)$ from the offset dependence of seismic recordings. In this chapter, we turn our attention to imaging methods for *dipping* reflectors. These imaging methods are usually referred to as "migration" techniques.

Offset is a geometrical nuisance when reflectors have dip. For this reason, we develop migration methods here and in the next chapter for forming images from hypothetical *zero-offset* seismic experiments. Although there is usually ample data recorded near zero-offset, we never record purely zero-offset seismic data. However, when we consider offset and dip together in chapter 8 we will encounter a widely-used technique (dip-moveout) that often converts finite-offset data into a useful estimate of the equivalent zero-offset data. For this reason, **zero-offset migration** methods are widely used today in industrial practice. Furthermore the concepts of zero-offset migration are the simplest starting point for approaching the complications of finite-offset migration.

1.1 MIGRATION DEFINED

The term "migration" probably got its name from some association with movement. A casual inspection of migrated and unmigrated sections shows that migration causes many reflection events to shift their positions. These shifts are necessary because the *apparent* positions of reflection events on unmigrated sections are generally not the *true* positions of the reflectors in the earth. It is not difficult to visualize why such "acoustic illusions" occur. An analysis of a zero-offset section shot above a dipping reflector illustrates most of the key concepts.

1.1.1 A dipping reflector

Consider the zero-offset seismic survey shown in Figure 1.1. This survey uses one source-receiver pair, and the receiver is always at the same location as the source. At each position, denoted by $S_1, S_2, \mathrm{and} S_3$ in the figure, the source emits waves and the receiver records the echoes as a single seismic trace. After each trace is recorded, the source-receiver pair is moved a small distance and the experiment is repeated.

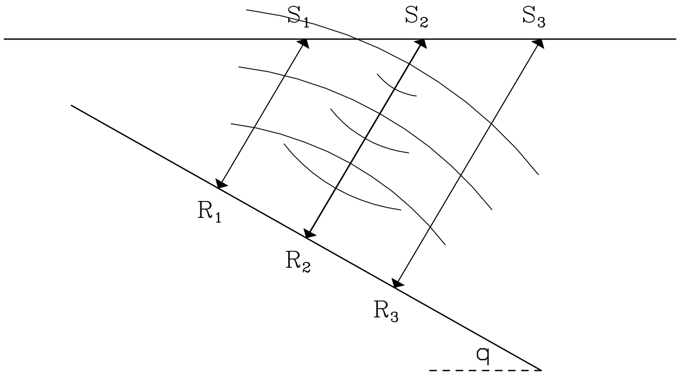

Figure 1.1: Raypaths and wavefronts for a zero-offset seismic line shot above a dipping reflector. The earth's propagation velocity is constant. krch-reflexpt [ER]

As shown in the figure, the source at S_2 emits a spherically-spreading wave that bounces off the reflector and then returns to the receiver at S_2. The raypaths drawn between S_i and R_i are orthogonal to the reflector and hence are called *normal rays*. These rays reveal how the zero-offset section misrepresents the truth. For example, the trace recorded at S_2 is dominated by the reflectivity near reflection point R_2, where the normal ray from S_2 hits the reflector. If the zero-offset section corresponding to Figure 1.1 is displayed, the reflectivity at R_2 will be falsely displayed as though it were directly beneath S_2, which it certainly is not. This lateral mispositioning is the first part of the illusion. The second part is vertical: if converted to depth, the zero-offset section will show R_2 to be deeper than it really is. The reason is that the slant path of the normal ray is longer than a vertical shaft drilled from the surface down to R_2.

1.1.2 Dipping-reflector shifts

A little geometry gives simple expressions for the horizontal and vertical position errors on the zero-offset section, which are to be corrected by migration. Figure 1.2 defines the required quantities for a reflection event recorded at S corresponding to the reflectivity at R. The two-

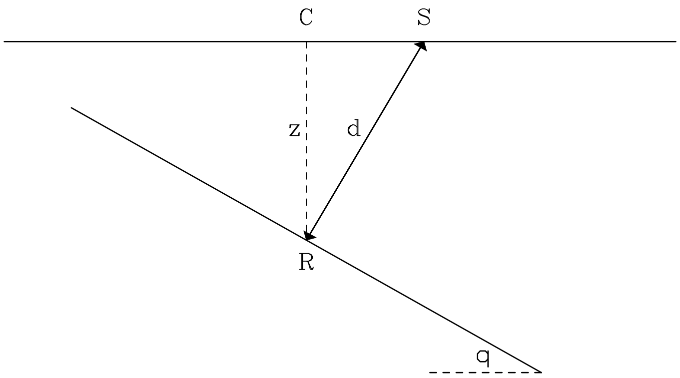

Figure 1.2: Geometry of the normal ray of length d and the vertical "shaft" of length z for a zero-offset experiment above a dipping reflector. krch-reflkine [ER]

way travel time for the event is related to the length d of the normal ray by

$$t \quad = \quad \frac{2d}{v} \ , \tag{1.1}$$

where v is the constant propagation velocity. Geometry of the triangle CRS shows that the true depth of the reflector at R is given by

$$z \quad = \quad d \, \cos\theta \ , \tag{1.2}$$

and the lateral shift between true position C and false position S is given by

$$\Delta x \quad = \quad d \, \sin\theta \quad = \quad \frac{v\,t}{2} \, \sin\theta \ . \tag{1.3}$$

It is conventional to rewrite equation (1.2) in terms of two-way *vertical* traveltime τ:

$$\tau \quad = \quad \frac{2z}{v} \quad = \quad t \, \cos\theta \ . \tag{1.4}$$

Thus both the vertical shift $t - \tau$ and the horizontal shift Δx are seen to vanish when the dip angle θ is zero.

1.1.3 Hand migration

Geophysicists recognized the need to correct these positioning errors on zero-offset sections long before it was practical to use computers to make the corrections. Thus a number of hand-migration techniques arose. It is instructive to see how one such scheme works. Equations (1.3) and (1.4) require knowledge of three quantities: t, v, and θ. Of these, the event time t is readily measured on the zero-offset section. The velocity v is usually *not* measurable on the zero offset section and must be estimated from finite-offset data, as was shown in chapter **??**. That leaves the dip angle θ. This can be related to the reflection slope p of the observed event, which is measurable on the zero-offset section:

$$p_0 \quad = \quad \frac{\partial t}{\partial y} \ , \tag{1.5}$$

where y (the midpoint coordinate) is the location of the source-receiver pair. The slope p_0 is sometimes called the *"time-dip of the event"* or more loosely as the *"dip of the event"*. It is obviously closely related to Snell's parameter, which we discussed in chapter **??**. The relationship between the measurable time-dip p_0 and the dip angle θ is called "**Tuchel's law**":

$$\sin\theta \quad = \quad \frac{v\,p_0}{2} \ . \tag{1.6}$$

This equation is clearly just another version of equation (**??**), in which a factor of 2 has been inserted to account for the two-way traveltime of the zero-offset section.

Rewriting the migration shift equations in terms of the measurable quantities t and p yields usable "hand-migration" formulas:

$$\Delta x \quad = \quad \frac{v^2 \, p \, t}{4} \tag{1.7}$$

$$\tau \quad = \quad t \, \sqrt{1 \, - \, \frac{v^2 p^2}{4}} \ . \tag{1.8}$$

Hand migration divides each observed reflection event into a set of small segments for which p has been measured. This is necessary because p is generally not constant along real seismic events. But we can consider more general events to be the union of a large number of very small dipping reflectors. Each such segment is then mapped from its unmigrated (y,t) location to its migrated (y,τ) location based on the equations above. Such a procedure is sometimes also known as "map migration."

Equations (1.7) and (1.8) are useful for giving an idea of what goes on in zero-offset migration. But using these equations directly for practical seismic migration can be tedious and error-prone because of the need to provide the time dip p as a separate set of input data values as a function of y and t. One nasty complication is that it is quite common to see *crossing events* on zero-offset sections. This happens whenever reflection energy coming from two different reflectors arrives at a receiver at the same time. When this happens the time dip p becomes a *multi-valued* function of the (y,t) coordinates. Furthermore, the recorded wavefield is now the sum of two different events. It is then difficult to figure out which part of summed amplitude to move in one direction and which part to move in the other direction.

For the above reasons, the seismic industry has generally turned away from hand-migration techniques in favor of more automatic methods. These methods require as inputs nothing more than

- The zero-offset section
- The velocity v .

There is no need to separately estimate a $p(y,t)$ field. The automatic migration program somehow "figures out" which way to move the events, even if they cross one another. Such automatic methods are generally referred to as *"wave-equation migration"* techniques, and are the subject of the remainder of this chapter. But before we introduce the automatic migration methods, we need to introduce one additional concept that greatly simplifies the migration of zero-offset sections.

1.1.4 A powerful analogy

Figure 1.3 shows two wave-propagation situations. The first is realistic field sounding. The second is a thought experiment in which the reflectors in the earth suddenly explode. Waves from the hypothetical explosion propagate up to the earth's surface where they are observed by a hypothetical string of geophones.

Notice in the figure that the ray paths in the field-recording case seem to be the same as those in the **exploding-reflector** case. It is a great conceptual advantage to imagine that the two wavefields, the observed and the hypothetical, are indeed the same. If they are the same, the many thousands of experiments that have really been done can be ignored, and attention can be focused on the one hypothetical experiment. One obvious difference between the two cases is that in the field geometry waves must first go down and then return upward along

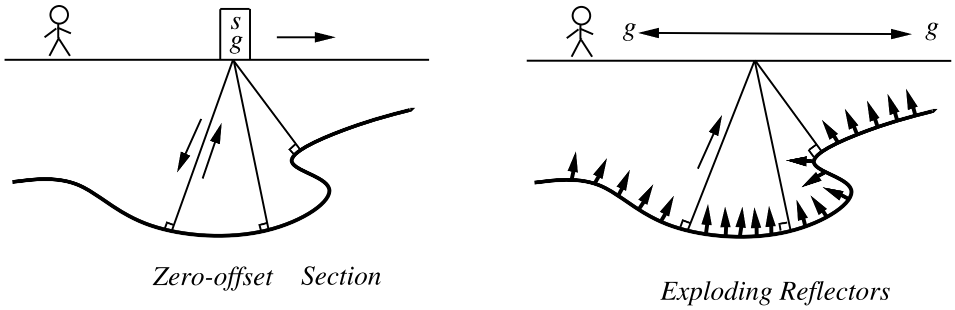

Figure 1.3: Echoes collected with a source-receiver pair moved to all points on the earth's surface (left) and the "exploding-reflectors" conceptual model (right). krch-expref [NR]

the same path, whereas in the hypothetical experiment they just go up. Travel time in field experiments could be divided by two. In practice, the data of the field experiments (two-way time) is analyzed assuming the sound velocity to be half its true value.

Figure 1.4 shows how points making up a line reflector diffract to a line reflection, and how points making up a line reflection migrate to a line reflector.

1.1.5 Limitations of the exploding-reflector concept

The exploding-reflector concept is a powerful and fortunate analogy. It enables us to think of the data of many experiments as though it were a single experiment. Unfortunately, the exploding-reflector concept has a serious shortcoming. No one has yet figured out how to extend the concept to apply to data recorded at nonzero offset. Furthermore, most data is recorded at rather large offsets. In a modern marine prospecting survey, there is not one hydrophone, but hundreds, which are strung out in a cable towed behind the ship. The recording cable is typically 2-3 kilometers long. Drilling may be about 3 kilometers deep. So in practice the angles are big. Therein lie both new problems and new opportunities, none of which will be considered until chapter 8.

Furthermore, even at zero offset, the exploding-reflector concept is not quantitatively correct. For the moment, note three obvious failings: First, Figure 1.5 shows rays that are not predicted by the exploding-reflector model. These rays will be present in a zero-offset section. Lateral velocity variation is required for this situation to exist.

Second, the exploding-reflector concept fails with **multiple reflections**. For a flat sea floor with a two-way travel time t_1, multiple reflections are predicted at times $2t_1$, $3t_1$, $4t_1$, etc. In the exploding-reflector geometry the first multiple goes from reflector to surface, then from surface to reflector, then from reflector to surface, for a total time $3t_1$. Subsequent multiples occur at times $5t_1$, $7t_1$, etc. Clearly the multiple reflections generated on the zero-offset section

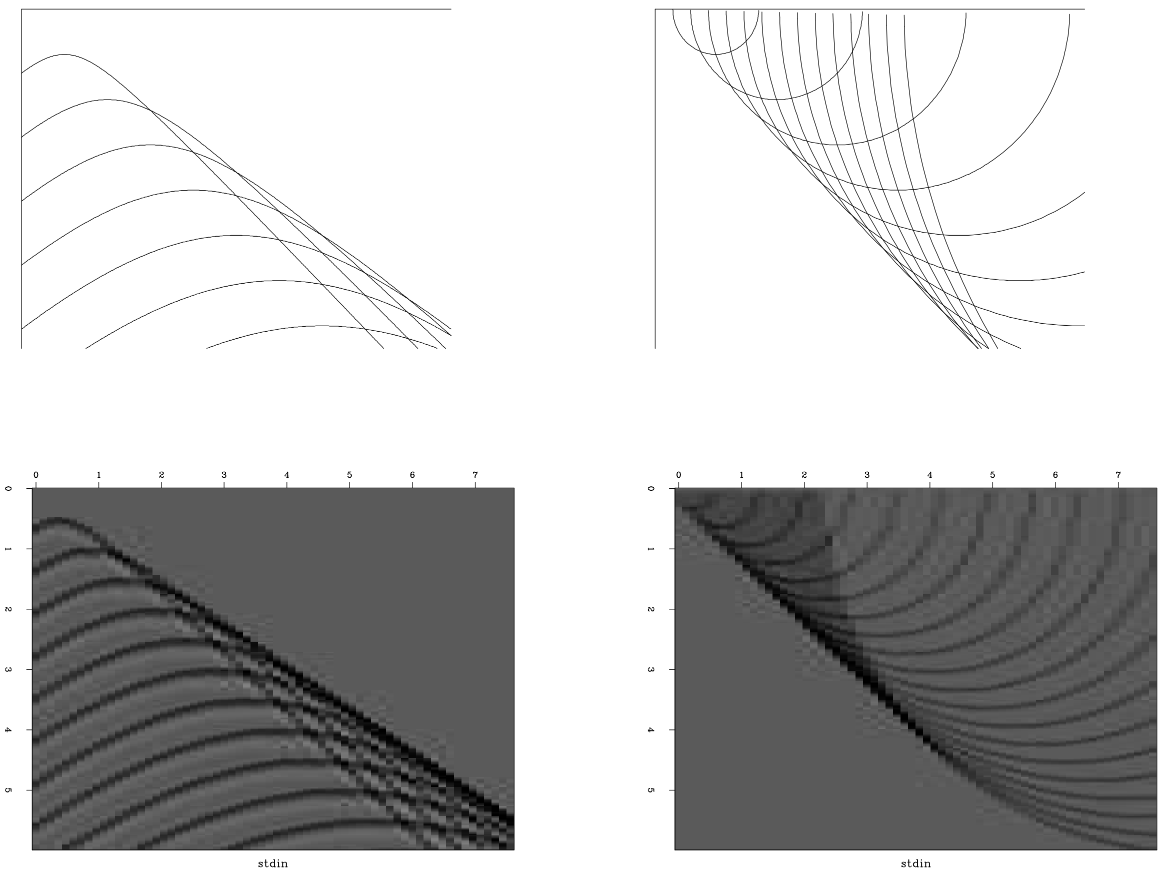

Figure 1.4: Left is a superposition of many hyperbolas. The top of each hyperbola lies along a straight line. That line is like a reflector, but instead of using a continuous line, it is a sequence of points. Constructive interference gives an apparent reflection off to the side. Right shows a superposition of semicircles. The bottom of each semicircle lies along a line that could be the line of an observed plane wave. Instead the plane wave is broken into point arrivals, each being interpreted as coming from a semicircular mirror. Adding the mirrors yields a more steeply dipping reflector. krch-dip [ER]

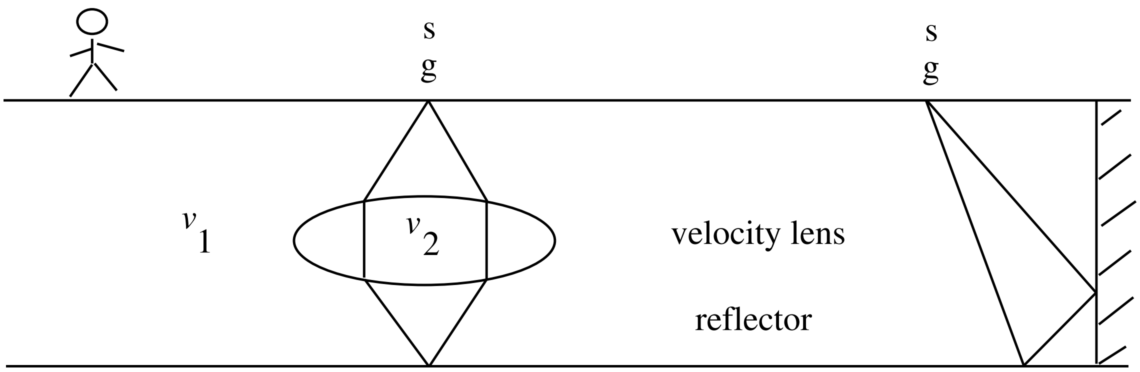

Figure 1.5: Two rays, not predicted by the exploding-reflector model, that would nevertheless be found on a zero-offset section. krch-fail [NR]

differ from those of the exploding-reflector model.

The third failing of the exploding-reflector model is where we are able to see waves bounced from both sides of an interface. The exploding-reflector model predicts the waves emitted by both sides have the same polarity. The physics of reflection coefficients says reflections from opposite sides have opposite polarities.

1.2 HYPERBOLA PROGRAMMING

First we will look at the simplest, most tutorial migration subroutine I could devise. Then we will write an improved version and look at some results.

1.2.1 Tutorial Kirchhoff code

Subroutine `kirchslow()` below is the best tutorial **Kirchhoff migration**-modeling program I could devise. A nice feature of this program is that it works OK while the edge complications do not clutter it. The program copies information from data space `data(it,iy)` to model space `modl(iz,ix)` or vice versa. Notice that of these four axes, three are independent (stated by loops) and the fourth is derived by the circle-hyperbola relation $t^2 = \tau^2 + x^2/v^2$. Subroutine `kirchslow()` for `adj=0` copies information from model space to data space, i.e. from the hyperbola top to its flanks. For `adj=1`, data summed over the hyperbola flanks is put at the hyperbola top.

```
# Kirchhoff migration and diffraction.  (tutorial, slow)
#
subroutine kirchslow(   adj, add,  velhalf, t0,dt,dx, modl,nt,nx,  data)
integer ix,iy,it,iz,nz, adj, add,                             nt,nx
real x0,y0,dy,z0,dz,t,x,y,z,hs,    velhalf, t0,dt,dx, modl(nt,nx), data(nt,nx)
call adjnull(           adj, add,                     modl,nt*nx,  data,nt*nx)
x0=0.;  y0=0;  dy=dx;  z0=t0;  dz=dt; nz=nt
do ix= 1, nx {  x = x0 + dx * (ix-1)
do iy= 1, nx {  y = y0 + dy * (iy-1)
do iz= 1, nz {  z = z0 + dz * (iz-1)           # z = travel-time depth
        hs=      (x-y) / velhalf
        t = sqrt( z * z  +  hs * hs )
        it = 1.5 + (t-t0) / dt
        if( it <= nt )
                if( adj == 0 )
                        data(it,iy) = data(it,iy) + modl(iz,ix)
                else
                        modl(iz,ix) = modl(iz,ix) + data(it,iy)

        }}}
return; end
```

Notice how this program has the ability to create a hyperbola given an input impulse in (x,z)-space, and a circle given an input impulse in (x,t)-space.

The three loops in subroutine `kirchslow()` may be interchanged at will without changing the result. To emphasize this flexibility, the loops are set at the same indentation level. We tend to think of fixed values of the outer two loops and then describe what happens on the inner loop. For example, if the outer two loops are those of the model space `modl(iz,ix)`, then for `adj=1` the program sums data along the hyperbola into the "fixed" point of model space. When loops are reordered, we think differently and opportunities arise for speed improvements.

1.2.2 Fast Kirchhoff code

Subroutine `kirchslow()` can easily be speeded by a factor that is commonly more than 30. The philosopy of this book is to avoid minor optimizations, but a factor of 30 really is significant, and the analysis required for the speed up is also interesting. Much of the inefficiency of `kirchslow()` arises when $x_{\max} \gg vt_{\max}$ because then many values of t are computed beyond $t_{\max}$. To avoid this, we notice that for fixed offset (`ix-iy`) and variable depth `iz`, as depth increases, time `it` eventually goes beyond the bottom of the mesh and, as soon as this happens, it will continue to happen for all larger values of `iz`. Thus we can `break` out of the `iz` loop the first time we go off the mesh to avoid computing anything beyond as shown in subroutine `kirchfast()`. (Some quality compromises, limiting the aperature or the dip, also yield speedup, but we avoid those.) Another big speedup arises from reusing square roots. Since the square root depends only on offset and depth, once computed it can be used for all `ix`. Finally, these changes of variables have left us with more complicated side boundaries, but once we work these out, the inner loops can be devoid of tests and in `kirchfast()` they are in a form that is highly optimizable by many compilers.

```
# Kirchhoff migration and diffraction.  (greased lightning)
#
subroutine kirchfast( adj, add, vrms,     t0,dt,dx, modl,nt,nx, data)
integer ix,iz,it,ib,  adj, add,                          nt,nx
real    amp,t,z,b,              vrms(nt), t0,dt,dx, modl(nt,nx),data(nt,nx)
call adjnull(         adj, add,                    modl,nt*nx, data,nt*nx)
do ib= -nx, nx {        b = dx * ib                      # b = offset
        do iz= 2, nt {  z = t0 + dt * (iz-1)             # z = travel-time depth
                        t = sqrt( z**2  +  (b*2/vrms(iz))**2 )
                        it = 1.5 + (t - t0) / dt
             if( it > nt )  break
                amp = (z / t)  *  sqrt( nt*dt / t )
                do ix= max0(1, 1-ib),  min0(nx, nx-ib)
                        if( adj == 0 )
                                data(it,ix+ib)=data(it,ix+ib)+modl(iz,ix   )*amp
                        else
                                modl(iz,ix   )=modl(iz,ix   )+data(it,ix+ib)*amp
                }
        }
return; end
```

Originally the two Kirchhoff programs produced identical output, but finally I could not resist adding an important feature to the fast program, scale factors $z/t = \cos\theta$ and $1/\sqrt{t}$ that are described elsewhere. The fast program allows for velocity variation with depth. When velocity varies laterally the story becomes much more complicated.

Figure 1.6 shows an example. The model includes dipping beds, syncline, anticline, fault, unconformity, and buried focus. The result is as expected with a "bow tie" at the buried focus. On a video screen, I can see hyperbolic events originating from the unconformity and the fault. At the right edge are a few faint edge artifacts. We could have reduced or eliminated these edge artifacts if we had extended the model to the sides with some empty space.

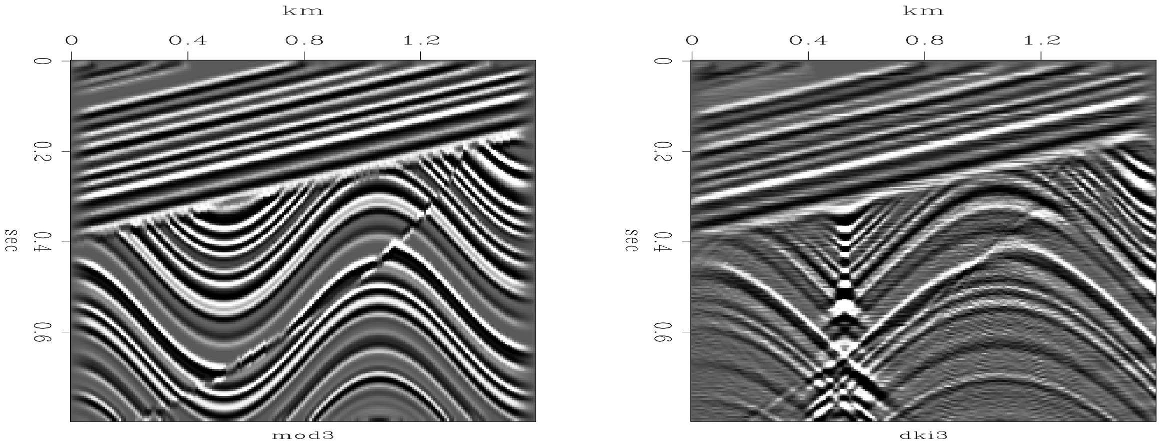

Figure 1.6: Left is the model. Right is diffraction to synthetic data. krch-kfgood [ER,M]

1.2.3 Kirchhoff artifacts

Reconstructing the earth model with the adjoint option in `kirchfast()` on the facing page yields the result in Figure 1.7. The reconstruction generally succeeds but is imperfect in a

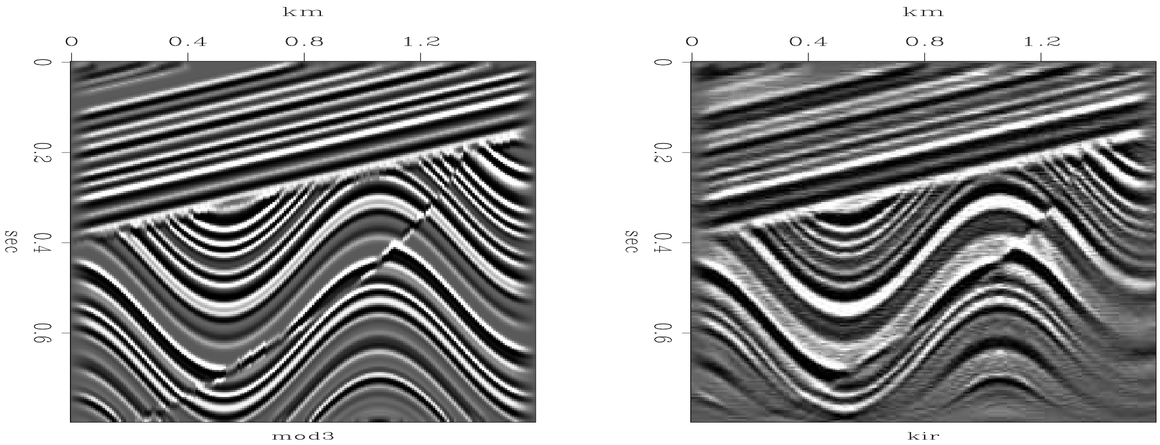

Figure 1.7: Left is the original model. Right is the reconstruction. krch-skmig [ER,M]

number of interesting ways. Near the bottom and right side, the reconstruction fades away, especially where the dips are steeper. Bottom fading results because in modeling the data we abandoned arrivals after a certain maximum time. Thus energy needed to reconstruct dipping beds near the bottom was abandoned. Likewise along the side we abandoned rays shooting off the frame.

Difficult migrations are well known for producing semicircular reflectors. Here we have controlled everything fairly well so none are obvious, but on a video screen I see some semicircles.

Next is the problem of the spectrum. Notice in Figure 1.7 that the reconstruction lacks the sharp crispness of the original. It is shown in chapter 6 that the spectrum of our reconstruction loses high frequencies by a scale of $1/|\omega|$. Philosophically, we can think of the hyperbola summation as integration, and integration boosts low frequencies. Figure 1.8 shows the average over x of the relevant spectra. First, notice the high frequencies are weak because there

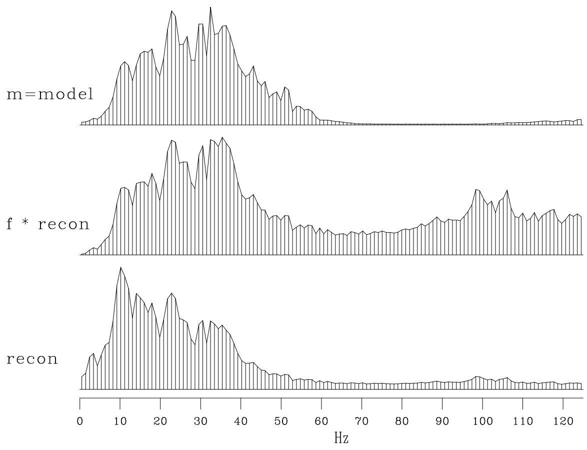

Figure 1.8: Top is the spectrum of the the model, i.e. the left side of Figure 1.7. Bottom is the spectrum of the the reconstruction, i.e. the right side of Figure 1.7. Middle is the reconstruction times frequency f. krch-kirspec [ER]

is little high frequency energy in the original model. Then notice that our cavalier approach to interpolation created more high frequency energy. Finally, notice that multiplying the spectrum of our migrated model by frequency, f, brought the important part of the spectral bands into agreement. This suggests applying an $|\omega|$ filter to our reconstruction, or $\sqrt{-i\omega}$ operator to both the modeling and the reconstruction, an idea implemented in subroutine `halfdifa()` on page 91.

Neither of these Kirchhoff codes addresses the issue of spatial **alias**ing. Spatial aliasing is a vexing issue of numerical analysis. The Kirchhoff codes shown here do not work as expected unless the space mesh size is suitably more refined than the time mesh. Figure 1.9 shows an example of forward modeling with an x mesh of 50 and 100 points. (Previous figures used 200 points on space. All use 200 mesh points on the time.) Subroutine `kirchfast()` on page 8 does interpolation by moving values to the nearest neighbor of the theoretical location. Had we taken the trouble to interpolate the two nearest points, our results would have been a little better, but the basic problem (resolved in chapter 10) would remain.

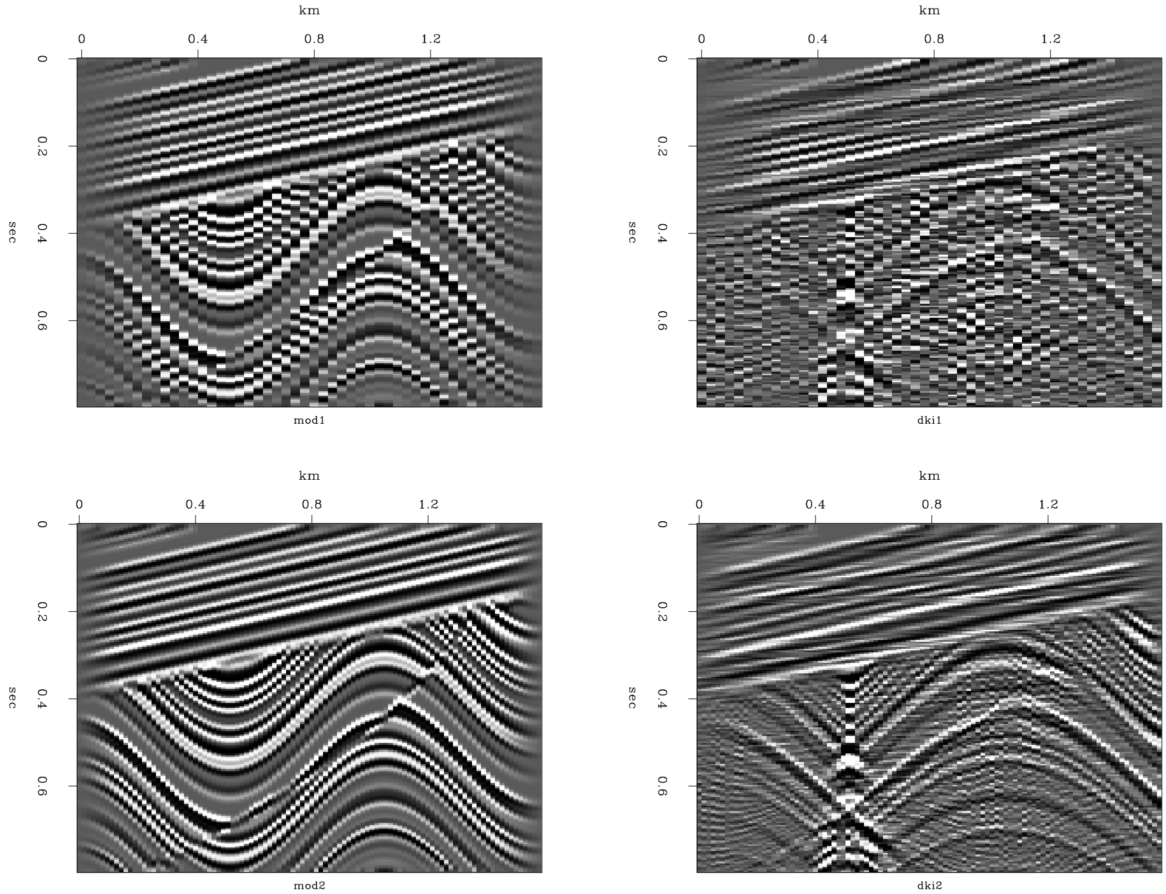

Figure 1.9: Left is model. Right is synthetic data from the model. Top has 50 points on the x-axis, bottom has 100. krch-skmod [ER]

1.2.4 Sampling and aliasing

Spatial ***alias****ing* means insufficient sampling of the data along the space axis. This difficulty is so universal, that all migration methods must consider it.

Data should be sampled at more than two points per wavelength. Otherwise the wave arrival direction becomes ambiguous. Figure 1.10 shows synthetic data that is sampled with insufficient density along the x-axis. You can see that the problem becomes more acute at high frequencies and steep dips.

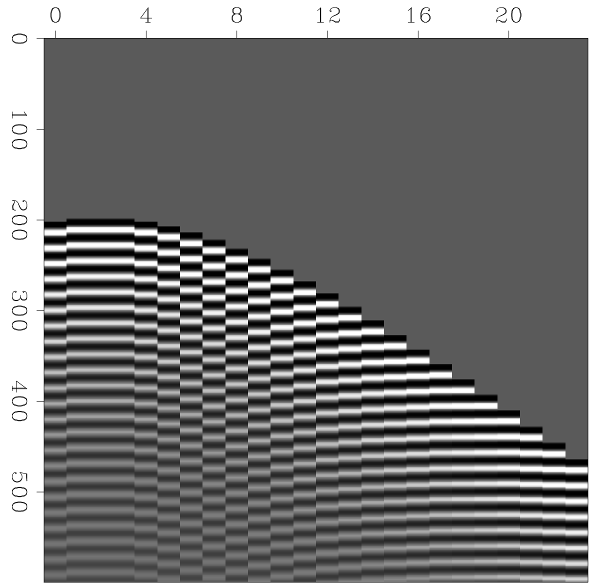

Figure 1.10: Insufficient spatial sampling of synthetic data. To better perceive the ambiguity of arrival angle, view the figures at a grazing angle from the side. krch-alias [ER]

There is no generally-accepted, automatic method for migrating spatially aliased data. In such cases, human beings may do better than machines, because of their skill in recognizing true slopes. When the data is adequately sampled however, computer migrations give better results than manual methods.

1.2.5 Kirchhoff migration of field data

Figure 1.11 shows migrated filed data.

The on-line movie behind the figure shows the migration before and after amplitude gain with time. You can get a bad result if you gain up the data, say with automatic gain or with t^2, for display before doing the migration. What happens is that the hyperbola flanks are then included incorrectly with too much strength.

The proper approach is to gain it first with $\sqrt{t}$ which converts it from 3-D wavefields to 2-D. Then migrate it with a 2-D migration like `kirchfast()`, and finally gain it further for display (because deep reflectors are usually weaker).

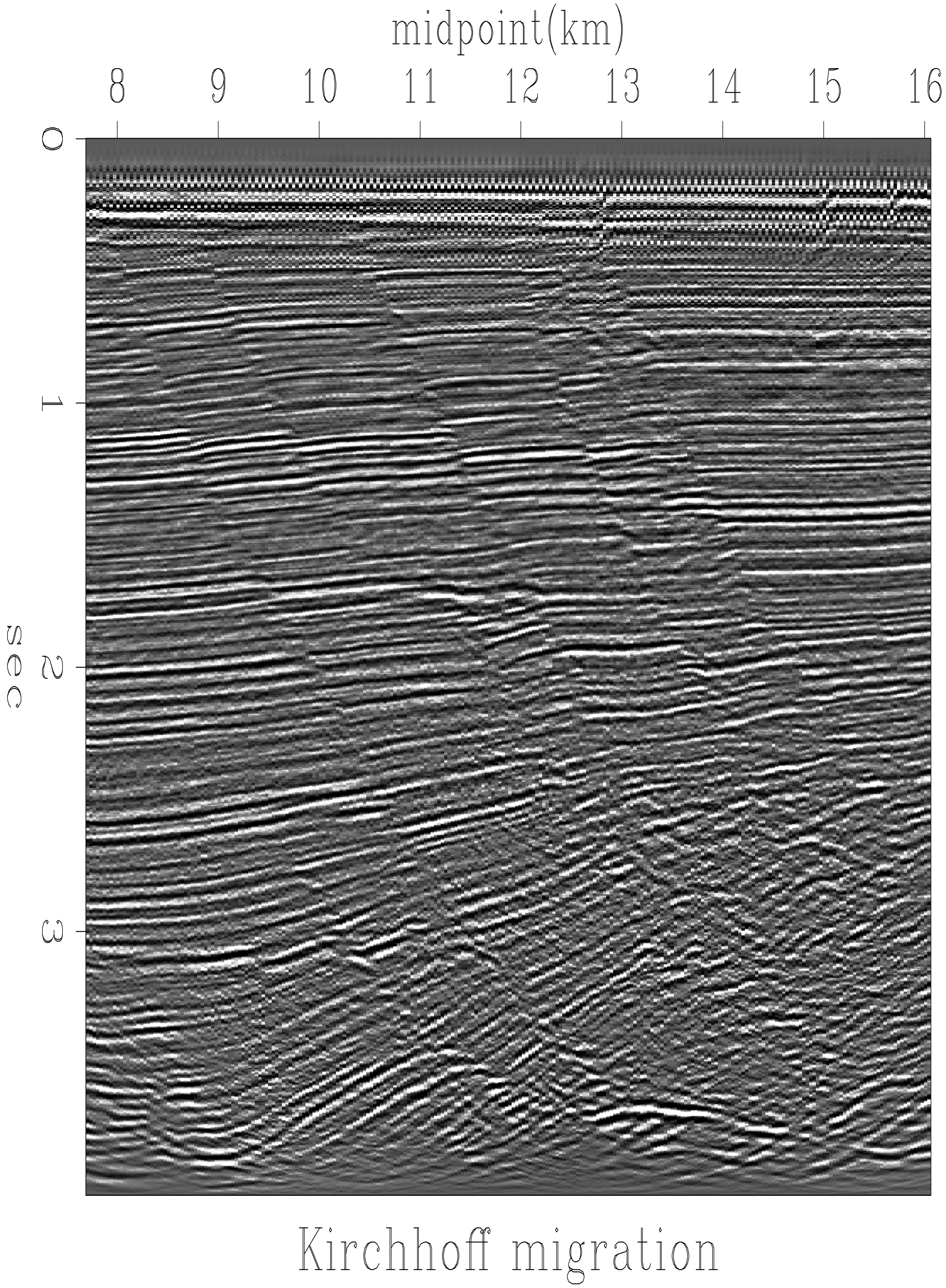

Figure 1.11: Kirchhoff migration of Figure **??**. Press button for movie comparing stack to migrated stack. krch-wgkirch [ER,M]

Index