Geoph 426 & 526
Signal Processing in Geophysics

M. D. Sacchi

Last revision: Winter 2022

Department of Physics
University of Alberta

Contact:

```
Dr M.D.Sacchi
Department of Physics,
University of Alberta,
Edmonton, Canada, AB, T6G 2J1

MSacchi@ualberta.ca
https://sites.ualberta.ca/~msacchi/
```

Contents

Chapter 1

Fourier Analysis

1.1 Introduction

We will first review fundamental aspects of Fourier Analysis, which you have already seen in a more theoretical manner in math classes. In particular, we will first study orthogonal expansions. We will start with the Fourier series, and then introduce the Fourier transform and its properties.

Along this course, we will deal with continuous and discrete signals. In this chapter, we explore the treatment of continuous signals. These are signals that depend on time t, which is considered a continuous variable. The extension to the discrete signals (time is discrete) is covered in Chapter 2.

1.1.1 Orthogonal Functions

We first present the approximation of a function (in general a time-dependent signal) in terms of a superposition of orthogonal functions.

A set of functions $\Phi_n(t)$, $n = 1, 2, 3, \ldots$ is said to be orthogonal in the interval $[t_1, t_2]$ if the following condition is satisfied

$$\int_{t_1}^{t_2} \phi_n(t)\phi_m(t)dt = k_m\delta_{m,n} \,, \tag{1.1}$$

where $\delta_{m,n}$ is the Kronecker operator given by

$$\delta_{m,n} = 0 \qquad \text{if} \qquad m \neq n$$
$$\delta_{m,n} = 1 \qquad \text{if} \qquad m = n\,.$$

In signal processing, one wants to represent a signal as a superposition of simple functions. For instance, as a superposition of sines or cosines. The convenience of this procedure will become clear as we learn about signal processing. In general, one can say that the representation should be in terms of functions with some attractive mathematical properties or with some physical meaning. For instance, oscillatory signals can be represented by the superposition of sines, cosines, or complex exponentials. Imagine the variation of the surface temperature with time at a given location on the earth; daily and seasonal variabilities can be represented by the sinusoidal functions of different periods.

Let assume that we would like to approximate a function $f(t)$ by a superposition of n orthogonal functions in the following way

$$f(t) \approx \sum_{n=1}^{N} c_n \phi_n(t)\,. \tag{1.2}$$

The coefficients $c_n\,, i = 1 \dots N$ can be obtained by minimizing the mean squared error (MSE) defined via the following expression

$$MSE = \frac{1}{t_2 - t_1}\int_{t_1}^{t_2} (f(t) - \sum_{n=1}^{N} c_n \phi_n(t))^2\, dt\,, \tag{1.3}$$

the last equation can be expanded as follows

$$MSE = \frac{1}{t_2 - t_1}\int_{t_1}^{t_2} (\, f(t)^2 + \sum_{n=1}^{N} c_n^2 \phi_n(t)^2 - 2\sum_{n=1}^{N} c_n \phi_n(t) f(t))\, dt\,. \tag{1.4}$$

I have omitted the cross-products of the form $\phi_n(t)\phi_m(t)$ because according to the definition given in equation 1.1 they cancel up. The last equation can be written as

$$MSE = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} f(t)^2 \, dt + \sum_{n=1}^{N} c_n^2 k_n - 2 \sum_{n=1}^{N} c_n \gamma_n \,, \tag{1.5}$$

where

$$\gamma_n = \int_{t_1}^{t_2} \phi_n(t) f(t) \, dt \,. \tag{1.6}$$

The term outside the integral in equation (1.5) can be rewritten as follows

$$\sum_{n=1}^{N} (c_n^2 k_n - 2 c_n \gamma_n) = \sum_{n=1}^{N} (c_n \sqrt{k_n} - \frac{\gamma_n}{\sqrt{k_n}})^2 - \sum_{n=1}^{N} \frac{\gamma_n^2}{k_n} \,.$$

We are now in the condition of rewriting the MSE via the following expression

$$MSE = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} f(t)^2 \, dt + \sum_{n=1}^{N} (c_n \sqrt{k_n} - \frac{\gamma_n}{\sqrt{k_n}})^2 - \sum_{n=1}^{N} \frac{\gamma_n^2}{k_n} \,. \tag{1.7}$$

It is clear that the MSE is minimum when the second term in the right-hand side of the last equation is zero

$$c_n \sqrt{(k_n)} = \frac{\gamma_n}{\sqrt{k_n}} \,, \tag{1.8}$$

or, in other words, the coefficient c_i is given by

$$c_n = \frac{\gamma_n}{k_n} = \frac{\int_{t_1}^{t_2} f(t) \phi_n(t) \, dt}{\int_{t_1}^{t_2} \phi(t)^2 dt} \,. \tag{1.9}$$

We have obtained an expression for the n-th coefficients of the expansion of $f(t)$. If the c_n , $n = 1 \ldots N$ are chosen according to the last expression, the mean squared error becomes

$$MSE = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} f(t)^2 \, dt - \sum_{n=1}^{N} c_n^2 k_n \,. \tag{1.10}$$

It can be shown that if $N \to \infty$ the mean squared error vanishes ($MSE \to 0$). In that case, the last expression becomes "Parseval's Theorem"

$$\frac{1}{t_2 - t_1} \int_{t_1}^{t_2} f(t)^2 dt = \sum_{n=1}^{\infty} c_n^2 k_n \,. \tag{1.11}$$

Parseval's theory basically tells us that the total "energy" of the signal is equal to the coefficients' total "energy."

1.1.2 Complex orthogonal functions

In our previous analysis we have considered orthogonal functions $\phi_n(t)$ that are real functions of the variable time. The representation of the signal $f(t)$ can also be in terms of orthogonal functions that are complex functions. If $\phi_n(t)$ are complex, expression 1.1 is given by

$$\int_{t_1}^{t_2} \phi_n(t)\phi_m^*(t)dt = k_m \delta_{m,n} \tag{1.12}$$

where * stands for the conjugate operator. In this case the coefficients required to represent $f(t)$ are given by

$$c_n = \frac{\gamma_n}{k_n} = \frac{\int_{t_1}^{t_2} f(t)\phi_n^*(t)\,dt}{\int_{t_1}^{t_2} |\phi(t)|^2 dt} \,. \tag{1.13}$$

1.1.3 Fourier Series

In this section we will represent periodic signal such as the one portrayed in Figure 1.1 Consider the orthogonal set given by complex exponentials also called harmonic functions

$$e^{i\, n\omega_0 t}\,, \qquad n = 0, \pm 1, \pm 2, \pm 3, \ldots \tag{1.14}$$

with the symbol $i = \sqrt{-1}$ indicating the unit imaginary number. It is easy to show that this set is orthogonal in $t \in [t_0, t_0 + \frac{2\pi}{\omega_0}]$. To prove the last statement we need to evaluate the following integral

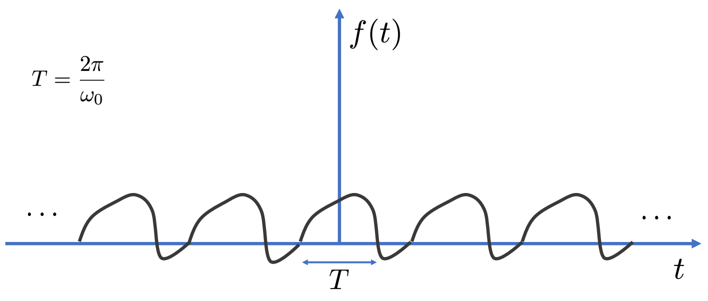


Figure 1.1: A periodic signal of period T can be represented via Fourier series.

$$\begin{aligned} I &= \int_{t_0}^{t_0+2\pi/\omega_0} \phi_n(t)\phi_k^*(t)\,dt \\ &= \int_{t_0}^{t_0+2\pi/\omega_0} e^{in\omega_0 t}e^{-ik\omega_0 t}\,dt \\ &= \tfrac{1}{i\omega_0(n-k)}e^{i(n-k)\omega_0 t_0}(e^{i2\pi(n-k)}-1)\,. \end{aligned} \tag{1.15}$$

It is easy to see that the integral takes the following values for any value of t_0

$$I = \begin{cases} 0 & \text{if} \quad n \neq k \\ \frac{2\pi}{\omega_0} & \text{if} \quad n = k \end{cases} \tag{1.16}$$

We have shown that the functions $e^{in\omega_0 t}\,, \quad n = 0, \pm1, \pm2\pm3, \dots$ are orthogonal functions. When representing oscillatory phenomena, complex exponentials are attractive functions because if you multiply two complex exponentials, the result is a new complex exponential. For instance, $e^{ix}.e^{iy} = e^{i(x+y)}$. I must point out that one could have used an expansion that uses *sin* or *cos* functions, but as you can imagine the algebra will become cumbersome. Practicality and simplicity make complex exponentials attractive functions to represent oscillatory phenomena.
When a signal is expanded in terms of exponentials we have a Fourier series

$$f(t) = \sum_{n=-\infty}^{\infty} F_n e^{in\omega_0 t} \,, \tag{1.17}$$

where the coefficients of the expansion are given by

$$F_n = \frac{\omega_0}{2\pi} \int_{t_0}^{t_0+2\pi/\omega_0} f(t) e^{-in\omega_0 t} \, dt \tag{1.18}$$

given that $T = \frac{2\pi}{\omega_0}$, F_n can also be written as follows

$$F_n = \frac{1}{T} \int_{t_0}^{t_0+T} f(t) e^{-in\omega_0 t} \, dt \,. \tag{1.19}$$

The coefficient F_n is the complex Fourier coefficient associated with the harmonic function of frequency $n\,\omega_0$. In this case, the signal $f(t)$ is considered to be periodic of period $T = 2\pi/\omega_0$. The periodic signal $f(t)$ has been decomposed into a superposition of complex exponentials of frequency $\omega_n = \omega_0 n$ and complex amplitude F_n. We usually refer to the plot of n or ω_n versus $|F_n|$ as the spectrum of $f(t)$, which is discrete because we have energy at discrete frequencies $n\omega_0$.

To analyze non-periodic signals, we need to introduce the Fourier Transform. In this case, the signal is represented in terms of a continuous spectrum of frequencies.

1.2 The Fourier Transform

So far we have found an expression that allows us to represent a periodic signal of period $T = 2\pi/\omega_0$ in terms of a superposition of elementary functions (complex exponentials). We have seen that the Fourier series can be used to represent a periodic or a non-periodic signal. We have to realize, however, that the Fourier series does not properly represent a non-periodic signal outside the interval of $[t_0, t_0 + T]$. In fact, outside $[t_0, t_0 + T]$ the Fourier series provides a periodic extension of $f(t)$.

We have also shown that a periodic signal has a discrete spectrum given by the coefficients of the expansion in terms of the Fourier series, which we have called $F_n, n = 0, \pm 1, \pm 2, \ldots$.

In this section, we will provide the representation for a non-periodic signal $f(t)$ in $t \in (-\infty, \infty)$ utilizing a continuous spectrum of frequencies. Let us assume that $f(t)$ is a periodic signal in the interval $[-T/2, T/2]$; we have learned that a periodic signal can be represented by a Fourier series as follows

$$f(t) = \sum_{n=-\infty}^{\infty} F_n e^{i\,n\omega_0 t} dt\,, \qquad \text{with} \qquad \omega_0 = \frac{2\pi}{T}\,, \tag{1.20}$$

where the coefficients are given by

$$F_n = \frac{1}{T}\int_{-T/2}^{T/2} f(t) e^{i\,n\omega_0 t} dt\,. \tag{1.21}$$

We can substitute equation (1.21) into (1.20) and obtain the following expression

$$f(t) = \sum_{n=-\infty}^{\infty} \frac{1}{T}\int_{-T/2}^{T/2} f(t) e^{-in\omega_0 t} dt e^{in\omega_0 t}\,. \tag{1.22}$$

Now we can make $T \to \infty$[1], we will also assume that the fundamental frequency $\omega_0 \to d\omega$, where $d\omega$ is a differential frequency. In this case, we can transform the discrete variable $n\omega_0$ into a continuous one ω, and finally, since now we have a summation on a continuous variable ω we will convert the sum $\sum$ into an integral $\int$

$$f(t) = \int_{-\infty}^{\infty} \frac{d\omega}{2\pi} (\int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt) e^{j\omega t} d\omega\,. \tag{1.23}$$

The integral in brackets is called the Fourier transform of $f(t)$:

[1]We do this to extend our periodic signal into a non-periodic one

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt\,. \tag{1.24}$$

It is clear from equation (1.23) that the formula to represent the signal in terms of $F(\omega)$ is now given by

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{i\omega t} d\omega\,. \tag{1.25}$$

The pair (1.24) and (1.25) are used to compute the Fourier transform and its inverse, respectively. Equation (1.25) is also refereed as the inverse Fourier transform.[2]

It is important to stress that the signal in $(-\infty, \infty)$ now has a continuous spectrum of frequencies. The Fourier transform is, in general, a complex function that can be written as follows

$$F(\omega) = |F(\omega)| e^{i\theta(\omega)} \tag{1.26}$$

where $|F(\omega)|$ is the amplitude spectrum and $\theta(\omega)$ is the phase spectrum. We will come back to the importance of amplitude and phase when dealing with seismic signals.

1.2.1 Properties of the Fourier Transform

We will not prove these properties here. Most of them can be easily demonstrated by using the definition of the Fourier Transform (several demonstrations will be part of an assignment).

We shall use the following notation to indicate that $F(\omega)$ is the Fourier Transform of $f(t)$

$$f(t) \leftrightarrow F(\omega)$$

[2]In fact, one can think that equation (1.24) is a *forward transform* or a transform *to go* to a new domain (the frequency domain), whereas equation (1.25) is an *inverse transform* or a transform *to come back* to the original domain (time) from the frequency domain.

Symmetry.

$$F(t) \leftrightarrow 2\pi f(-\omega)$$

Linearity. If

$$f_1(t) \leftrightarrow F_1(\omega)$$
$$f_2(t) \leftrightarrow F_2(\omega)$$

then

$$f_1(t) + f_2(t) \leftrightarrow F_1(\omega) + F_2(\omega)$$

Scale. If

$$f(at) \leftrightarrow \frac{1}{|a|} F(\frac{\omega}{a})$$

Convolution. If

$$f_1(t) \leftrightarrow F_1(\omega)$$
$$f_2(t) \leftrightarrow F_2(\omega)$$

then

$$\int_{-\infty}^{\infty} f_1(u) f_2(t-u) du \leftrightarrow F_1(\omega) F_2(\omega)$$

or in a few words:*time convolution* $\leftrightarrow$ *frequency multiplication.*[3]
Convolution in frequency. Similar to the previous one, but now

$$f_1(t).f_2(t) \leftrightarrow \frac{1}{2\pi} \int_{-\infty}^{\infty} F_1(v) F_2(\omega - v) dv$$

or in different words, *time multiplication* $\leftrightarrow$ *frequency convolution.*[4]

[3]This is a very important property and we will make extensive use of it. Most physical systems can be described as linear and time invariant systems, this leads to a convolution integral.

[4]We will use this property to estimate the Fourier Transform of a signal that has been recorded in a finite temporal window.

Time delay. I like this one, we can use it to delay or advance a signal in time:

$$f(t-\tau) \leftrightarrow F(\omega)e^{-i\omega t_0}$$

Modulation. This property is used by AM radios where a low frequency signal $f(t)$ is is multiplied by a carrier signal of frequency ω_0 and the spectrum of $f(t)$ is shifted to higher frequencies:

$$f(t)e^{j\omega_0 t} \leftrightarrow F(\omega-\omega_0)$$

Time derivatives. This is used to compute derivatives (actually, using the discrete Fourier transform which we have not seen at this point of the course):

$$\frac{df(t)}{dt} \leftrightarrow i\omega F(\omega)$$

It is clear that to take the derivative of $f(t)$ is equivalent to amplify the high frequencies of $f(t)$.
The property can be extended to derivatives of order n:

$$\frac{d^n f(t)}{dt^n} \leftrightarrow (i\omega)^n F(\omega)$$

or even to compute fractional derivatives. If a is not an integer, one can define a fractional derivative as follows

$$\frac{d^a f(t)}{dt^a} \leftrightarrow (i\omega)^a F(\omega)$$

.

1.2.2 The Fourier Transform of some signals

$\boxed{\text{A Boxcar}}$

We will compute the Fourier Transform of the following function which is called a boxcar

$$f(t) = \begin{cases} 1 & |t| < T/2 \\ 0 & otherwise \end{cases} \tag{1.27}$$

We substitute $f(t)$ into the definition of the Fourier Transform (equation (1.24)) and solve the integral

$$\begin{aligned} F(\omega) &= \int_{-T/2}^{T/2} 1.e^{-i\omega t} dt \\ &= \tfrac{1}{-i\omega}(e^{-i\omega T/2} - e^{j\omega T/2}) \\ &= T sinc(\omega T/2)\,, \end{aligned} \tag{1.28}$$

where in last equation $sinc(x) = sin(x)/x$. The Fourier Transform of the boxcar function is a *sinc* function. We will come back to the importance of the knowing the Fourier Transform of the box car function when dealing with the spectrum of signal that have been truncated in time.

In Figures (1.2.2) and (1.2.2), I have displayed the Fourier transform of two boxcar functions of width $T = 10$ and 20 s, respectively. Notice that the width of $F(\omega)$ increases when the width of $f(t)$ decreases.

Delta function:

We will compute the Fourier Transform of the delta function

$$f(t) = \delta(t)\,.$$

The δ function is defined according to

$$\int g(u)\delta(u)du = g(0)\,.$$

It easy to see from the above definition that the Fourier Transform of the delta function is given by

$$F(\omega) = \int_{-\infty}^{\infty} \delta(t) e^{-i\omega t} dt = 1$$

Therefore, we have

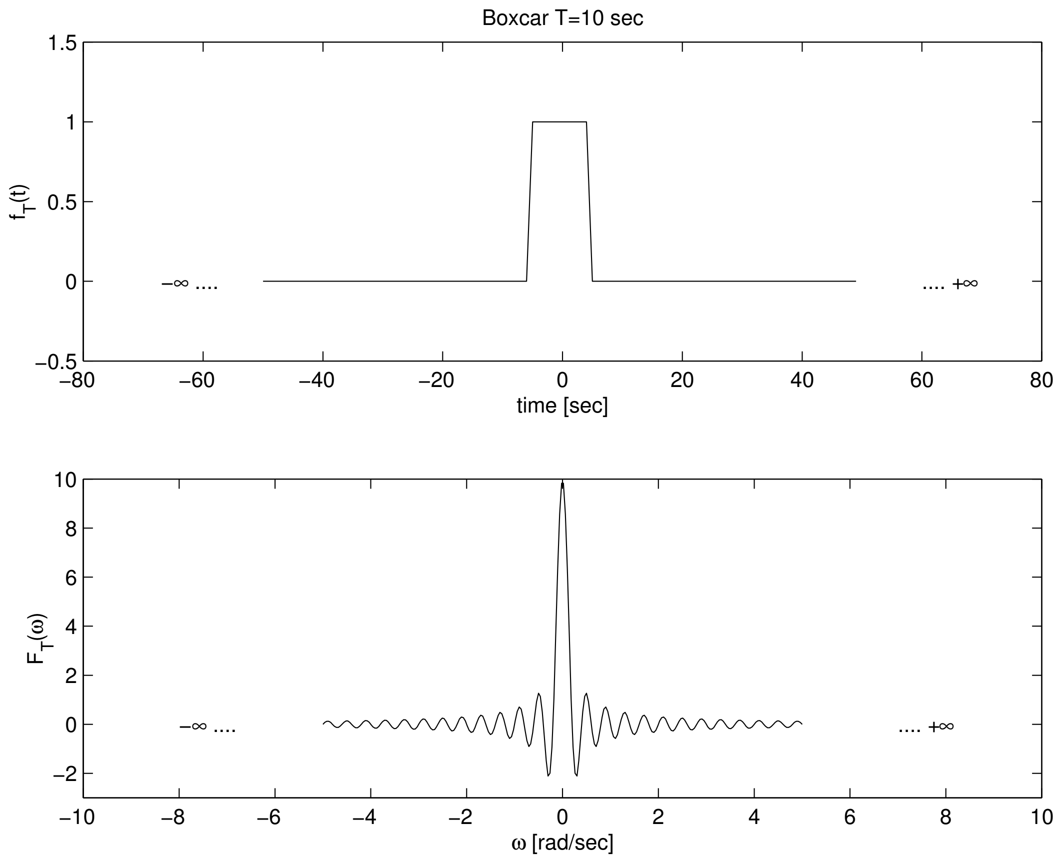


Figure 1.2: The Fourier transform of a boxcar function of width $T = 10\,\text{s}$.

$$\delta(t) \leftrightarrow F(\omega) = 1 \forall \omega$$

Similarly, if we apply the "time delay" property, one can compute the Fourier Transform of a delayed delta function

$$\delta(t-\tau) \leftrightarrow 1\, e^{-i\omega\tau}\,.$$

The delayed delta function has amplitude $|F(\omega)| = 1$ and phase $\theta(\omega) = -\omega\tau$. This phase is often called a linear phase because it depends linearly on ω.

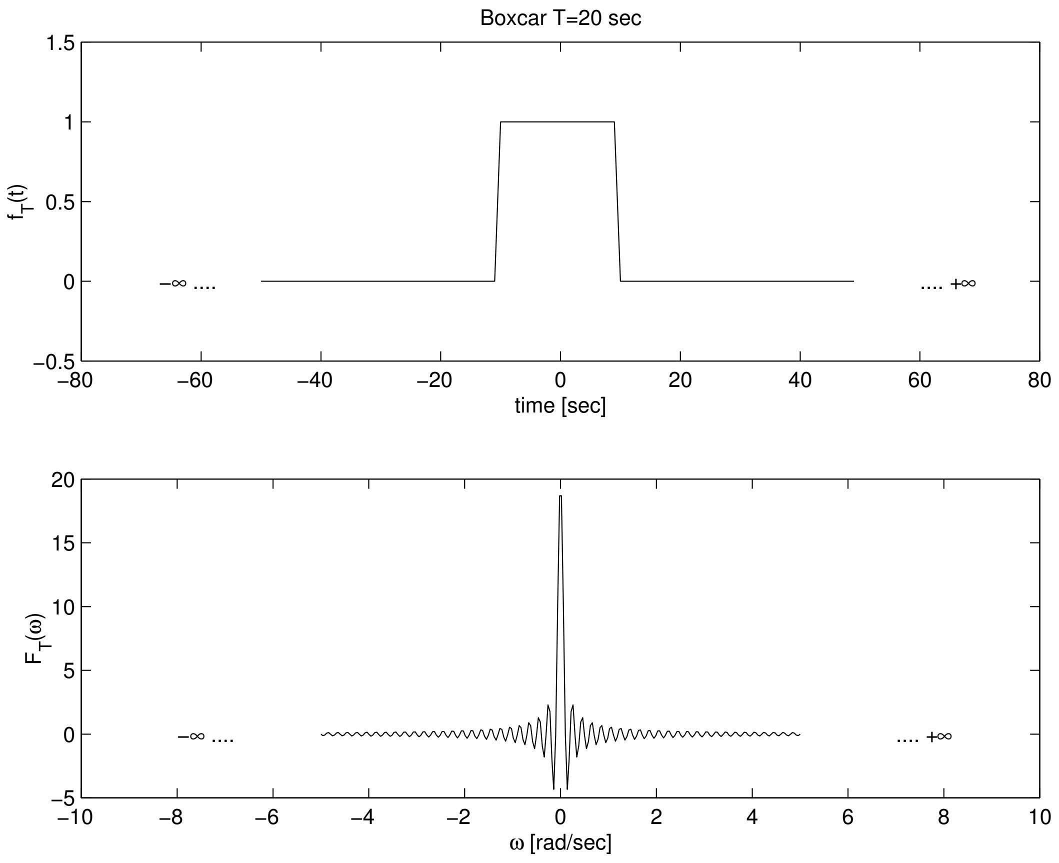


Figure 1.3: The Fourier transform of a boxcar function of width $T = 20\,\mathrm{s}$.

It is clear that the δ function has a continuous amplitude spectrum that contains all frequencies. The delta function is also the ideal seismic wavelet that one would like to have in seismic exploration. Clearly, one cannot physically designed a delta function as it will require a device that can produce frequencies in the range $-\infty < \omega < \infty$.

$\boxed{\text{A complex exponential:}}$

We will compute the Fourier transform of the complex exponential of frequency ω_0

$$f(t) = e^{i\omega_0 t} \qquad -\infty < t < \infty \tag{1.29}$$

We can combine the Fourier Transform of the delta function with the symmetry property to obtain the Fourier Transform of a complex exponential. Using the Fourier Transform of the delayed delta function

$$\delta(t-\tau) \leftrightarrow 1e^{-i\omega\tau}$$

and after applying the symmetry property leads to

$$F(t) \leftrightarrow 2\pi f(-\omega)\,.$$

from where we obtain

$$e^{i\omega_0 t} \leftrightarrow 2\pi\delta(\omega-\omega_0)$$

In other words, the FT of complex sinusoid of frequency ω_0 is a delta at the corresponding frequency $\omega = \omega_0$. The above can be used to compute the Fourier transform of $cos(\omega_0 t)$ and $sin(\omega_0 t)$.

1.2.3 Truncation in time

Given $f(t)$ $t \in (-\infty, \infty)$, with $f(t) \leftrightarrow F(\omega)$, how do we obtain the FT of the signal when the signal is recorded in a finite interval $t \in [-T/2, T/2]$?

First, we call $f_T(t)$ the observed signal in $t \in [-T/2, T/2]$ and $f(t)$ the *ideal* signal in $t \in (-\infty, \infty)$. The signals $f_T(t)$ and $f(t)$ are related via a truncation operator $b_T(t)$

$$f_T(t) = f(t)\,.b_T(t) \tag{1.30}$$

where $b_T(t)$ is a box function like the one already analyzed (see equations 1.27 and 1.28). Using the frequency convolution theorem,

$$f_T(t) = f(t).b_T(t) \leftrightarrow \frac{1}{2\pi}\int_{-\infty}^{\infty} F(v)B_T(\omega-v)dv\,,$$

we can write

$$F_T(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(v)B_T(\omega - v)dv = \frac{1}{2\pi}F(\omega) * B_T(\omega)\,, \tag{1.31}$$

where $B_T(\omega) = Tsinc(\omega T/2)$. This is remarkably interesting result (it is?). We are saying that our observation window is affecting the Fourier Transform of the signal. We want to know $F(\omega)$ but since we are recording the signal in a finite interval, we have only access to $F_T(\omega)$. The latter is a distorted version of $F(\omega)$ which is given by

$$F_T(\omega) = \frac{1}{2\pi}T\int_{-\infty}^{\infty} F(u)sinc((\omega - u)T/2)du\,. \tag{1.32}$$

It is clear from the above that one does not see $F(\omega)$ but its convolution with a *sinc* function.

Let us consider a simple example where $f(t) = e^{i\omega_0 t}$ for $-\infty < t < \infty$ and $f_T(t) = f(t).b^T(t)$. The Fourier Transform of $f_T(t)$ is given by

$$F_T(\omega) = \frac{1}{2\pi}T\int_{-\infty}^{\infty} 2\pi\delta(\omega - \omega_0)sinc((\omega - u)T/2)du \tag{1.33}$$

which leads to

$$F_T(\omega) = Tsinc((\omega - \omega_0)T/2)\,. \tag{1.34}$$

Expression 1.34 is a *sinc* function with a peak at $\omega = \omega_0$. In Figure (1.2.3) we portray the superposition of two complex sinusoids of the form

$$f_T(t) = e^{i\omega_1 t} + e^{i\omega_2 t}\,, \qquad t \in [-10, 10]\,\mathrm{s}\,.$$

The Fourier transform of the two complex exponents if measured in an infinity time interval is given by two delta functions at frequencies ω_1 and ω_2. But since we are observing the signal in a finite length interval, the *ideal* Fourier Transform of $f(t)$ is convolved with $B_T(\omega)$ (The Fourier Transform of the boxcar function). For this example, I have chosen the following frequencies $\omega_1 = 0.5\,\mathrm{rad/sec}$ and $\omega_2 = 1.\,\mathrm{rad/sec}$.

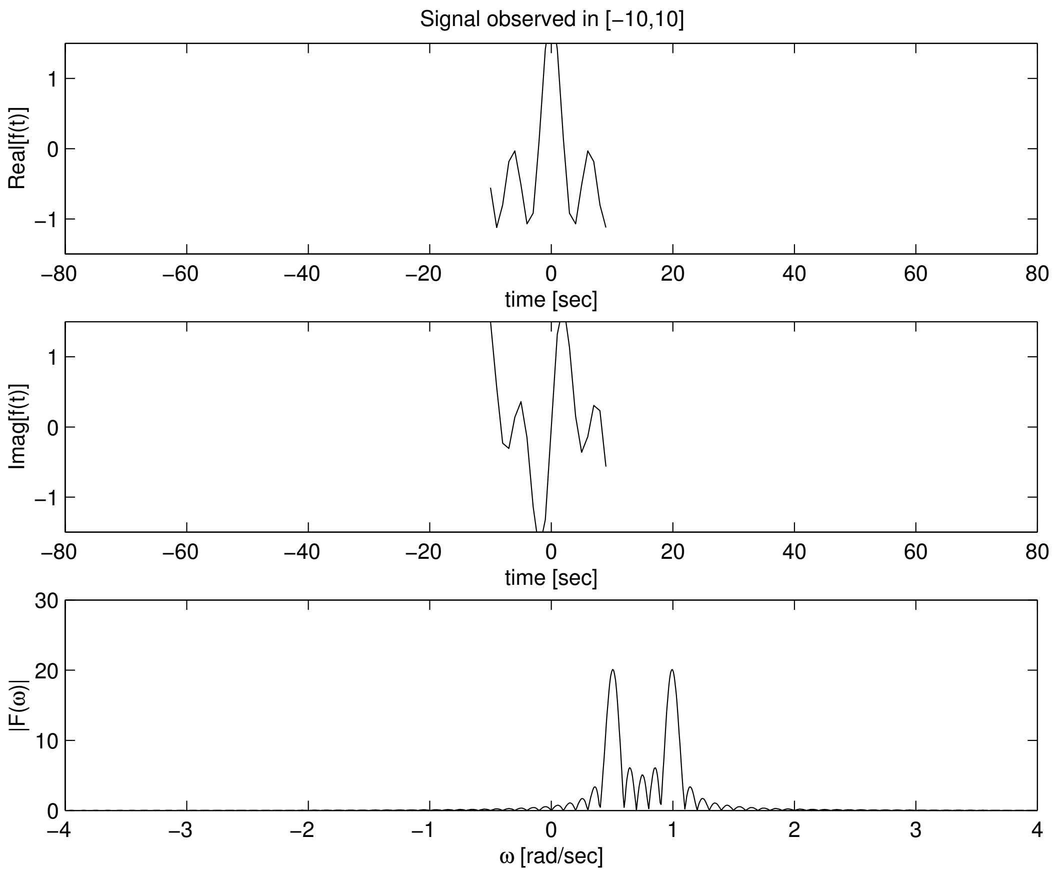


Figure 1.4: The Fourier transform of a the superposition of two complex exponentials observed in a window of length $T = 20$s. Up: real part of the signal. Centre: Imaginary part of the signal. Bottom: Amplitude spectrum of the Fourier Transform ($|F(\omega)|$).

1.3 Symmetries of the $F(\omega)$ for real signals.

Before continuing with the Fourier transform and its applications a few words about the symmetries of $F(\omega)$ are needed. This is very important in the discrete case because it allows to write faster algorithms for signal processing.

Let us start with the definition of the Fourier transform,

$$F(\omega) = \int f(t) e^{-i\omega t} dt \,. \tag{1.35}$$

If the signal $f(t)$ is a real signal, we can write

$$F(\omega) = R(\omega) + iG(\omega) \tag{1.36}$$

where

$$R(\omega) = \int f(t) cos(\omega t) dt \tag{1.37}$$

and

$$G(\omega) = -\int f(t) sin(\omega t) dt \,. \tag{1.38}$$

Since *cos* is an even function and *sin* an odd function

$$R(\omega) = R(-\omega) \,, \tag{1.39}$$

and

$$G(\omega) = -G(-\omega) \,. \tag{1.40}$$

If you know $F(\omega)$ for $\omega \geq 0$, you can compute $F(\omega)$ for $\omega < 0$ by applying the above identities.

In fact, we can always write

$$F(\omega) = R(\omega) + iG(\omega) \tag{1.41}$$

and

$$F(-\omega) = R(-\omega) + iG(-\omega) \tag{1.42}$$

by combining last equation with equation (1.40) we obtain

$$F(-\omega) = R(\omega) - iG(\omega) \,. \tag{1.43}$$

The last equation can be used to compute the negative semi-axis of the Fourier transform. This property is often referred as the Hermitian symmetry of the Fourier Transform. You can also write the latter as follows

$$F(-\omega) = F(\omega)^*$$

where the $*$ is used to denote complex conjugate. This property is only valid for real time series. The symmetry of the Fourier Transform explains why we often plot one semi-axis (in general the positive one) when displaying the Fourier amplitude or phase spectrum of a real signal.

The symmetry of the real and imaginary parts of the Fourier transform can also be used to obtain the symmetry of the amplitude and phase of the Fourier transform:

$$F(\omega) = |F(\omega)|e^{i\theta(\omega)}\,.$$

It is east to prove that the amplitude is an even function

$$|F(\omega)| = |F(-\omega)|\,. \tag{1.44}$$

Similarly, the phase is an odd function

$$\theta(\omega) = -\theta(-\omega)\,. \tag{1.45}$$

I stress again that the symmetries discussed in this section are valid for signals that are real functions.

1.4 References

Gabel R. and Roberts R., 1991, Signal and Linear Systems, Wiley; 3rd edition

Chapter 2

Discrete Signals

So far, we have described the Fourier transform of a continuous (analog) signal. Now, we will start to study discrete signals, which are also called time series. I will provide the connection between the continuous and the discrete world, which is essential because it will lead to numerical implementations to utilize field data.

2.1 Nyquist-Shannon sampling theorem

We will designate $f(t)$ the analog signal and $f_s(t)$ the associated discrete signal. One can think that f_s is obtained by sampling $f(t)$ every Δt seconds

$$f_s(t) = f(t) \sum_{k=-\infty}^{\infty} \delta(t - k\Delta t)\,. \tag{2.1}$$

By the frequency convolution property we can obtain the Fourier Transform of the sampled signal

$$F_s(\omega) = \frac{1}{2\pi} F(\omega) * \omega_0 \sum_{k=-\infty}^{\infty} \delta(\omega - k\omega_0)\,, \qquad \omega_0 = \frac{2\pi}{\Delta T} \tag{2.2}$$

where in the last equation, I have assumed that we know how to compute the Fourier Transform of the sampling operator $\sum_{k=-\infty}^{\infty}\delta(t-k\Delta t)$ (Papoulis, 1962). After a few mathematical manipulations, it is easy to see that

$$F_s(\omega)=\frac{1}{\Delta T}\sum_{k=-\infty}^{\infty}F(\omega-n\omega_0\,. \tag{2.3}$$

One can observe that the Fourier Transform of the sampled signal is a periodic function with period ω_0. If one wants to compute $F_s(\omega)$ in such a way that $F(\omega)$ can be completely recovered, the signal $f(t)$ must be a band-limited signal. This is a signal where the spectral components outside the interval $[-\omega_{max},\omega_{max}]$ are zero. When the following condition is satisfied

$$\omega_0 \geq 2\omega_{max}$$

there is no overlap of spectral contributions and therefore $F_s(\omega)\,,\ \omega\in[-w_{wmax},w_max]$ is equivalent, within a scale factor $1/(\Delta T)$, to the Fourier Transform of the analog signal $F(\omega)$. The last condition can be written as follows:

$$\frac{2\pi}{\Delta T}\geq 2\times 2\pi f_{max}$$

which reduces to

$$\Delta T \leq \frac{1}{2f_{max}}\,.$$

The last equation is also designated as the Sampling theorem or Nyquist–Shannon sampling theorem. It basically tells us that to recover the Fourier Transform of the original signal we need to sample the data according to the last inequality.

Real-world signals are continuous and become discrete after going trough acquisition systems (i.e., digital seismograph). To avoid alias, an analog filter is usually placed in the acquisition system before sampling. The data are first band-limited using analog filters, then sampled and finally, stored digitally.

The aliasing effect is described in Figures (2.2)-(2.5). Figure (2.2) corresponds to the Fourier transform of a continuous signal. We can observe

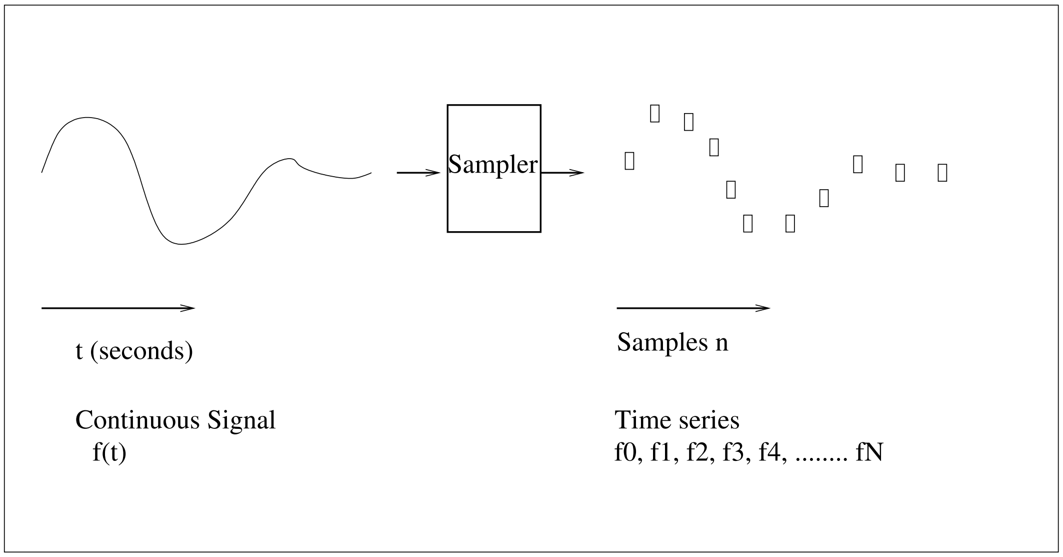


Figure 2.1: Discretization of a continuous signal.

that to properly recover the Fourier transform of the continuous signal, we need to sample our data according to $\omega_0 \leq 2\omega_{max}$. This is true for Figures (2.3) and (2.4). In these two figures, it easy to see that the Fourier transform of the original (continuous) signal is precisely represented by the Fourier transform of the discretized signal in the interval $[-\omega_{max}, \omega_{max}]$. In Figure (2.5), we portray an example where the data has been under-sampled and, therefore, the Fourier transform of the continuous signal cannot be recovered from the Fourier transform of the discretized signal (the signal is said to be aliased).

2.2 References

Papoulis A., 1962, Fourier Integral and Its Applications, McGraw-Hill.

Oppenheim A., and Schafer R.W., 1975, Digital Signal Processing, Prentice-Hall, Inc.,

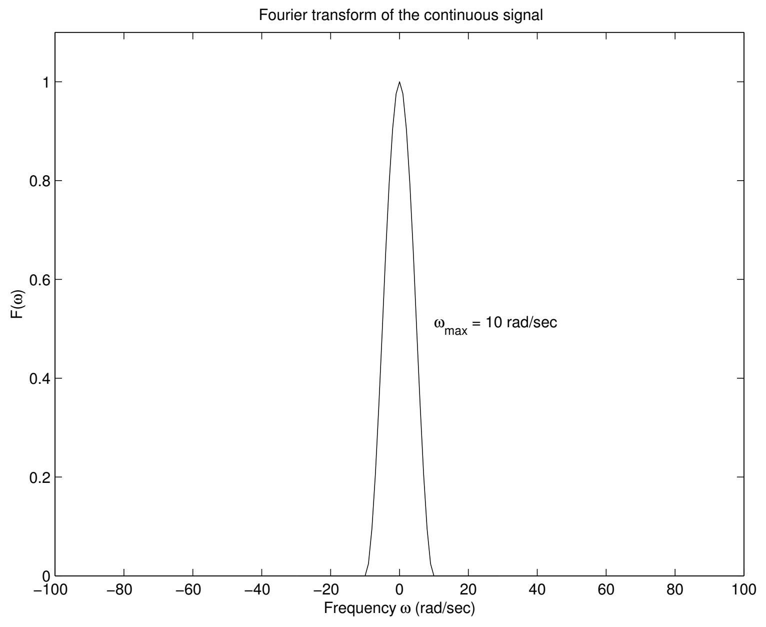


Figure 2.2: The Fourier Transform of a continuous signal.

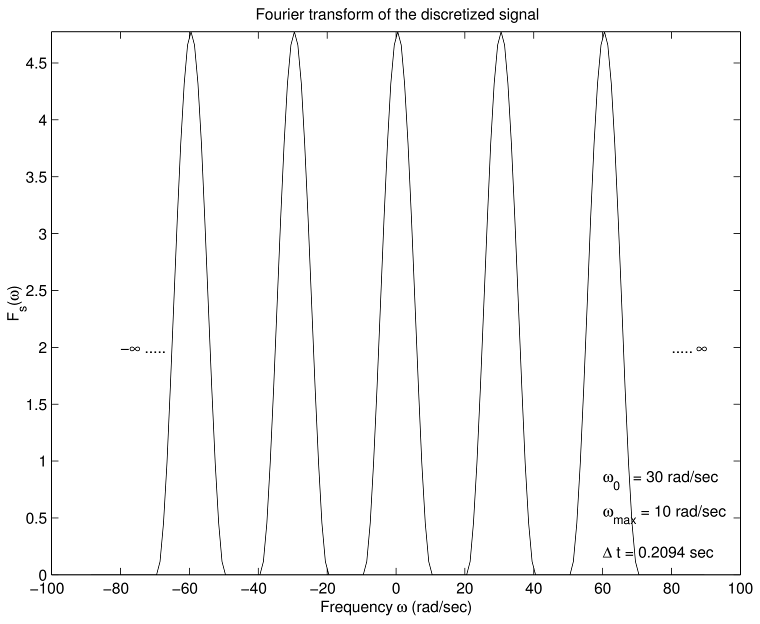


Figure 2.3: The Fourier Transform the continuous signal after being discretized, in this case $\omega_{max} = 10$ and $\omega_0 = 30$

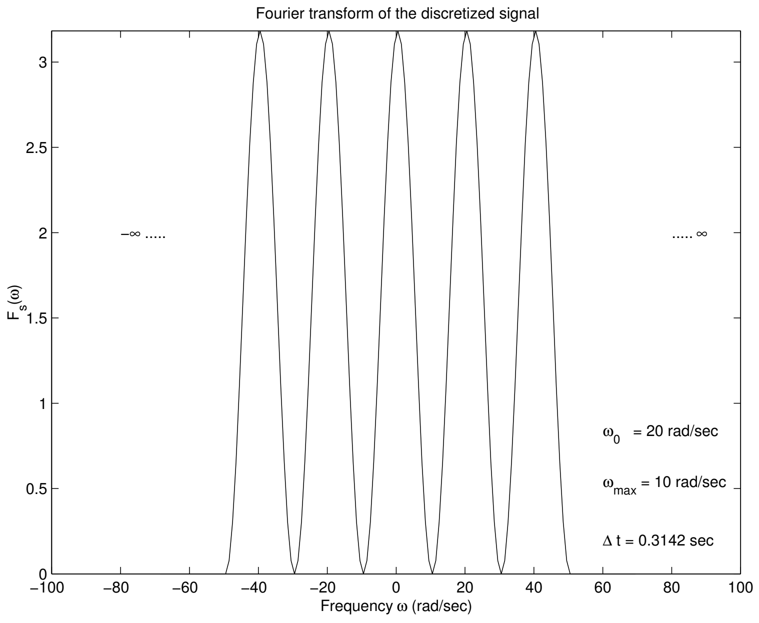


Figure 2.4: The Fourier transform the continuous signal after being discretized, in this case $\omega_{max} = 10$ and $\omega_0 = 20$. The Fourier transform of the continuous signal is perfectly represented in the interval $[-\omega_{max}, \omega_{max}]$.

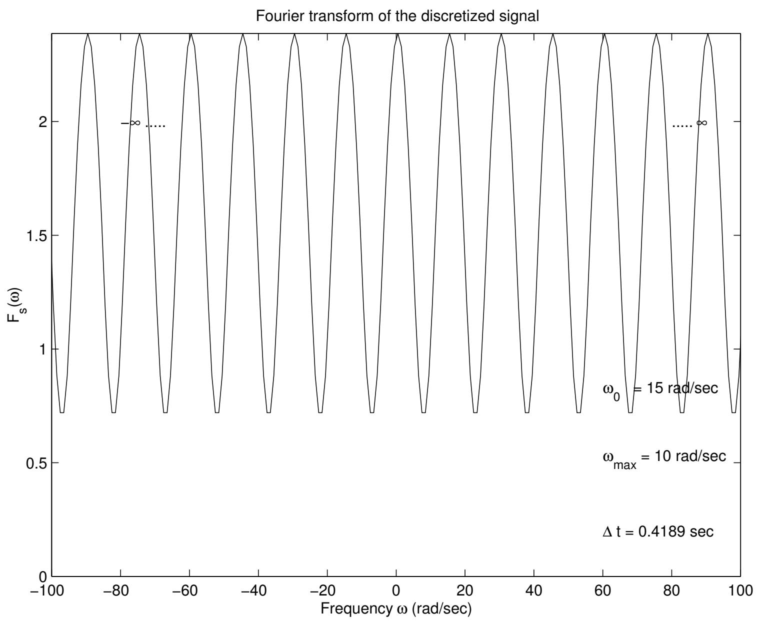


Figure 2.5: The Fourier transform the continuous signal after being discretized, in this case $\omega_{max} = 10$ and $\omega_0 = 15$. The signal is aliased. Note that the Nyquist–Shannon theorem is not satisfied. The Fourier Transform of the continuous signal cannot be recovered from the Fourier Transform of the sampled signal.

Chapter 3

Linear systems, the z transform, and convolution

This chapter will discuss liner systems (a simple way of describing a physical system). A continuous-time and time-invariant linear system will lead to the convolution integral and, in the discrete case, to the convolution sum. I will also present the z-transform, a tool to represent discrete series.

3.1 Linear systems for continuous signals

Linear systems are useful to define input-output relationships for continuous and discrete signals. Let us assume that we have a linear system where the input to the system is the continuous signal $x(t)$ and the output is given by $y(t)$

$$x(t) \to y(t) \,.$$

If the system is linear, the following properties must be satisfied

P1 : For any scalar α

$$\alpha x(t) \to \alpha y(t) \,.$$

P2: If

$$x_1(t) \to y_1(t)$$

and

$$x_2(t) \to y_2(t)$$

then

$$x_1(t) + x_2(t) \to y_1(t) + y_2(t)\,.$$

P3: Properties **P1** and **P2** can be combined into a single property

$$\alpha x_1(t) + \beta x_2(t) \to \alpha y_1(t) + \beta y_2(t)\,.$$

We will say that the linear system is time-invariant if and only if

$$x(t-T) \to y(t-T)\,,$$

and this is true for any arbitrary T. In other words, if the input signal is delayed by an amount of T, the output signal is delayed by the same amount.

We will represent our linear system as follows

$$\mathcal{H}[x(t)] = y(t)\,. \tag{3.1}$$

Where $\mathcal{H}$ represents the linear system in an operator form. If the system is linear, the function $\mathcal{H}$ has the following general expression

$$y(t) = \mathcal{H}[x(t)] = \int_{-\infty}^{\infty} h(t,\tau)x(\tau)d\tau\,. \tag{3.2}$$

It is easy to prove that the above expression defines a linear system (satisfies Property **P3**). When the system is linear and time-invariant the following property should also be satisfied

$$y(t-T) = \mathcal{H}[(x(t-T)]\,. \tag{3.3}$$

In this case, we need to rewrite equation 3.2 to satisfy the requirement mentioned above. For this purpose we write $h(t,\tau)$ as follows

$$h(t,\tau) = h(t-\tau)\,. \tag{3.4}$$

If we replace $h(t-\tau)$ in equation (3.2) we end up with the following expression

$$y(t) = \int_{-\infty}^{\infty} h(t-\tau)x(\tau)d\tau\,. \tag{3.5}$$

It is clear that the above equation defines a linear system, but it is not clear that it is also a time-invariant system. To prove that 3.5 corresponds to the input-output relationship of a time-invariant linear system we will apply the following change of variables

$$u = t - \tau\,.$$

Then,

$$y(t) = -\int_{\infty}^{-\infty} h(u)x(t-u)du = \int_{-\infty}^{\infty} h(u)x(t-u)du = H[x(t)]\,, \tag{3.6}$$

substituting t by $= t - T$

$$y(t-T) = \int_{\infty}^{\infty} h(u)x(t-T-u)du = H[x(t-T)]\,. \tag{3.7}$$

We have proved that the convolution integral defines a time-invariant linear system. Using the convolution theorem, "*convolution in the time domain $\longrightarrow$ multiplication in the frequency domain*", we can rewrite the convolution integral as follows

$$Y(\omega) = H(\omega)\,.X(\omega)\,.$$

The function $h(t)$ is also called the impulse response of the system. The Fourier transform of the impulse response, $H(\omega)$, is the system's transfer function. If the input to a system is given by $x(t) = \delta(t)$ the output is given by $y(t) = h(t)$. This statement can be easily proved by substituting $x(t) = \delta(t)$ into the convolution integral:

$$y(t) = \int_{-\infty}^{\infty} h(u)\delta(t-u)du = h(t)\,. \tag{3.8}$$

It turns out that if you do not know $h(t)$, it can be obtained by exciting the system with a δ function and measuring the output signal $y(t) = h(t)$ (Figure 3.3).

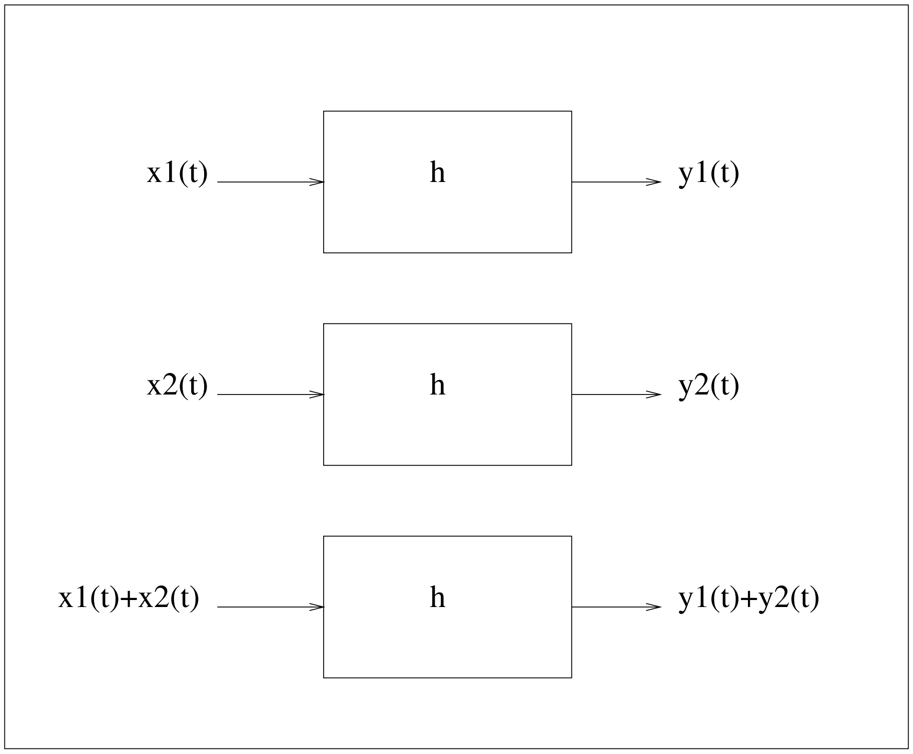


Figure 3.1: A linear system. The symbol h is the impulse response of the system.

3.1.1 Discrete convolution

When working with discrete signals, we define a linear system where a convolution sum describes the input-output relationship. In this case, we can turn the convolution integral into a sum

$$y_k = \sum_{n=-\infty}^{\infty} h_n x_{k-n} \,. \tag{3.9}$$

In general, we will be concerned with finite length signals. For instance, let us consider the convolution of the discrete signals x_n (input) and h_n (impulse reponse) that yields the new signal y_n (output) . The length of the signals mentioned above are

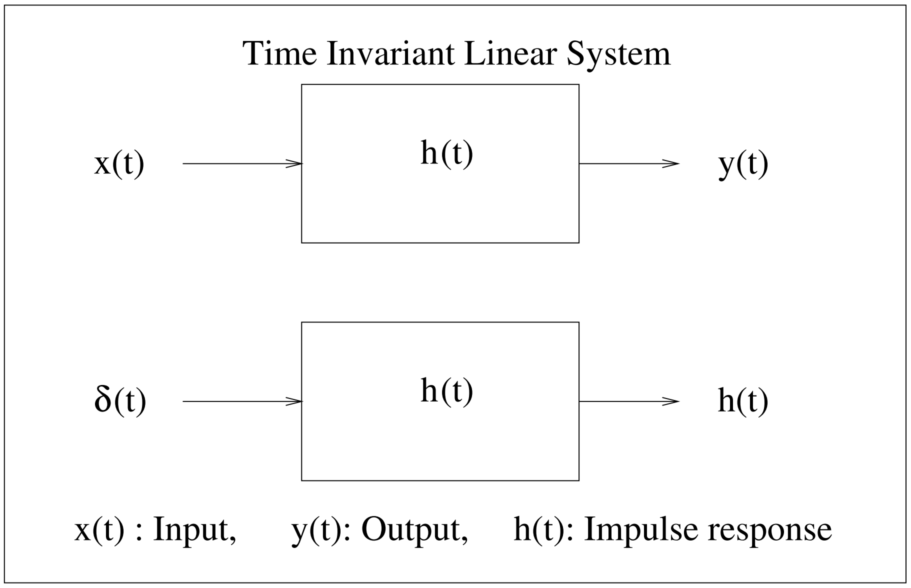


Figure 3.2: A continuous linear time invariant system. The input $x(t)$ produces an output signal denoted by $y(t)$. If the input to the system is $x(t) = \delta(t)$ the output is $y(t) = h(t)$. The signal $h(t)$ is the impulse response of the system.

$x_n,\;\; n = 0 : NX - 1$ is a signal of length NX

$y_n,\;\; n = 0 : NY - 1$ is a signal of length NY

$h_n,\;\; n = 0 : NH - 1$ is a signal of length NH .

In this case the convolution sum will be composed only of samples defined in the above intervals, i.e., $x_n, n = 0 : NX - 1$. One can write, therefore, convolution as follows

$$y_k = \sum_{n=p_1}^{p_2} h_{k-n} x_n \, , k = q_1, \ldots, q_2 \, , \tag{3.10}$$

where p_1, p_2, q_1 and q_2 indicate finite summation limits and the limits of the output signal. A simple example clarifies what are these limits. We usually

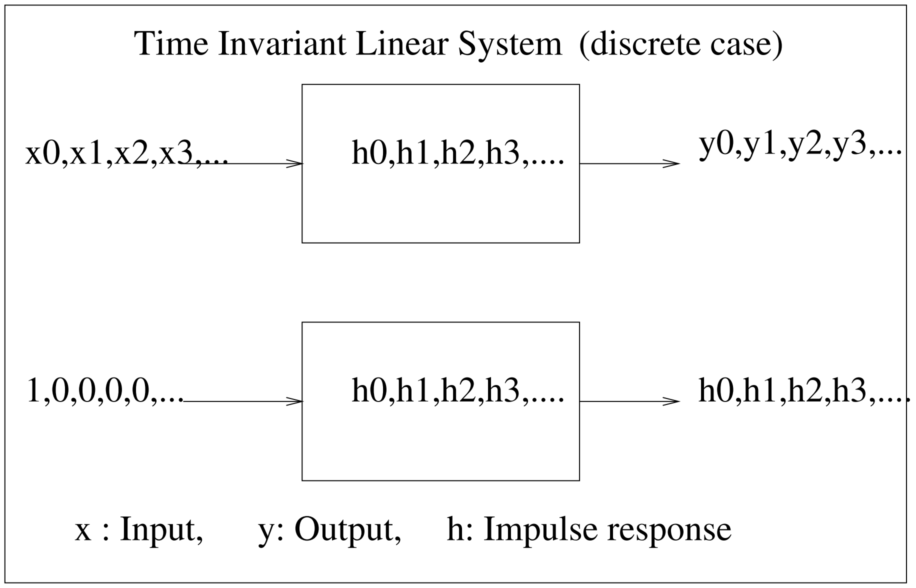


Figure 3.3: A discrete linear system. The input signal is a discrete signal x_n and the output signal is the discrete signal y_n. When the system is excited with a unit impulse signal δ_n the output is the impulse response h_n.

indicate the convolution sum with the symbol $*$

$$y_k = h_k * x_k \,. \tag{3.11}$$

Assuming that $x = [x_0, x_1, x_2, x_3, x_4]$ and $h = [h_0, h_1, h_2]$, and after carrying out the convolution sum one arrives to the following expression

$$\begin{array}{rcllll} y_0 &=& x_0h_0 & & & \\ y_1 &=& x_1h_0 & +x_0h_1 & & \\ y_2 &=& x_2h_0 & +x_1h_1 & +x_0h_2 & \\ y_3 &=& x_3h_0 & +x_2h_1 & +x_1h_2 & . \\ y_4 &=& x_4h_0 & +x_3h_1 & +x_2h_2 & \\ y_5 &=& & x_4h_1 & +x_3h_2 & \\ y_6 &=& & & x_4h_2 & \end{array} \tag{3.12}$$

Notice that to arrive to the system 3.12, I carefully ommitted tems that we do not have. For instance, terms that involve multiplications with x_5 or h_3 are zero because those samples are considered zero.

The output time series is given by $y = [y_0, y_1, y_2, ..., y_7]$. [1] Note that the above system of equations can be written as follows

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{pmatrix} = \begin{pmatrix} x_0 & 0 & 0 \\ x_1 & x_0 & 0 \\ x_2 & x_1 & x_0 \\ x_3 & x_2 & x_1 \\ x_4 & x_3 & x_2 \\ 0 & x_4 & x_3 \\ 0 & 0 & x_4 \end{pmatrix} \begin{pmatrix} h_0 \\ h_1 \\ h_2 \end{pmatrix} . \tag{3.13}$$

The last equation can be written in compact matrix-times-vector form

$$\mathbf{y} = \mathbf{X}\mathbf{h}\,. \tag{3.14}$$

Where I have used lowercase bold fonts to indicate vectors **y** and **h** and uppercase bold fonts to indicate the matrix **X**. I will try to use this convention throughout the course. The matrix **X** is the convolution matrix of the signal

[1]Please, take a look at the length of the new time series $NY = NX + NH - 1$.

(vector) **x**. This matrix is also called a Toeplitz matrix or diagonal-constant matrix, named after Otto Toeplitz, a German mathematician.

3.1.2 An algorithm to compute the convolution sum

One can see that the convolution sum can be carried out as a matrix-times-vector multiplication. However, there is a cheaper way of doing it. Take your time and look at the system given in expression 3.12. For instance, consider y_3

$$y_3 = x_3 h_0 + x_2 h_1 + x_1 h_2 \,,$$

the output index 3 is equal to the sum $3+0$ (indices of the first term of the sum), $2+1$ (indices of the second term of the sum) and $1+2$ (indices of the third term of the sum). The latter leads to a straightforward algorithm where we loop over the indices of the signal x_l and h_k to produce the output index y_{l+k}.

Please, remember that Fortran and MATLAB have a vector indexing system that looks like

```
x(1)  x(2)  x(3)  x(4) .....  x(NX)
```

where x_0 =`x(1)`, x_1 =`x(2)`, ... x_{NX-1}=`x(NX)`. This has to be taken into account at the time of writing the computer code. For example, below I provide MATLAB, Fortran and Python codes to perform the convolution of two series. You can also use the built-in MATLAB function verb+conv+ to perform the same task.

The following MATLAB scripts allows to convolve two signals. It follows the logic explained above

```
%
%  Convolution of x with h - Matlab
%  This code uses row vectors
%
x = [2, 1, 2, 3, -1];
h = [2,-1, 2];
nx = length(x);
nh = length(h);
ny= nx + nh - 1;
y = zeros(1,ny);
for j = 1:nh
  for n = 1:nx
   y(j+n-1) = y(j+n-1) + h(j) * x(n);
  end
end
```

The code in Fortran looks like

```
   subroutine convolution (nx,x,nh,h,ny,y)
c
c Convolution of two time series
c
   real x(100),y(100),h(100)
   ny = nx+nh-1
   do k=1,ny
    y(k) = 0.
   enddo
    do j = 1,nh
      do n = 1,nx
        y(j+n-1) = y(j+n-1) + h(j) * x(n)
      enddo
    enddo
   return
```

Last, I also provide the program in `Python`. Careful with `Python`, the first element is `x[0]`. Also `range(0,3)` stops at `3`, which means `0,1,2`.

```
import numpy as np
x=(2,2,1)
h=(2,-1)
nx = 3
nh = 2
y = np.zeros(nx+nh-1)
for k in range (0,nx):
    for j in range (0,nh):
        y[k+j]=y[k+j] + x[k]*h[j]
```

These codes can be used to compute the convolution of two time series. You can also form the matrix-vector multiplication as I explained in 3.14.

3.1.3 The convolution sum commutes

Consider equation 3.13 which leads to convolution expressed in matrix-times-vector form

$$\mathbf{y} = \mathbf{X}\mathbf{h}$$

where $\mathbf{X}$ is the convolution matrix with entries given by the elements of the signal $\mathbf{x}$. It is easy to reorganize the system 3.13 in the following way

$$\mathbf{y} = \mathbf{H}\mathbf{x}$$

where now $\mathbf{H}$ is the matrix formed with the elements of $\mathbf{h}$. In other words,

$$y_k = x_k * h_k \quad \rightarrow \mathbf{y} = \mathbf{X}\mathbf{h}$$

$$y_k = h_k * x_k \quad \rightarrow \mathbf{y} = \mathbf{H}\mathbf{x}$$

which shows that $h_k * x_k = x_k * h_k$.

3.2 The Z transform

A digitized seismogram, a gravity profile, a time series of monthly averages of temperature, etc. is a sequential collection of samples (a series). For instance, a 4 points times series is represented by the following collection of samples

$$x_0, x_1, x_2, x_3 \,. \tag{3.15}$$

In the following, x_n indicates the sample at time $n\Delta t$. This signal can be obtained by uniformly sampling a continuous signal periodically every Δt seconds.

The z transform of x_k, $k = 0, 1, 2, \ldots$ is defined as

$$X(z) = \sum_{k=0}^{\infty} x_n z^n \tag{3.16}$$

For a finite length time series $x_k, k = 0, \ldots, N-1$ we write

$$X(z) = \sum_{k=0}^{N-1} x_n z^n \,. \tag{3.17}$$

A simple example is a time series composed of 4 samples

$$x = \underset{\uparrow}{4}, 12, -1, -3 \,, \tag{3.18}$$

where the arrow indicates the sample x_0. The z transform of this series is a polynomial in the variable z of degree 3

$$X(z) = 4 + 12z - 1z^2 + 3z^3 \,. \tag{3.19}$$

Now, let us assume that we have a noncausal sequence[2]

$$x = -1, 3, 4, \underset{\uparrow}{3}, 5, 6, -10 \,. \tag{3.20}$$

[2]I will use the arrow to indicate the sample corresponding to $t = 0$, the absence of the arrow indicates that the first sample is the $t = 0$ sample.

In this case, the z transform is given by

$$X(z) = -z^{-3} + 3z^{-2} + 4z^{-1} + 3 + 5z + 6z^2 - 10z^3\,. \tag{3.21}$$

3.2.1 Convolution and the z transform

Let us examine the example that we already used to prove that convolution is equivalent to matrix-times-vector multiplication. Again, we consider two times series, $x = [x_0, x_1, x_2, x_3, x_4]$ and $h = [h_0, h_1, h_2]$. The z transforms of these series are

$$X(z) = x_0 + x_1 z + x_2 z^2 + x_3 z^3 + x_4 z^4$$

$$H(z) = h_0 + h_1 z + x_2 z^2\,.$$

Now, let us compute the product of the above polynomials

$$\begin{aligned} X(z).H(z) \;=\;& x_0 h_0 + && (3.22)\\ & (x_1 h_0 + x_0 h_1) z + \\ & (x_2 h_0 + x_1 h_1 + x_0 h_2) z^2 + \\ & (x_3 h_0 + x_2 h_1 + x_1 h_2) z^3 + \\ & (x_4 h_0 + x_3 h_1 + x_2 h_2) z^4 + && (3.23)\\ & (x_4 h_1 + x_3 h_2) z^5 + \\ & (x_5 h_2) z^6 \end{aligned}$$

From the expressions in 3.12 one can see that the coefficient of this new polynomial are the samples of the time series $y = [y_0, y_1, \dots, y_6]$ obtained by the convolution of x and h, in other words, $X(z)\,.H(z)$ is also the z transform of the time series y_k

$$Y(z) = X(z)\,.H(z)\,. \tag{3.24}$$

Therefore, to convolve two time series is equivalent to multiply their z transforms

$$y_k = h_k * x_k \qquad \mathbf{y} = \mathbf{H}\mathbf{x} \qquad Y(z) = H(z).X(z)\,.$$

It is interesting to notice that we have learned three ways of doing the convolution sum. We can code up the formula, use matrix-time-vector multiplication, or use polynomial multiplication via the z transform.

3.3 First encounter with deconvolution

We will come back to this point when dealing with seismic signals. It is clear that the convolution process via the z transform entails the multiplication of two polynomials. This is only feasible for short time series.

In the convolution process two time series are convolved to produce a new time series

$$y_k = h_k * x_k \quad \to \quad Y(z) = H(z)\,.X(z)$$

In the deconvolution process we will attempt to estimate x_k from y_k and x_k. Using the z transform, this is equivalent to polynomial division

$$X(z) = \frac{Y(z)}{H(z)}\,. \tag{3.25}$$

The inverse operator is defined as

$$F(z) \;=\; \frac{1}{H(z)}\,, \tag{3.26}$$

therefore, the signal $X(z)$ can be recovered via

$$X(z) = F(z)\,.\,Y(z)\,. \tag{3.27}$$

It is clear that if one is capable of finding $F(z) = \sum_k f_k z^k$, then the coefficients f_k define the discrete inverse filter in time domain that recovers x_k via convolution

$$x_k \;=\; f_k \,*\, y_k\,. \tag{3.28}$$

This is quite important in seismological data processing. We will assume that the observed seismogram is composed of two time series: the Earth's impulse response, and the seismic wavelet (also called the source function).

s_k: Seismogram (this is what you measure)

q_k: Earth's impulse response (this is your unknown)

w_k: Wavelet (well... assume that you know it!)

where

$$s_k \; = w_k * q_k \,. \tag{3.29}$$

In the deconvolution process we attempt to design an inverse to remove the wavelet

$$s_k \; = w_k \; * q_k \,. \to S(z) = W(z).Q(z)$$

If we apply the inverse filter of the wavelet to both sides of last equation we have

$$f_k * s_k \; = f_k * w_k * q_k \,. \to F(z).S(z) = F(z).W(z).Q(z)$$

it is clear that if $F(z) = \frac{1}{W(z)}$ the output sequence is the impulse response (our unknown)

$$q_k = f_k * s_k \,.$$

In the following sections we will analyze the problem of inverting the undesired signal (w_k).

3.4 Elementary signals: dipoles

In this section we will analyze the deconvolution of very simple signals. We will see that by understanding how to work with simple signals we will be capable of dealing with more complicated signals.

3.4.1 Minimum phase dipoles

A simple manner of visualizing the properties of a time series in the z domain is by decomposing the polynomial into dipoles or elementary polynomials of the form

$$1 + az \tag{3.30}$$

As an example, we compute the Z-transform of the series $x = [4, 12, -1, 3]$

$$X(z) = 4 + 12z - 1z^2 + 3z^3 = 4(1 + \frac{1}{2}z)(1 - \frac{1}{2}z)(1 + 3z)\,. \tag{3.31}$$

We have already seen that two multiply the z transform of two time series is equivalent to convolve the time series in the time domain. Therefore, the above expression can also be expressed as convolution of several time series

$$4, 12, -1, 3 = 4[\,(1, \frac{1}{2}) * (1, -\frac{1}{2}) * (1, 3z)\ . \tag{3.32}$$

In order to simplify the problem, we will analyze the properties of a single dipole. The extension to time series that require the multiplication of several dipoles is straightforward.

Let us assume that the dipole, which I will call $D(z)$, is given by

$$D(z) = 1 + az\,. \tag{3.33}$$

This dipole corresponds to a time series composed of two elements: $1, a$. Now, let assume that we want to compute the inverse of the dipole, in other words we would like to compute a function $F(z)$ such that

$$F(z)D(z) = 1\,. \tag{3.34}$$

This problem can be solved by expanding the inverse of the dipole in a series

$$F(z) = \frac{1}{D(z)} = \frac{1}{1+az}\,, \tag{3.35}$$

if $|a| < 1$, the denominator can be expanded according to the following expression [3]:

$$F(z) = 1 - az + (az)^2 - (az)^3 + (az)^4 \ldots \,. \tag{3.36}$$

Since $|a| < 1$ the above series is a convergent series. $F(z)$ is the z transform of the time series $f_k, k = 0, \ldots, \infty$ given by

$$\underset{\uparrow}{1}, -a, a^2, -a^3, a^4, \ldots \tag{3.37}$$

which represent the inverse filter of the dipole. The convolution of the dipole with the filter yields

$$(\underset{\uparrow}{1}, a) * (\underset{\uparrow}{1}, -a, a^2, -a^3, a^4, \ldots) = \underset{\uparrow}{1}, 0, 0, 0, 0, 0, \ldots \tag{3.38}$$

which represent a single spike at $n = 0$.

The dipole $(1, a)$ is a minimum phase sequence provided that $|a| < 1$. We have shown that a minimum phase dipole has a casual inverse given by $1, -a, a^2, -a^3, a^4, \ldots$ If $|a| \approx 1 < 1$ the coefficients of the inverse filter will slowly tend to zero. On the other hand if $|a| \approx 0$, only a few coefficient will be required to properly model the inverse of the dipole.

We can visualize this fact with a very simple example. Let us compute the inverse of the following dipoles: $(1, 0.9)$ and $(1, 0.01)$. In the first case we have $a = 0.9$

$$F(z) = \frac{1}{1+0.9z} = 1 - 0.9z + 0.81z^2 - 0.729z^3 + 0.6561z^4 \ldots \,. \tag{3.39}$$

In the second case, we have

[3] A geometric series.

$$F(z) = \frac{1}{1+0.1z} = 1 - 0.1z + 0.01z^2 - 0.001z^3 + 0.0001z^4 \ldots . \tag{3.40}$$

It is clear that when $a = 0.1$, we can truncate our expansion without affecting the performance of the filter. To show the last statement we convolve the dipoles with their truncated inverses. In both examples, we truncate the inverse to 5 coefficients

$$(1, 0.9) * (1, -0.9, 0.81, -0.729, 0.6561) = (1, 0.0, 0.0, 0.0, 0.59) \tag{3.41}$$

$$(1, 0.1) * (1, -0.1, 0.01, -0.001, 0.0001) = (1, 0.0, 0.0, 0.0, 0.0) . \tag{3.42}$$

It is clear that the truncation is negligible when $a = 0.1$. This is not true when $a \approx 1$. In this case, a long filer is needed to properly invert the dipole. The shortcoming above can be overcome by adopting a least-squares strategy to compute the inverse filter (this is the basis of *spiking deconvolution*.)

So far we have define a minimum phase dipole as a signal of the type $(1, a)$ where $|a| < 1$. It is important to stress that the z transform of this signal has a root, ξ, which lies outside the unit circle,

$$X(z) = 1 + az \Rightarrow X(\xi) = 1 + a\xi = 0 \Rightarrow \xi = -\frac{1}{a} \tag{3.43}$$

since $|a| < 1$, the root satisfies the following $|\xi| > 1$.

A seismic signal is more complicated than a simple dipole. But we can always factorize the z transform of any signal in terms of elementary dipoles. If the signal is minimum phase, the decomposition is in terms of minimum phase dipoles

$$X(z) = x_0 + x_1 z + x_2 z^2 + x_3 z^3 \ldots = A(1+a_1 z)(1+a_2 z)(1+a_3 z) \ldots . \tag{3.44}$$

If $|a_i| < 1, \forall i$, the signal is a minimum phase signal. In this case, all the zeros lie outside the unit circle

$$X(\xi) = 0 \Rightarrow \xi_i = -\frac{1}{a_i} \Rightarrow |a_i| < 1 \Rightarrow |\xi_i| > 1\,. \tag{3.45}$$

Now, let us assume that $X(z)$ is a minimum phase signal of length N, that can be factorized in terms of minimum phase dipoles. The inverse filter $F(z)$ of $X(z)$ must satisfy the following expression

$$\begin{aligned} & X(z)F(z) = 1 \qquad (3.46)\\ & (1+a_1z)(1+a_2z)(1+a_3z)\ldots.F(z) = 1\,. \end{aligned}$$

From the above equation, we can write

$$\begin{aligned} F(z) &= (1+a_1z)^{-1}(1+a_2z)^{-1}(1+a_3z)^{-1}\ldots \qquad (3.47)\\ &= [(1-a_1z+(a_1z)^2-(a_1z)^3\ldots][(1-a_2z+(a_2z)^2-(a_2z)^3\ldots]\\ & \quad [(1-a_3z+(a_3z)^2-(a_3z)^3\ldots]\ldots. \end{aligned}$$

The inverse operator can be written as

$$f_0, f_1, f_2, f_3, \ldots = (1, -a_1, a_1^2, -a_1^3) * (1, -a_2, a_2^2, -a_2^3) * (1, -a_3, a_3^2, -a_3^3)\ldots \tag{3.48}$$

In Figures 3.4, 3.5 and 3.6 we examine the inverse of various minimum phase dipoles. In the first case (Figure 3.4), the root is close to the unit circle, and therefore the inverse filter requires a large number of the coefficient to avoid truncation artifacts. In Figures 3.5 and 3.6, we have used dipoles with roots at $\xi = 2$ and $\xi = 10$, respectively. In these examples, the truncation artifacts are minimal.

3.4.2 Maximum phase dipoles

Elementary signal of the form $(1, b)$, $|b| > 1$ are called maximum phase dipoles. A maximum phase dipole has a zero inside the unit circle

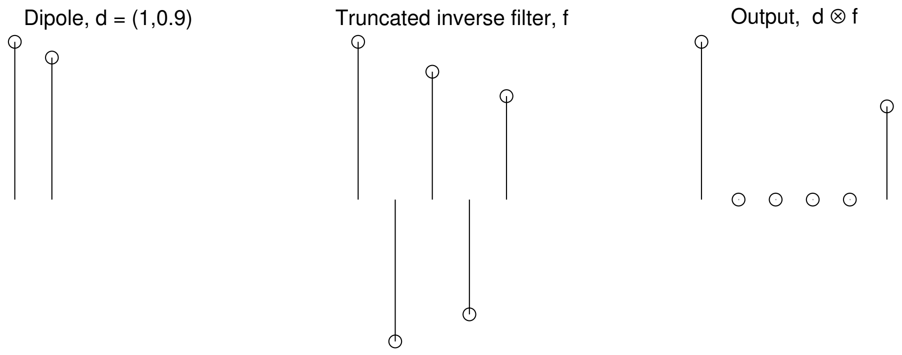


Figure 3.4: Inversion of a minimum phase dipole. The slow convergence of the inverse filter is a consequence of having a zero close to the unit circle.

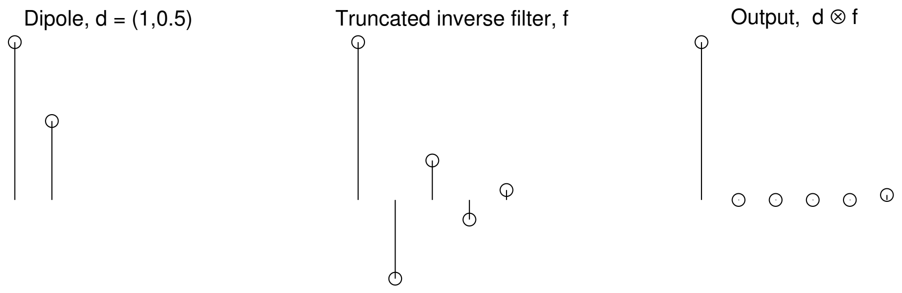


Figure 3.5: Inversion of a minimum phase dipole.

$$D(z) = 1 + bz \Rightarrow D(\xi) = 1 + b\xi = 0 \Rightarrow \xi = -1/b\,. \qquad (3.49)$$

Since $|b| < 1$, it is easy to see that $|\xi| < 1$. In this section we will prove that the inverse of a maximum phase dipole is a noncasual sequence. The inverse of the maximum phase dipole can be computed by expanding the denominator in series

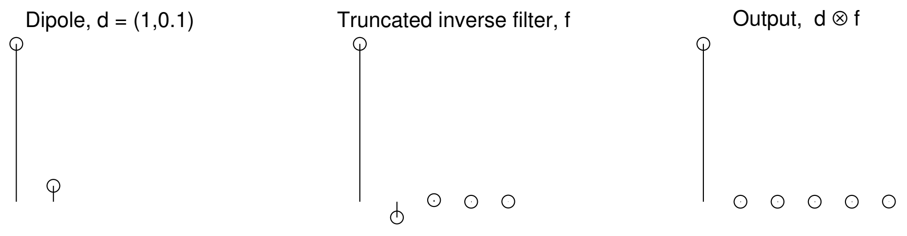


Figure 3.6: Inversion of a minimum phase dipole. In this case the zero of the dipole is far from the unit circle, this explains the fast convergence of the inverse filter.

$$F(z)D(z) = 1 \Rightarrow F(z) = \frac{1}{D(z)} = \frac{1}{1+bz}\,. \tag{3.50}$$

If last equation is expanded in a series of positive powers of z we have

$$\frac{1}{1+bz} = 1 - bz + (bz)^2 - (bz)^3 \dots \tag{3.51}$$

The later is a series that does not converge; the magnitude of the coefficients of the operator $(1, -b, b^2, -b^3 \dots)$ increases as we add more terms. The trick to overcoming this problem is to compute a *stable non-casual* operator. First, we rearrange expression (3.50)

$$F(z) = \frac{1}{1+bz} = \frac{1}{bz(1+(bz)^{-1})} \tag{3.52}$$

this expression admits a stable expansion of the form

$$F(z) = (bz)^{-1}(1 - (bz)^{-1} + (bz)^{-1} - (bz)^{-3} \dots)\,. \tag{3.53}$$

Now the inverse is stable and noncasual, the associated operator is given by

$$f = \dots, -b^{-3}, b^{-2}, -b^{-1}, \underset{\uparrow}{0}\,. \tag{3.54}$$

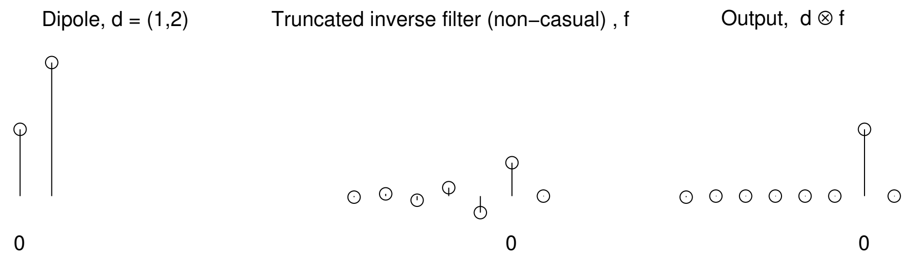


Figure 3.7: A maximum phase dipole. Its noncasual truncated inverse, f, and the output $d * f$.

The following example will clarify the problem. First, given the maximum phase dipole $(1, 2)$ we compute the noncasual inverse sequence (truncated to 6 coefficients):

$$f = (-0.0156, 0.0312, -0.0625, 0.125, -0.25, 0.5, \underset{\uparrow}{0}) \tag{3.55}$$

the convolution of f with the maximum phase dipole produces the following output sequence

$$\begin{aligned} d * f &= (-0.0156, 0.0312, -0.0625, 0.125, -0.25, 0.5, \underset{\uparrow}{0}) * (\underset{\uparrow}{1}, 2) \qquad (3.56) \\ &= (-0.0156, 0, 0, 0, 0, 0, \underset{\uparrow}{1}, 0) \,. \end{aligned}$$

Figure 3.7 provides an example where we compute the stable (convergent) inverse of a maximum phase dipole.

3.4.3 Autocorrelation function of dipoles

The autocorrelation function of a sequence with z-transform $X(z)$ is defined as

$$R(z) = X(z)X^*(z^{-1}) \tag{3.57}$$

In this section, we will analyze properties of minimum and maximum phase dipoles that are very useful at the time of designing deconvolution operators.

We will consider two dipoles. First, a minimum phase dipole of the form $(1, a)$, $|a| < 1$ and then a maximum phase dipole of the form $(a^*, 1)^4$. In the z domain, these two dipoles are expressed as follows

$$D_{min}(z) = 1 + az \tag{3.58}$$

and

$$D_{max}(z) = a^* + z\,. \tag{3.59}$$

The autocorrelation function of the minimum phase dipole is given by

$$R_{min}(z) = a^* z^{-1} + (1 + |a|^2) + a\,z\,. \tag{3.60}$$

Similarly, the autocorrelation function of the maximum phase dipole is given by

$$R_{max}(z) = a^* z^{-1} + (1 + |a|^2) + a\,z\,. \tag{3.61}$$

We have arrived at a significant conclusion

$$R_{max}(z) = R_{min}(z) = R(z) \tag{3.62}$$

or, in other words, two different sequences can have the same autocorrelation function. The autocorrelation sequence in both cases is the following time series

$$a^*, \underset{\uparrow}{(1 + a^2)}, a \tag{3.63}$$

or

[4] Note that for real dipoles, $a^* = a$

$$r_k = \begin{cases} a & \text{if } k=1 \\ 1+a^2 & \text{if } k=0 \\ a^* & \text{if } k=-1 \\ 0 & \text{otherwise} \end{cases} \tag{3.64}$$

If the dipoles are real ($a = a^*$), the the autocorrelation function is given by a symmetric sequence about zero. Note that the autocorrelation function $R(z)$ is the z-transform of the autocorrelation sequence

$$R(z) = r_1 z^{-1} + r_0 + r_1 z^{-1} = a^* z^{-1} + (1+a^2) + a z^{-1} \,. \tag{3.65}$$

In general, for more complicated signals (so far, we have only considered dipoles), the autocorrelation function of the signal is the Z-transform of the autocorrelation sequence which is given by

$$r_k = \sum_n x_n^* x_{n+k} \,, \tag{3.66}$$

$$R(z) = X(z)\,.X^*(z^{-1})\,, \tag{3.67}$$

where k is the time-lag of the autocorrelation function.

Let us assume that we are only able to measure the autocorrelation of a dipole. Given the autocorrelation of the dipole, you are asked to find the associated dipole. You have two possible solutions. One is the minimum phase dipole; the other is the maximum phase dipole. It is also true that these two sequences have the same amplitude spectrum. We define the amplitude spectrum using the Discrete-time Fourier Transform (DTFT)[5]

$$R(\omega) = R(z)|_{z=e^{-i\omega}} \qquad \omega \in [-\pi, \pi] \tag{3.68}$$

[5]This is the Fourier Transform for discrete series

or

$$R(\omega) = [X(z)\,.X^*(z^{-1})]_{z=e^{-i\omega}} \tag{3.69}$$

To evaluate the amplitude spectrum of the signal, we replace z by $e^{-i\omega}$. This is equivalent to adopt Discrete-time Fourier transform instead of the z transform. We will come back to this point in again when deriving the Discrete-time Fourier Transform. If the signal is a minimum phase dipole, it Fourier transform is given by

$$D_{min}(z) = 1 + az \;\Rightarrow z = e^{-i\omega} \;\Rightarrow D_{min}(\omega) = 1 + ae^{-i\omega}\,. \tag{3.70}$$

Whereas for the maximum phase dipole

$$D_{max}(z) = a + z \;\Rightarrow z = e^{-i\omega} \;\Rightarrow D_{max}(\omega) = a + 1e^{-iw} \tag{3.71}$$

Now can now evaluate the amplitude and phase spectrum of the minimum and maximum phase dipoles

$$R_{D_{min}}(\omega) = \sqrt{1 + 2a\,\cos(\omega) + a^2} \tag{3.72}$$

$$\theta_{min}(\omega) = \arctan(\frac{a\,\sin(\omega)}{1 + a\,\cos(\omega)})\,. \tag{3.73}$$

For the maximum phase dipole, we have

$$R_{D_{max}}(\omega) = \sqrt{1 + 2a\,\cos(\omega) + a^2} \tag{3.74}$$

$$\theta_{max}(\omega) = \arctan(\frac{\sin(\omega)}{a + \cos(\omega)})\,. \tag{3.75}$$

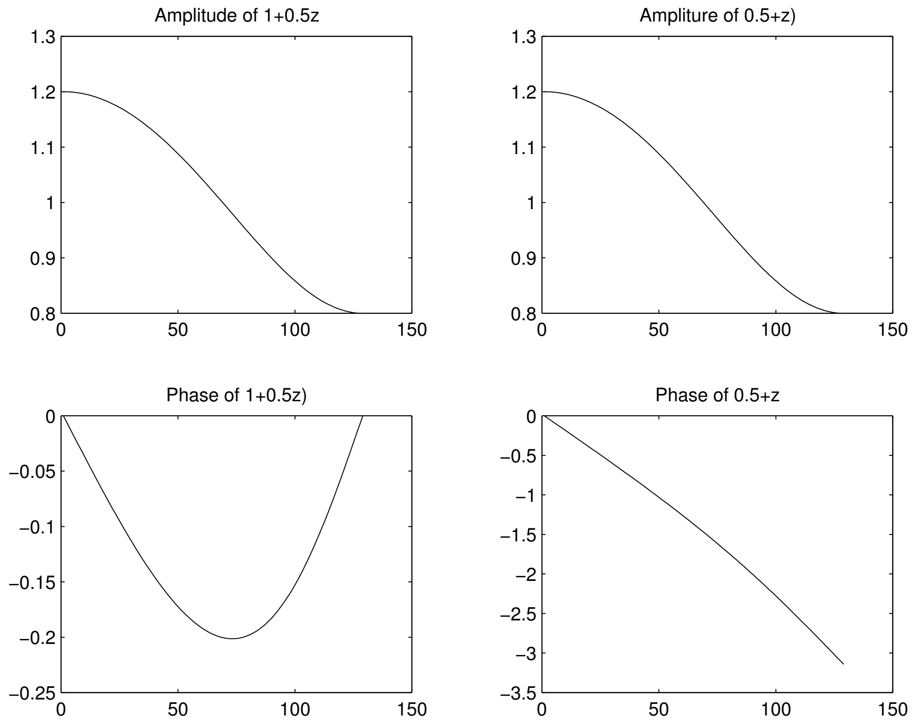


Figure 3.8: Amplitude and phase spectrum of a minimum phase dipole $1+az$ and a maximum phase dipole $a*+z$, $|a|<1$.

In Figure (3.8), we portray the amplitude and phase spectrum for a minimum phase dipole of the form $(1, 0.5)$ and a maximum phase dipole $(0.5, 1)$. Note that the amplitude spectra of these signals are equal, and their phase spectra are different.

We will see this has implications when estimating wavelet from autocrorelation functions, a classical problem in exploration seismology.

3.4.4 Least-squares inversion of a minimum phase dipole

We have already seen that one of the problems of inverting a dipole via a geometric series is that the filter results in a long operator. This is particularly true when the dipole has zero close to the unit circle.

Our problem is to find a filter where, when applied to the dipole, its output resembles the ideal output one would have obtained by using an infinite number of terms in the series expansion of the filter $F(z) = 1/(1+az) = 1 - az + (az)^2 - (az)^3 + \ldots$.

In our case, we want to invert the minimum phase dipole $(1, a)$, $|a| < 1$[6]. In preceding sections we found an expression for the ideal inverse filter, the z transform of the ideal inverse filter satisfies the following equation

$$D(z)F(z) = 1\,. \tag{3.76}$$

Now, our task is to construct a finite length filter with the following property

$$D(z)F_N(z) \approx 1\,, \tag{3.77}$$

where $F_N(z)$ denotes the z transform of the finite length operator of length N. If we assume a filter of length $N = 3$, the above equation can be written in the time domain as

$$(1, a) * (f_0, f_1, f_2) \approx (\underset{\uparrow}{1}, 0, 0, 0)\,. \tag{3.78}$$

The latter can be written in matrix form as follows

$$\begin{pmatrix} 1 & 0 & 0 \\ a & 1 & 0 \\ 0 & a & 1 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \end{pmatrix} \approx \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}\,. \tag{3.79}$$

The last system of equations corresponds to an over-determined system of equations. The length of the filter is $N = 3$, the length of the dipole is $M = 2$; therefore, the length of the desired output is $4 = M + N - 1$. To avoid notational clutter, we will represent the last system as follows

[6]Let us consider that $a = a^*$, (a is real)

$$\mathbf{C}\mathbf{f} \approx \mathbf{b}\,, \tag{3.80}$$

where $\mathbf{C}$ is the matrix that contains the dipole padded adequately with zeros to appropriately represent the convolution ($*$). The unknown inverse filter is denoted by the vector $\mathbf{f}$ and the desired output by $\mathbf{o}$. I have used the symbol $\approx$ to stress that the filter of length N convolved with the dipole with approximate the desired (ideal) output because an infinite length filter is approximated by one of finite length. Equation 3.80 can also be expressed as follows

$$\mathbf{C}\mathbf{f} - \mathbf{b} = \mathbf{e}\,, \tag{3.81}$$

$\mathbf{f}$ is where $\mathbf{e}$ is the approximation error. The least-squares inverse filter is obtained via the least-squares method where the filter is computed by miniminzing the sum of the squeres of the errors $\mathbf{e}$ [7]

$$J = \|\mathbf{e}\|_2^2 = \|\mathbf{C}\,\mathbf{f} - \mathbf{o}\|^2\,. \tag{3.82}$$

The least-squares solution is obtained by minimizing J respect to $\mathbf{f}$. The latter leads to the following system of normal equations

$$\mathbf{C}^T\,\mathbf{C}\,\mathbf{f} = \mathbf{C}^T\,\mathbf{o}\,. \tag{3.83}$$

Now, we have a system of the equations where the solution is computed by inverting the square matrix $\mathbf{C}^T\,\mathbf{C}$,

$$\hat{\mathbf{f}} = \mathbf{R}^{-1}\mathbf{C}^T\,\mathbf{o}\,, \tag{3.84}$$

where $\mathbf{R} = \mathbf{C}^T\,\mathbf{C}$. The story does not end here, it turns out that the matrix $\mathbf{R}$ has a special structure,

[7]The cost function J can be written as follows $J = \sum_k e_k^2 = \|\mathbf{e}\|_2^2 = \mathbf{e}^T\mathbf{e}\,.$

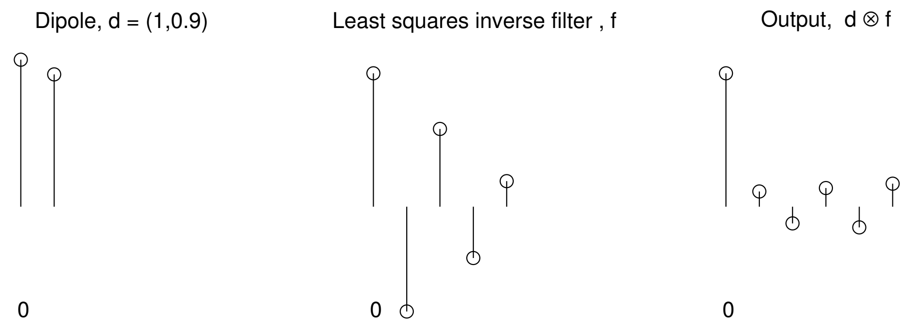


Figure 3.9: Inversion of a minimum phase dipole using the least-squares method.

$$\mathbf{R} = \begin{pmatrix} 1+a^2 & a & 0 \\ a & 1+a^2 & a \\ 0 & a & 1+a^2 \end{pmatrix} . \tag{3.85}$$

One can see that each row of the matrix $\mathbf{R}$ is composed of elements of the autocorrelation sequence given by equation (3.64)

$$\mathbf{R} = \begin{pmatrix} r_0 & r_1 & 0 \\ r_1 & r_0 & r_1 \\ 0 & r_1 & r_0 \end{pmatrix} . \tag{3.86}$$

The above matrix is a Toeplitz form. One interesting feature of a Toeplitz matrix (in this case, it is symmetric Toeplitz matrix) is that only one row of the matrix is needed to define all its elements. It is also a constant diagonal matrix. This unique structure is used by a fast algorithm, Levinson's algorithm, to solve equation 3.83 in N^2 operations where N is the length of the filter.

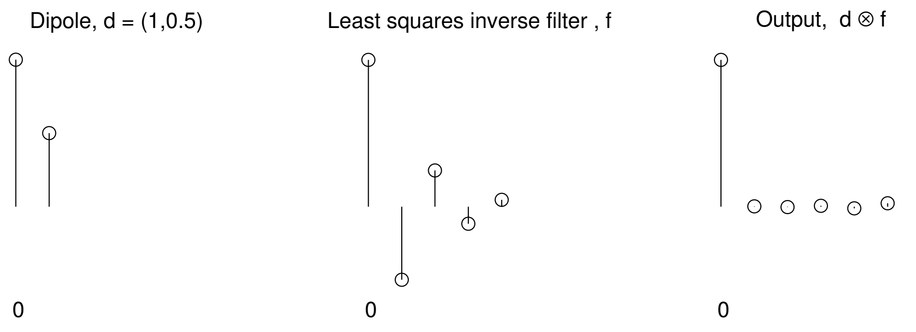


Figure 3.10: Inversion of a minimum phase dipole using the least-squares method.

3.4.5 inversion of minimum phase sequences

So far, we have discussed the problem of inverting elementary dipoles, and we have observed that minimum phase dipoles accept a casual and stable inverse.

This is also valid for more complicated signals (i.e., a seismic wavelet). In this case, the convolution matrix columns are given by the wavelet of length NW padded adequately with zeros to represent convolution.

Given a minimum phase wavelet, this a signal that can be decomposed trough factorization in minimum phase dipoles[8], the goal is to find the inverse filter of the wavelet. This is, again, the filter that converts the wavelet into a spike via convolution. Given the wavelet $w_k, k = 0, \dots, NW-1$, the filter $f_k, k = 0 \dots, NF-1$, the goal is to find a filter such that

$$(w_0, w_1, \dots w_{NW-1}) * (f_0, f_1, \dots, f_{NF-1}) \approx (1, 0, \dots, 0)\,, \tag{3.87}$$

where the output is a signal of length $NW + NF - 1$. In matrix form, we can write the following expression (assuming $NW = 7$ and $NF = 4$)

[8]In other words, the zeros of the z transform of the wavelet are outside the unit circle.

$$
\begin{pmatrix}
w_0 & 0 & 0 & 0 \\
w_1 & w_0 & 0 & 0 \\
w_2 & w_1 & w_0 & 0 \\
w_3 & w_2 & w_1 & w_0 \\
w_4 & w_3 & w_2 & w_1 \\
w_5 & w_4 & w_3 & w_2 \\
w_6 & w_5 & w_4 & w_3 \\
0 & w_6 & w_5 & w_4 \\
0 & 0 & w_6 & w_5 \\
0 & 0 & 0 & w_6
\end{pmatrix}
\begin{pmatrix}
f_0 \\ f_1 \\ f_2 \\ f_3
\end{pmatrix}
\approx
\begin{pmatrix}
1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0
\end{pmatrix}
. \tag{3.88}
$$

Again, this system is written in matrix form as $\mathbf{Cf} \approx \mathbf{d}$. We will compute the inverse filter by minimizing the error function (mean squared error) ϵ:

$$
J = ||\mathbf{e}||^2 = ||\mathbf{C}\,\mathbf{f} - \mathbf{o}||^2 \,. \tag{3.89}
$$

The Euclidean norm of the error vector $\mathbf{e} = \mathbf{C}\,\mathbf{f} - \mathbf{b}$ can be written down as

$$
J = \mathbf{e}^T\,\mathbf{e} = (\mathbf{C}\,\mathbf{f} - \mathbf{o})^T\,(\mathbf{C}\,\mathbf{f} - \mathbf{o})\,. \tag{3.90}
$$

The cost function J is minimized with respect to the unknown filter $\mathbf{f}$

$$
\frac{d\epsilon}{d\mathbf{f}} = 0\,. \tag{3.91}
$$

Taking derivatives with respect to the filter coefficients and equating them to zero leads to the following system of normal equations

$$
\mathbf{C}^T\,\mathbf{C}\,\mathbf{f} = \mathbf{C}^T\,\mathbf{o}\,. \tag{3.92}
$$

It is clear that the inverse filter is solved by inverting the Toeplitz form $\mathbf{R} = \mathbf{C}^T\,\mathbf{C}$, but this matrix (which depends on the wavelet) might have a set of close eigenvalues to zero.

If the matrix is ill-conditioned, a set of eigenvalues are zero or close to zero. This will lead to numerical instabilities at the time of the inversion. This shortcoming can be avoided by using a regularization strategy. Instead of minimizing the misfit function J we will minimize the following penalized objective function

$$J = \|\mathbf{Cf} - \mathbf{o}\|_2^2 + \mu\|\mathbf{f}\|^2\,. \tag{3.93}$$

The solution is now given by a penalized least-squares estimator where the parameter μ is also called the regularization parameter [9] The condition

$$\frac{dJ}{d\mathbf{f}} = 0\,, \tag{3.94}$$

leads to the following solution

$$\hat{\mathbf{f}} = (\mathbf{R} + \mu\mathbf{I})^{-1}\mathbf{C}^T\mathbf{d}\,. \tag{3.95}$$

The parameter μ provides protection against small eigenvalues, which may lead to an unstable filter. It is important to note that we are trying to accomplish two different goals in the objective function J. On the one hand, we want to minimize the error energy $\|\mathbf{Cf} - \mathbf{o}\|_2^2\|_2^2$. On the other hand, we try to keep the filter's energy $\|\mathbf{f}\|^2$ bounded. When $\mu \to 0$, the error function will be minimum, but the filter may have an undesired oscillatory behaviour. When μ is large, the filter's energy will be small, and the error's energy will be large. In this case, we have the so-called matching filter of the form

$$\hat{\mathbf{f}} = \mu^{-1}\mathbf{C}^T\mathbf{d}\,. \tag{3.96}$$

[9] mu is also called ridge regression, damping, or pre-whitening parameter.

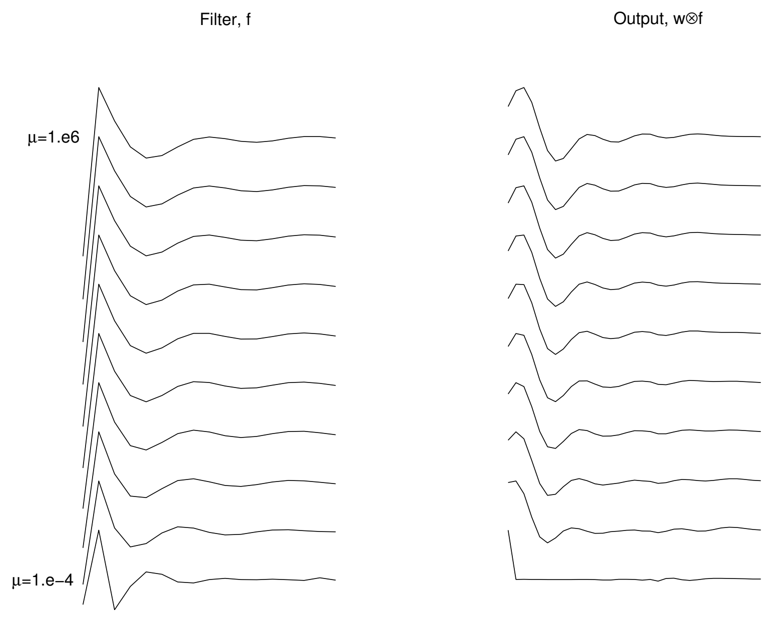


Figure 3.11: A minimum phase wavelet inverted using different tradeoff parameters (μ).

Last equation was obtained by applying the following replacement $(\mathbf{R}+\mu\mathbf{I}) \approx \mu\mathbf{I}$, which is valid only when μ is large.

In Figure (3.11) we illustrate the effect of the tradeoff parameter in the filter design and in the actual output of the deconvolution. It is clear that when μ is small, the output of the sequence is a spike. When we increase μ the output is not as sharp as when μ is closed to zero. This concept is fundamental when dealing with noisy signals. We will come back later to this problem when we analyze the deconvolution of reflectivity sequences.

In Figure (3.12) we portray the so-called tradeoff curve. This is a curve where we display the error norm $||\mathbf{Cf}-\mathbf{o}||_2^2$ versus the norm of the filter $||\mathbf{f}||_2^2$ for varying value of the tradeoff parameter μ. This curve is also called the Tikhonov curve [10] or the L-curve. This curve represents the tradeoff that exists between resolution and variance reduction in linear inverse problems.

[10]After Russian Mathematician Andrey Nikolayevich Tikhonov.

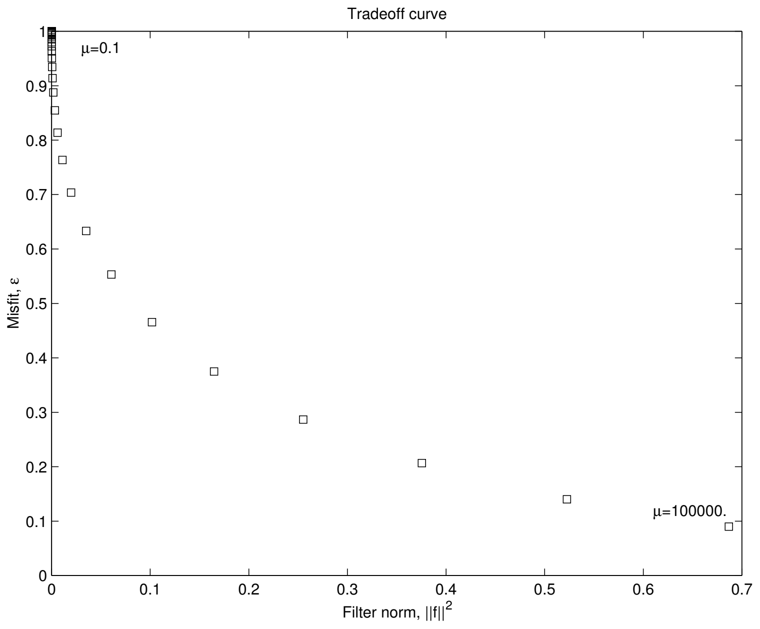


Figure 3.12: Tradeoff curve for the previous example. The vertical axis indicates the norm of the error $\|\mathbf{e}\|_2^2$, and the horizontal axis is the norm of the filter $\|\mathbf{f}\|_2^2$.

3.5 MATLAB examples

3.5.1 Inversion of dipoles

```
% Dipole.m
% This code is used to invert a minimum phase dipole via
% geometric series filter.  The inverse filter is truncated
% to N samples

N = 5;
a = 0.1;
t = 1:1:N;
d = [1 a];
f = (-a).^(t-1);
o = conv(d,f)

% Plot The dipole, the filter and the output

figure(1); stem(d); figure(2); stem(f); figure(3); stem(o);
```

3.5.2 Amplitude and phase of dipoles

```
% A MATLAB code to compute amplitude and phase
% of min and max phase dipoles

a = 0.2;

% Min phase dipole

d_min = [1,a];

% Max phase dipole

d_max = [a,1];

% Compute amplitude and phase using an FFT

D_min = fft(d_min,256);
A_min = abs(D_min); theta_min = angle(D_min);

D_max = fft(d_max,256);
A_max = abs(D_max); theta_max = angle(D_max);

% Plot the results

n = 256/2+1;
subplot(221);
plot(A_min(1:n));title('Amplitude of 1+0.2z')
subplot(222);
plot(A_max(1:n));title('Amplitude of 0.2+z)')
subplot(223);
plot(unwrap(theta_min(1:n))); title('Phase of 1+0.2z)')
subplot(224);
plot(unwrap(theta_max(1:n))); title('Phase of 0.2+z')
```

3.5.3 Least-squares inverse filter of a dipole

```
% LS_dipole.m
% Least-squares inverse of a
% minimum  phase dipole

NF = 5;
a = 0.5;
d = [1,a]';
ND = max(size(d)) ;
NO = ND+NF-1
b = [1,zeros(1,NO-1)]';
C = convmtx(d,NO-1);

R = C'*C;
rhs = C'*b;
f = inv(R)*rhs;
o = conv(f,d);
figure(1); stem(d); figure(2); stem(f); figure(3); stem(o);
```

3.5.4 Least-squares inverse filter of a wavelet.

```
function [f,o] = LS_min(w,NF,mu);
% LS_min.m
% Given an input mimimum phase wavelet w this programs
% computes the wavelet inverse filter
% and the actual output o.
% NF is the filter length.
% Note that w is a column wavelet
% mu is the pre-whitening

NW = max(size(w));
NO = NW+NF-1
b = [1,zeros(1,NO-1)]';
C = convmtx(w,NF);
R = C'*C;
rhs = C'*b;
I = eye(NF);
f = (R+mu*I)\rhs;
o = conv(f,w);
return
```

3.6 The autocorrelation function

Consider a time series of the form

$$X(z) = x_0 + x_1 z + x_2 z^2$$

and compute the following function (autocorrelation function)

$$R(z) = X(z)\, X^*(z^{-1}) \tag{3.97}$$

$$R(z) = x_0x_2^*z^{-2}+(x_0x_1^*+x_1x_2^*)z^{-1}+(x_0x_0^*+x_1x_1^*+x_2x_2^2)+(x_1x_0^*+x_2x_1^*)z+x_2x_0^*z^2\,. \tag{3.98}$$

The function $R(z)$ is the Z-transform of a sequence r_k that we call the autocorrelation sequence

$$R(z) = \sum_{k=-\infty}^{\infty} r_k z^k \tag{3.99}$$

where

$$\begin{aligned} r_{-2} &= x_0x_2^* \\ r_{-1} &= x_0x_1^* + x_1x_2^* \\ r_0 &= x_0x_0^* + x_1x_1^* + x_2x_2^* \\ r_1 &= x_1x_0^* + x_2x_1^* \\ r_2 &= x_2x_0^* \\ r_k &= 0 \quad \text{otherwise}\,. \end{aligned} \tag{3.100}$$

It is easy to show that for a time series of length NX

$$x_0, x_1, x_2, x_3, \ldots, x_{NX-1}$$

the autocorrelation coefficient can be computed using the following formulas

$$\begin{aligned} r_{-k} &= \textstyle\sum_{i=0}^{NX-1-k} x_i x_{i+k}^* \quad k = 1,2,3,\ldots,NX-1 \\ r_0 &= \textstyle\sum_{i=0}^{NX-1} x_i x_i^* \\ r_k &= \textstyle\sum_{i=0}^{NX-1-k} x_{i+k} x_i^* \quad k = 1,2,3,\ldots,NX-1 \quad \text{[Note]}^{11} \end{aligned} \tag{3.101}$$

Properties of the autocorrelation sequence:

1. Hermitian Symmetry: $r_k = r^*_{-k} \quad k = \pm 1, \pm 2, \dots$
2. $r_0 > |r_k| \quad k = \pm 1, \pm 2 \dots$
3. r_0 represents the energy of the signal; for a zero mean stationary stochastic process r_0/NX is an estimator of the variance of the process:

$$\hat{\sigma}^2 = \frac{r_0}{NX} = \frac{\sum_{k=0}^{NX-1} |x_k|^2}{NX}\,.$$

4. If $x_0, x_1, \dots, x_{NX-1}$ is a real time series then, $r_k = r_{-k}$.

3.6.1 The Toeplitz matrix and autocorrelation coefficients

We adopted the least-squares method to find an inverse operator that enables us to collapse a wavelet into a spike. We have seen that the least-squares filter is computed by solving a system of equations of the form

$$\mathbf{C}^T\mathbf{C}\mathbf{f} \;=\; \mathbf{C}\mathbf{b}\,. \tag{3.102}$$

Where $\mathbf{C}$ is a matrix with entries given by the wavelet properly pad with zeros and shifted to represent a convolution operator, in our example

$$\mathbf{C} = \begin{pmatrix} w_0 & 0 & 0 & 0 \\ w_1 & w_0 & 0 & 0 \\ w_2 & w_1 & w_0 & 0 \\ w_3 & w_2 & w_1 & w_0 \\ w_4 & w_3 & w_2 & w_1 \\ w_5 & w_4 & w_3 & w_2 \\ w_6 & w_5 & w_4 & w_3 \\ 0 & w_6 & w_5 & w_4 \\ 0 & 0 & w_6 & w_5 \\ 0 & 0 & 0 & w_6 \end{pmatrix} \tag{3.103}$$

This is the convolution matrix for a wavelet or length $NW = 7$ and a filter of length $NF = 4$. It is easy to see that the Toeplitz matrix $\mathbf{R} = \mathbf{C}^T\mathbf{C}$ is given by

$$\mathbf{R} = \begin{pmatrix} r_0 & r_1 & r_2 & r_3 \\ r_1 & r_0 & r_1 & r_2 \\ r_2 & r_1 & r_0 & r_1 \\ r_3 & r_2 & r_1 & r_0 \end{pmatrix} \tag{3.104}$$

where the elements of $\mathbf{R}$ are given by

$$r_k = \sum_{i=0}^{NW-1-k} w_{i+k}\, w_i \qquad k = 0, 1, 2, 3, \dots, NF-1\,. \tag{3.105}$$

The coefficients r_k are the **correlation coefficients** of the the wavelet. It is interesting to note that the zero lag autocorrelation coefficient ($k = 0$) represents the energy of the wavelet

$$r_0 = \sum_{k=0}^{NW-1} w_k^2\,. \tag{3.106}$$

It is important to stress that at the time of computing the Toeplitz the matrix we do not need to compute the product $\mathbf{C}^T\mathbf{C}$; it is more efficient to compute the elements of the Toeplitz matrix using the expression of the autocorrelation coefficients.

The following code can be used to compute the autocorrelation sequence of a real time series.

```
function [r0,r] = correlation(x);
%
% Function to compute the autocorrelation sequence
% of a real series
% IN x: time series
% OUT r0: zero lag autocorrelation
%   r : vector containing autocorrelation samples
%     for lags k=1,2,3...nx-1
%
 r0 = sum(x.*x);
 nx = length(x);
 for k=1:nx-1;
 r(k) = 0;
  for j = 1:nx-k
  r(k) = r(k) + x(j) * x(j+k);
  end
 end
```

3.7 Inversion of non-minimum phase wavelets: optimum lag spiking filters

Minimum phase wavelets are inverted using the least-squares method using the desired output of the form $(1, 0, 0, \ldots)$. The resulting filter is often called the Wiener filter or the spiking deconvolution operator. In general seismic wavelets are not minimum phase (some roots might lie inside the unit circle, they are mixed-phase). An appealing feature of the least-squares inversion approach is that the filter is also a minimum phase signal.

If the wavelet is not a minimum phase signal, the actual output (the filter's convolution with the wavelet) does not resemble the desired output $(1, 0, 0, \ldots)$. The problem can be alleviated by defining an optimum lag Wiener or Spiking filter. This is an inverse filter where the desired output is the following sequence

$$(0, 0, 0, 0 \ldots, 1, 0, 0, 0, \ldots) \tag{3.107}$$

In essence, we delay the 1 in the desired output sequence to generate a filter that turns the wavelet into a delayed spike.

The filter design problem is equivalent to what has already been studied in the preceding section. However, now the right side term in the vector of the desired output $\mathbf{o}$ is a spike that has been **delayed** by an amount we called L (lag). The optimal lag L_{opt} is given by the value of L where the actual output resembles the desired output. We need to define a measure capable of measuring how close the actual output is to the desired output. This is done by defining a filter performance norm

$$P = 1 - E \tag{3.108}$$

$$E = \frac{1}{r_0}||\mathbf{C}\,\hat{\mathbf{f}} - \mathbf{b}||^2 \tag{3.109}$$

where E is the normalized mean square error, r_0 is the zero lag autocorrelation coefficient. It can be shown that

$$0 \leq E \leq 1m.$$

When $E = 0$, we have a perfect filter where the desired and the actual output are equal. When $E = 1$, there is no agreement between the desired and the actual output. On the other hand, the filter performance is maximized, $P = 1$, for the optimal filter. In practical applications, we search for the value of L that maximizes the filter performance P, the value L where P is maximized is usually called the *optimum lag*.

Chapter 4

Discrete Fourier Transforms

This chapter presents the transition from the z transform to the Discrete Fourier Transform (DFT). The DFT is used to compute the Fourier transform of discrete data.

4.1 The discrete-time Fourier Transform

We first consider the case where the signal x_n, $-\infty < n < \infty$ is discrete and given at time samples n with associated time $t = n\Delta t$. We understand that Δt is the sampling interval, which satisfies the Nyquist condition. We have already discussed Nyquist-Shannon sampling, where we showed that a discrete signal has a periodic spectrum that represents the spectrum of the continuous signal counterpart. The periodic spectrum of the discrete signal represents the original continuous signal accurately in $\omega \in [-\pi, \pi]$ when Nyquist condition is satisfied $f_{max} < f_{Nyquist} = \frac{1}{2\Delta t}$. Then the discrete-time Fourier transform is given by the following pair

$$X(\omega) = \sum_{n=-\infty}^{\infty} x_n e^{-i\omega n} \tag{4.1}$$

$$x_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\omega) e^{i\omega n} d\omega \,. \tag{4.2}$$

At this point, we have defined $X(\omega)$ as the discrete-time Fourier transform because the time series x_n is only discrete in its time variable. Clearly, the variable ω is continuous. The inverse transform to evaluate x_n from $X(\omega)$ requires the solution of an integral. This is not an ideal transform pair to operate numerically in our computers. Therefore, we will introduce a new transform where time is discrete, and the frequency axis ω is also discrete. This new transform is called the Discrete Fourier Transform (DFT) and plays an important role in signal and imaging processing because it permits to evaluate numerically both the forward discrete Fourier transform (transform to go from the time domain to frequency domain) and the inverse discrete Fourier transform (the transform to return from the frequency domain to the time domain) [1]

4.2 The z transform and the Discrete Fourier Transform (DFT)

We have already defined the Z-transform of a time series as follows

$$X(z) = \sum_{n=0}^{N-1} x_n z^n \,. \tag{4.3}$$

The z transform provides a representation of our time series in terms of a polynomial. Let us introduce the following change of variable

$$z = e^{-i\omega} \tag{4.4}$$

in this case the z transform becomes

$$X(\omega) = \sum_{n=0}^{N-1} x_n e^{-i\omega n} \,. \tag{4.5}$$

[1] I am using time (t) and frequency ω but consider that I could have also used space x and wavenumber k if the signal depends on the spatial variable x such in the case of gravity and magnetic profiles.

We have evaluated the z transform on the unit circle. The variable z can be written as $z = \rho e^{i\theta}$ where ρ and θ represent the amplitude and phase of the complex z, respectively. If we consider the particular representation of z given by $z = e^{-i\omega}$, one is basically evaluating z on the unit circle $\rho = 1$. The phase of z is interpreted as the angular frequency ω which is given in radians. It is easy to make an analogy with the Fourier transform (Fourier integral) for continuous-time signals

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt \,. \tag{4.6}$$

In the last equation the frequency is given in *radian/sec* when the time is measured in seconds. A continuous-time signal is multiplied by the Fourier kernel $e^{-i\omega t}$ and integrated over time to yield the Fourier transform. In the DFT, the integration is replaced by summation and the Fourier kernel is now $e^{-i\omega n}$. Since n does not have units, ω must be given in radians.

So far, the frequency ω is a continuous variable, but let us assume that one wishes also to discretize ω in the same way we have discretized the temporal variable t. The limits of ω are given by $[0, 2\pi)$. Remember that ω is an angular frequency and the spectrum of a discrete signal is periodic in $(-\pi, \pi]$ or equivalently $[0, 2\pi)$. If the time series is a signal of length N points , we can discretize the frequency axis as follows

$$\omega_k = \Delta\omega\, k = \frac{2\pi}{N}\, k \,, \quad k = 0, 1, \ldots, N-1 \,. \tag{4.7}$$

In other words, we are sampling ω every $\Delta\omega = 2\pi/N$ radians. Now we can define the DFT as follows

$$X(\omega_k) = \sum_{n=0}^{N-1} x(n) e^{-i\omega_k\, n} \,, k = 0, 1, \ldots, N-1 \,. \tag{4.8}$$

or

$$X(\omega_k) = X_k = \sum_{n=0}^{N-1} x(n) e^{-i2\pi\, k\, n} \,, k = 0, 1, \ldots, N-1 \,. \tag{4.9}$$

Clearly, the DFT is a transformation of a N points signal into N Fourier coefficients $X_k = X(\omega_k)$. We can also write down our transform in matrix form

$$\begin{pmatrix} X_0 \\ X_1 \\ X_2 \\ \vdots \\ X_{N-1} \end{pmatrix} = \begin{pmatrix} 1 & & 1 & \dots & 1 \\ 1 & e^{-i2\pi/N} & e^{-i2\pi 2/N} & \dots & e^{-i2\pi(N-1)/N} \\ 1 & e^{-i2\pi 2/N} & e^{-i2\pi 4/N} & \dots & e^{-i2\pi 2(N-1)/N} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & e^{-i2\pi(N-1)/N} & e^{-i2\pi 2(N-1)/N} & \dots & e^{-i2\pi(N-1)(N-1)/N} \end{pmatrix} . \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{pmatrix} \tag{4.10}$$

The last equation can be written in compact form as follows

$$\mathbf{X} = \mathbf{F} \,.\, \mathbf{x} \,. \tag{4.11}$$

The DFT can be interpreted as a matrix-times-vector operation where a discrete-time domain signal is mapped to the frequency domain via a simple operation $\mathbf{Fx}$.

The N-dimensional vector representing the time-domain signal is transformed into another N-dimensional vector representing discrete samples of the Discrete Fourier Transform coefficients. Imagine a table of coefficients $X_0, X_1, X_2, \dots, X_{N-1}$ that correspond to complex amplitudes associated to frequency indices $0, 1, 2, \dots N-1$ or angular frequencies $\omega_0, \omega_1, \dots \omega_{N-1}$ (See table 4.1).

4.2.1 Inverse DFT

The remaining problem entails the invertibility of the DFT. We need a transform to come back from the frequency domain to the time domain. In other words, we need $\mathbf{F}^{-1}$.

We propose the following inverse transform

k	X_k	$\omega_k = 2\pi k/N$
0	X_0	0
1	X_1	$\frac{\pi}{4}$
2	X_2	$\frac{\pi}{2}$
3	X_3	$\frac{3\pi}{4}$
4	X_4	π Nyquist
5	X_5	$\frac{5\pi}{8}$
6	X_6	$\frac{3\pi}{2}$
7	X_7	$\frac{7\pi}{4}$ almost 2π

Table 4.1: Fourier coefficients and angular frequencies for $N = 8$.

$$x_n = \sum_{l=0}^{N-1} \alpha_l e^{i2\pi ln/N}\,, \tag{4.12}$$

where the coefficients α_l must be determined. This formula is analogous to the one used to invert the Fourier transform. However, it is important to note that we have interchanged the integration symbol by a summation. The parameters α_k are our unknowns. To find the unknowns, we proceed as follows. First we replace the last equation into equation (4.8),

$$X_k = \sum_{n=0}^{N-1} \sum_{l=0}^{N-1} \alpha_l e^{i2\pi n(l-k)/N}\,. \tag{4.13}$$

The last equation can be rewritten as

$$X_k = \sum_{l=0}^{N-1} \alpha_l \sum_{n=0}^{N-1} e^{i2\pi n(l-k)/N} = \sum_{l=0}^{N-1} \alpha_l s_{l-k}\,, \tag{4.14}$$

where the sequence s_{l-k} is given by

$$s_{l-k} = \sum_{n=0}^{N-1} e^{i2\pi n(l-k)/N}\,. \tag{4.15}$$

At this point, we realize the last equation is the geometric series[2] with sum given by

$$\sum_{n=0}^{N-1} u^n = \begin{cases} N & \text{if} \quad u = 1 \\ \frac{u^N}{1-u} & \text{if} \quad u \neq 1 \end{cases} . \tag{4.16}$$

In equation (4.15) we can identify $u = e^{i2\pi n(l-k)/N}$, therefore

$$s_{l-k} = \begin{cases} N & \text{if} \quad l = k \\ 0 & \text{if} \quad l \neq k \end{cases} , \tag{4.17}$$

after introducing the final result into equation (4.14) we obtain the following expression for our unknown coefficients α_k

$$X_k = N\alpha_k, \quad k = 0, \dots, N-1 \, . \tag{4.18}$$

Therefore, our inversion formula becomes

$$x_n = \frac{1}{N} \sum_{l=0}^{N-1} X_l e^{i2\pi ln/N} \, . \tag{4.19}$$

This equation can also be written as follows

$$\mathbf{x} = \frac{1}{N} \mathbf{F}^H \mathbf{X} \, . \tag{4.20}$$

The matrix $\mathbf{F}^H$ is the Hermitian transpose of the matrix $\mathbf{F}$. It is clear that the $N \times N$ matrix $\mathbf{F}$ is an orthogonal matrix,

$$\mathbf{F}^H \, \mathbf{F} = N \, \mathbf{I}_N \, , \tag{4.21}$$

[2]We have used a geometric series to find the inverse of a dipole in Chapter 2

where $\mathbf{I}_N$ is an $N \times N$ identity matrix. Finally, we have a pair of transforms, the DFT and the IDFT (inverse DFT) which are given by

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N}\,, \quad k = 0, \dots, N-1\,, \tag{4.22}$$

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k e^{i2\pi kn/N} \quad n = 0, \dots, N-1\,. \tag{4.23}$$

The DFT is used to map a discrete signal into the frequency domain. The IDFT is used to map the DFT coefficients X_k back to the time domain. Because the DFT is an orthogonal transformation, the inverse is computed using the conjugate transform (we don't need to calculate the inverse).

The cost of inverting an $N \times N$ matrix is proportional to N^3; the cost of multiplying a matrix by a vector is proportional to N^2. Therefore, computing the DFT or the IDFT has a cost proportional to N^2 operations. We will further diminish the computation cost of the DFT by using the Fast Fourier Transform (FFT).

4.2.2 Zero padding

The DFT allows us to transform an N-points time series into N frequency coefficients X_k, where the index k is associated to the discrete frequency ω_k,

$$\omega_k = \frac{2\pi k}{N} = \Delta\omega k\,, k = 0, 1, \dots, N-1\,.$$

The frequency axis is sampled every $\Delta\omega$ radians. At this point, it appears that $\Delta\omega$ is controlled by the number of samples of the time series N. Zero padding can be used to decrease the frequency interval $\Delta\omega$, in this case, we define a new time series that consists of the original time series followed by $M-N$ zeros,

$$x = [x_0, x_1, x_2, \dots, x_{N-1}\underbrace{0, 0, \dots, 0}_{M-N}]\,.$$

The new time series is an M-points time series with a DFT given by

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi nk/M} = \sum_{n=0}^{M-1} x_n e^{-i2\pi nk/M} \,, \quad k = 0, \ldots, M-1 \qquad (4.24)$$

The sampling interval of the frequency axis is now given by

$$\Delta\omega = \frac{2\pi}{M} < \frac{2\pi}{N} \,.$$

In general, the trick of zero padding is used to oversample the frequency axis when plotting the DFT. It is also vital to pad with zeros at the time of performing discrete convolution using the DFT. We will return to the last point when examining frequency domain deconvolution.

In Figures (4.1) and (4.2) we portray the effect of padding a time series. In Figure (4.1) we have the original time series and the associated DFT (the real and imaginary part). In Figure (4.2) the original time series after zero padding (20 zeros) is used to compute the DFT.

In the following example I show how to pad with zeros a time series. This codes was utilized to generate Figures (4.1) and (4.2).

```
% Zero padding - Example
N = 30;            % Length of the TS
L = 20;            % Number of zeros to pad
n = 1:1:N;
x = sin(2.*pi*(n-1)*0.1);
x = x./n;
if L>=1; x = [x, zeros(1,L)]; % Pad with zeros if L>0
N = length(x);
n = 1:1:N;
end;
X = fft(x);     % Compute the DFT
w = 2*pi*n/N;   % Compute the freq. axis in rads in [0,2pi).
subplot(311); plot(n,x); xlabel('n'); ylabel('x');
subplot(312); stem(w,real(X)); xlabel('\omega [rad]'); ylabel('Real[X_k]')
subplot(313); stem(w,imag(X)); xlabel('\omega [rad]'); ylabel('Imag[X_k]')
```

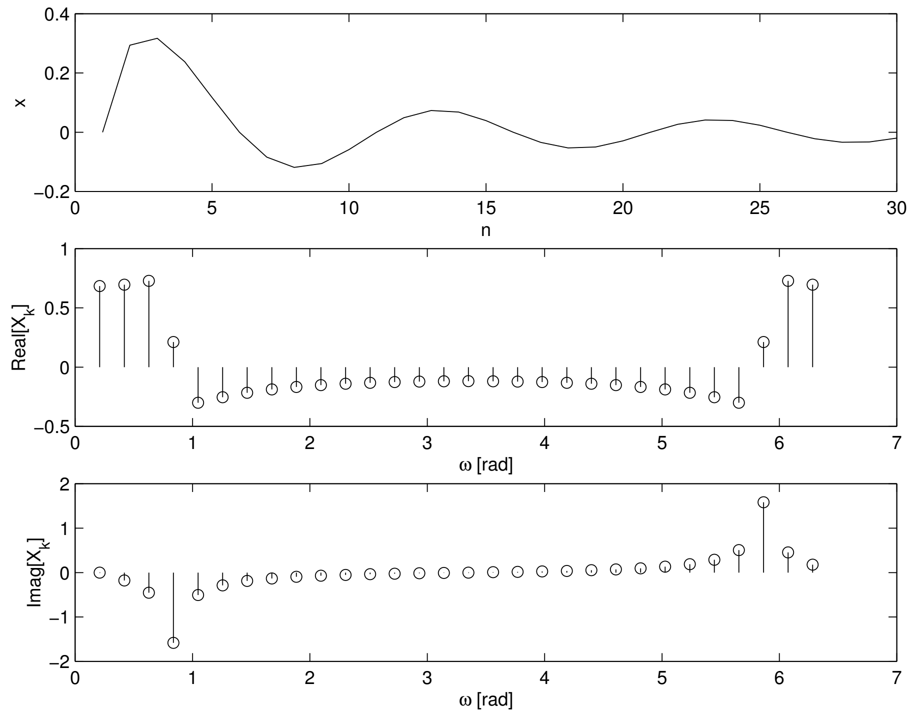


Figure 4.1: A time series and the real and imaginary parts of the DFT. Note that freq. axis is given in radians $(0, 2\pi)$

4.2.3 The Fast Fourier Transform (FFT)

The FFT is not a new transform; the FFT is just a fast algorithm to compute the DFT. The FFT is based on the halving trick, that is a trick to compute the DFT of a length N time series by using the DFT of two sub-series of length N/2. Let's start assuming that we have a time series of length $2N$

$$z_0, z_1, z_2, z_3, \dots, z_{2N-1} \,.$$

First, we will assume that one wants to compute the DFT of the time series z. Using the definition

$$Z_k = \sum_{n=0}^{2N-1} z_n \, e^{-i2\pi nk/(2N)}, \;\; k = 0 : 2N-1 \,, \tag{4.25}$$

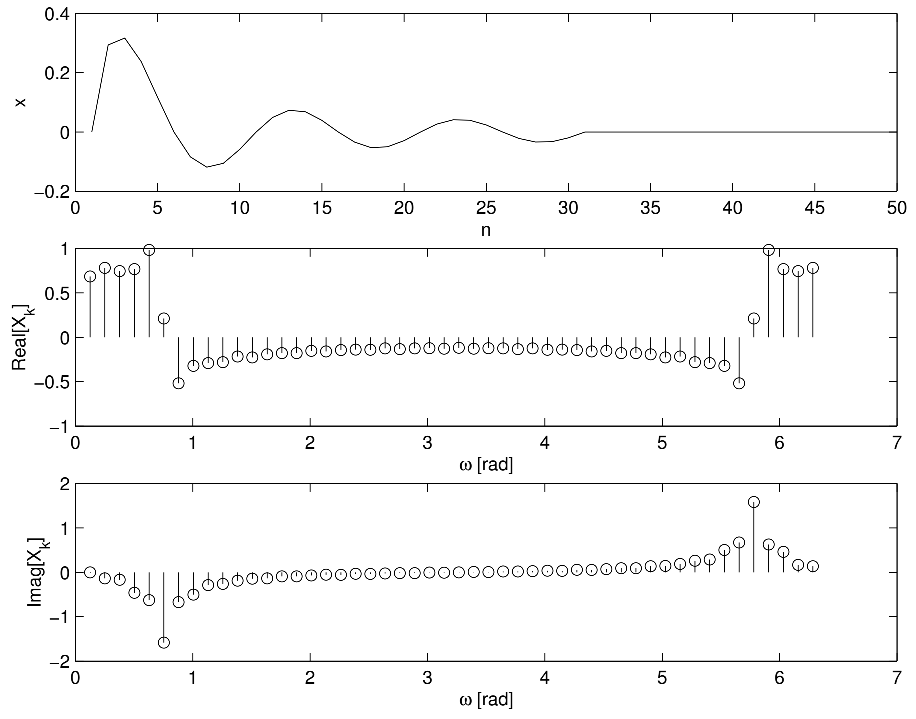


Figure 4.2: A time series and the real and imaginary parts of the DFT. In this case the time series was padded with zeros in order to decrease the frequency interval $\Delta\omega$.

we can rewrite the last equation in terms of two time series composed of even samples $x = z_0, z_2, z_4 \ldots$ and odd samples $y = z_1, z_3, z_5 \ldots$, respectively.

$$Z_k = \sum_{n=0}^{N-1} z_{2n}\, e^{-i2\pi 2nk/(2N)} + \sum_{n=0}^{N-1} z_{2n+1}\, e^{-i2\pi(2n+1)k/(2N)} \,. \tag{4.26}$$

The right hand side term can be written in terms of the DFTs of x (even samples) and y (odd samples)

$$Z_k = X_k + e^{-i2\pi k/(2N)}\, Y_K, \;\; k = 0 : N-1 \,. \tag{4.27}$$

The last equation provides a formula to compute the first N samples of the DFT of z based on the N samples of the DFT of x and y. Now, note that we need another formula to retrieve the second half of the samples of the DFT of z,

$$Z_k = \sum_{n=0}^{2N-1} z_n \, e^{-i2\pi nk/(2N)} \, , k = N, \ldots, 2N-1 \, . \tag{4.28}$$

In the last equation we apply the following substitution: $j = k - N$

$$Z_{j+N} = \sum_{n=0}^{2N-1} z_n \, e^{-i2\pi n(j+N)/(2N)} \, , k = N, \ldots, 2N-1 \, . \tag{4.29}$$

After rewriting the last expression in terms of x and y we end up with the following formula

$$Z_{j+N} = X_j - e^{-i2\pi k/(2N)} \, Y_j \, , \;\; j = 0, N-1 \, . \tag{4.30}$$

Now we have two expressions to compute the DFT of a series of length $2N$ as a function of two time series of length N. Good FFT algorithms repeat this trick until the final time series are series of length 1. The recursions given in (4.27) and (4.30) are applied to recover the DFT of the original time series. It can be proved that the total number of operations of the FFT is proportional to $N \ln_2(N)$ (for a time series of length N). This is an important saving compared to the standard DFT, which involves a number of operations proportional to N^2.

A simple modification to formulas (4.27) and (4.30) will permit us to compute the inverse DFT.

I would recommend always padding your signal with enough zeros so that the length of the signal given to the FFT algorithm is 2^K, where K is an integer. The latter leads to signals of length $2, 4, 8, 16, 32, 64, 128, 256, 512, 1024$... and to extremely efficient FFTs. For instance, if your signal has $N_t = 99$ points, I advise you to pad it with enough zeros to take it to $N = 128 > N_t$ points. You might want to make $\Delta\omega$ really small for plotting purposes; therefore, you could also choose to pad to a length $N = 256$ or $N = 512$.

I will use the name DFT when referring to the transform itself, which transforms data from discrete-time to discrete-frequency. However, when I write computer code, I will use the FFT. Results are equivalent; the only difference is that the FFT is a fast implementation of the DFT. There are applications where one might need to write a simple DFT sum, such as when the input signal is unevenly sampled, but this is probably a topic for discussion in a more advanced portion of a geophysical signal processing course.

4.2.4 Working with the DFT/FFT

Symmetries of the DFT are essential and often a real headache when developing codes. Getting them right is critical. Consider a real-time series, for instance, particle velocity of the ground measured by a geophone and sampled every Δt seconds. This signal is input to the DFT to carry out a given process in the frequency domain, such as filtering; once you apply the frequency filter and return to the time domain via the inverse DFT, the signal must be real. If you don't consider the DFT symmetries for real signals, you might end up with an output signal that contains an imaginary part and possibly the wrong real part. Trying to fix this in Matlab or Python by extracting the real part of the output will only hide a coding error under the carpet.

Symmetries

Let us start with the DFT of a **real** time series of length N

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi nk/N}\,, \;\; k = 0, \dots, N-1 \tag{4.31}$$

the frequency in radians is given by

$$\omega_k = 2\pi k/N\,, k = 0, 1, \dots, N-1\,.$$

Using the following property

$$e^{i2\pi(N-k)n/N} = e^{-i2\pi kn/N} \tag{4.32}$$

we can re-write equation (4.31) as follows:

$$X_{N-k} = \sum_{n=0}^{N-1} x_n e^{-i2\pi n(N-k)/N} = \sum_{n=0}^{N-1} x_n e^{i2\pi n(N+k)/N} \; = \; X_k^*\,. \tag{4.33}$$

The following example is used to illustrate the last point. The time series is $x = [2, 3, 1, 3, 4, 5, -1, 2]$ The DFT is given by

```
Sample k     X_k          N-k (N=8)

 0    19.0000               8
 1    -4.1213 - 1.2929i     7
 2     6.0000 - 3.0000i     6
 3     0.1213 + 2.7071i     5
 4    -7.0000               4
 5     0.1213 - 2.7071i     3
 6     6.0000 + 3.0000i     2
 7    -4.1213 + 1.2929i     1
```

The first $N/2+1$ samples are required to define the remaining $N/2-1$ samples of the DFT. For a real-time series of length N where N is even, one has N/2+1 independent DFT coefficients. It is important to note that the first $N/2+1$ samples correspond to positive frequencies. The remaining correspond to negative frequencies, which also correspond to frequencies in $(\pi, 2\pi)$. Stay tuned for the next subsection.

The frequency axis

In the previous example I compute the DFT, X_k in terms of samples k. We have already mentioned that k is related to angular frequency as follows: $\omega_k = 2\pi k/N$. Let us define the sampling interval of the frequency axis as $\Delta\omega = 2\pi/N$, therefore, $\omega_k = \Delta\omega\, k$, $k = 0, \ldots, N-1$. In the previous example we have

```
k  omega_k         X_k
0     0      19.0000
1  0.7854      -4.1213 - 1.2929i
2  1.5708       6.0000 - 3.0000i
3  2.3562       0.1213 + 2.7071i
4  3.1416      -7.0000
```

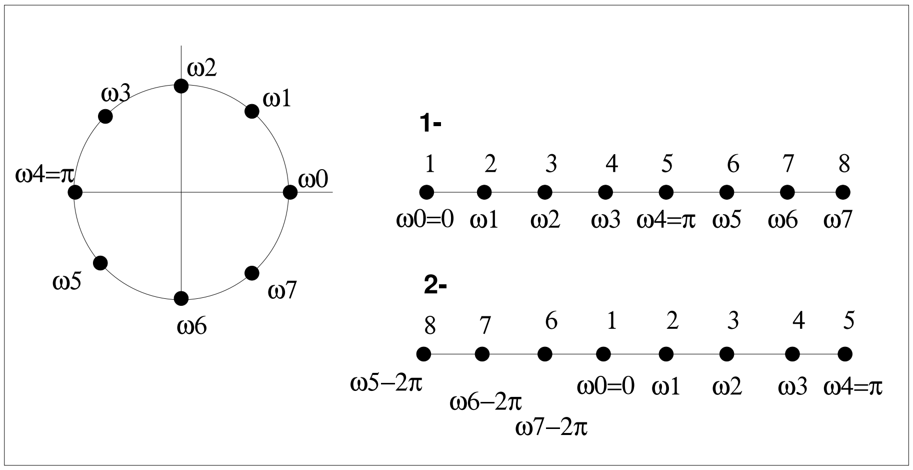

ω2
ω3
ω1
ω4=π
ω0
ω5
ω7
ω6
1-
1 2 3 4 5 6 7 8
ω0=0 ω1 ω2 ω3 ω4=π ω5 ω6 ω7
2-
8 7 6 1 2 3 4 5
ω5−2π ω6−2π ω7−2π ω0=0 ω1 ω2 ω3 ω4=π

Figure 4.3: Distribution of frequencies of the DFT. The DFT can be plotted as in the $[0, 2\pi)$ interval or in the $(-\pi, \pi]$ interval. The output of the DFT (or FFT) organizes Fourier coefficients in $[0, 2\pi)$ but humans prefer to visualize the coefficients versus a frequency axis in $(-\pi, \pi]$ which leaves the X_0 complex amplitude in the centre (almost the centre, there is no centre because N is even) of the plot for frequency $\omega = 0$.

```
5  3.9270        0.1213 - 2.7071i
6  4.7124        6.0000 + 3.0000i
7  5.4978       -4.1213 + 1.2929i
```

Note that the central frequency is $\omega_4 = \pi$, the last frequency is almost 2π which is also he first negative frequency $\Delta\omega - 2\pi$ in the axis $(-\pi, \pi]$.

It does not make much sense to talk about frequencies above π radians. In fact, the frequency $\omega = \pi$ is the Nyquist frequency in rads. What is the meaning of frequencies above $\omega > \pi$?. Well, this simply reflects the way we have discretized the frequency ω when computing the DFT. The DFT discretizes the frequency ω in the interval $[0, 2\pi)$ radians which can also be represented in the interval $(-\pi, \pi]$ as shown in Figure **??**.

4.3 The 2D DFT

The 2D Fourier transform is defined as follows via the following expression

$$F(\omega_1, \omega_2) = \int \int f(x_1, x_2) e^{-i(\omega_1 \, x_1 + \omega_2 \, x_2)} dx_1 \, dx_2 \,, \tag{4.34}$$

similarly, we can define the inversion formula

$$f(x_1, x_2) = \int \int F(\omega_1, \omega_2) e^{i(\omega_1 \, x_1 + \omega_2 \, x_2)} d\omega_1 \, d\omega_2 \,. \tag{4.35}$$

Whereas the 1D FT is used to decomposed signals in a decomposition of sin and cos, one can image the 2D FT as a decomposition of a signal in terms of plane waves. It is important to stress that for our signal processing applications we will be dealing with the 2D DFT (this is the discrete version of the FT). Let us first consider a 2D discrete signal such as a gravity map in discrete form $x_{m,n}$

$$x_{m,n}, \; n = 0, \ldots, N-1, \; m = 0, \ldots, M-1 \,.$$

The formulas for the forward and inverse DFT in the 2D case are given by

$$X_{k,l} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x_{m,n} e^{-i2\pi km/M} e^{-i2\pi ln/N} \,, \; k = 0, \ldots, M, \; l = 0, \ldots, N \,. \tag{4.36}$$

$$x_{k,l} = \frac{1}{N \, M} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} X_{m,n} e^{i2\pi km/M} e^{i2\pi ln/N} \,, \; k = 0, \ldots, M, \; l = 0, \ldots, N \,. \tag{4.37}$$

The 2D DFT is computed by calling 1D DFT along columns and rows of $x_{m,n}$. This is very simple: you first compute the DFT of all the columns of $x_{n,m}$, then you compute the DFT to rows of the previous result. In fact, 2D DFT codes are just 1D FFTs codes working on rows and columns. The 2D DFT is important at the time of filtering 2D images (i.e., gravity maps, seismic records). Notice that in the 2D DFT we need to consider 2D symmetries.

4.4 Frequency domain filtering and finite impulse response filters

So far, we have studied operators (filters) capable of collapsing a wavelet into a spike. These filters are often called spiking filters or Wiener filters. In this section, we will examine the problem of designing frequency domain filters and FIR (Finite Impulse Response) filters. These are filters that are used to eliminate undesired spectral components from our data.

4.4.1 Frequency domain filtering

Assume an input discrete signal x_n, $n = 0, \dots N-1$ with DFT X_k, $k = 0, \dots N-1$. One can filter (exclude) frequency components by multiplying X_k by a filter B_k that eliminate amplitude belonging to frequencies indices k.

Say you have N frequencies and the sampling rate of the signal is Δt and you want to eliminate frequencies between $f = 30$hz and $f = f_{Nyquist}$. How do you compute B_k? First, let's be clear that we don't want to introduce phase distortions, we simple want to multiple X_k by 1 for those frequencies we want to preserve and by 0 those that we want to eliminate. The first problem is how to convert $f = 30$hz to a sample $k = k_c$. It is clear that the Nyquist frequency corresponds to $\omega = \pi$ radians or to $f_{Nyquist} = 1/(2\Delta t)$.

Recall the relationship between ω_k and f_k

$$\omega_k = \frac{2\pi}{N} k$$

$$f_k = \frac{\omega_k}{2\pi\Delta t} = \frac{k}{\Delta t N}\,.$$

Hence you can find the sample corresponding to $f = 30$Hz by doing the following

$$k_c = [30 \times \Delta t \times N]$$

where [.] means round to the nearest integer of the argument. It is clear that for the Nyquist frequency we have

$$k_n = [\frac{1}{2\Delta t} \times \Delta t \times N] = \frac{N}{2}\,.$$

Therefore, we can have the following Operator B_k

$$B_k = 1, \quad k = 0 : k_c - 1$$
$$B_k = 0, \quad k = k_c : N/2$$

Once you have obtained B_k, $k = 0, \dots N/2$ you simple compute $Y_k = X_k.B_k$,$k = 0, \dots N/2$. Then you use the symmetry property of the DFT for real signals to compute Y_k, $k = N/2 + 1 \dots N - 1$. Finally, you use the IDFT of Y_k to compute the filtered signal. If symmetries were properly considered, the output of the IDFT must be a real signal because the input was real.

4.4.2 Low Pass FIR filters

In this case we want to design a filter that operates in the time domain with a amplitude spectrum with the following characteristics

$$B(\omega) = \begin{cases} 1 & -\omega_c \le \omega \le \omega_c \\ 0 & otherwise \end{cases} \tag{4.38}$$

We will assume that the filter phase is zero. In the previous expression ω_c is the cut-off frequency. This filter can be either applied in the frequency domain or in the time domain. It is clear that if the signal to be filtered is called $X(\omega)$, then the filtered signal is given by

$$Y(\omega) = X(\omega)\,.\,F(\omega) \tag{4.39}$$

In general, it is more convenient to design *short* filters in the time domain and applied them via convolution[3]

$$y(t) = x(t) *, b(t) \tag{4.40}$$

[3] note that we are working with continuous signals.

where the sequence b_k is the Impulse Response of the filter with desired amplitude response $B(\omega)$. We can use the inverse Fourier transform to find an expression for $b(t)$,

$$b(t) = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} B(\omega)e^{i\omega t}\,.d\omega \tag{4.41}$$

Evaluating the last integral leads to the following expression for the filter $f(t)$:

$$b(t) = \frac{\omega_c}{\pi}\frac{sin(\omega_c\,t)}{\omega_c\,t} = \frac{\omega_c}{\pi}\,sinc(\omega_c\,t)\,, \qquad \infty < t < \infty\,. \tag{4.42}$$

This is the impulse response of the continues system with amplitude response $B(\omega)$. We need to discretized the previous expression to obtain the impulse response of a discrete system

$$b_n = \Delta t\, b(t)|_{t=n\,\Delta t} \tag{4.43}$$

the factor Δt comes from equation (1.44); this is a scaling factor that allows us to say that the Fourier transform of the discrete and continuous signals are equal in $[-\pi/\Delta t, \pi/\Delta t]$. The final expression of the digital filter is given by

$$b_n = \Delta t\frac{\omega_c}{\pi}\,sinc(\omega_c\,n\Delta t),\ \ n = \ldots, -3, -2, -1, 0, 1, 2, 3\ldots\,. \tag{4.44}$$

It is clear that this is a IIR filter (infinite impulse response filter). A FIR filter is obtained by truncating the IIR filter:

$$b_n = \Delta t\frac{\omega_c}{\pi}\,sinc(\omega_c\,n\delta t),\ \ n = -L\ldots, -3, -2, -1, 0, 1, 2, 3\ldots L\,. \tag{4.45}$$

In this case we have a filter of length $2\,L+1$. When the filter is truncated the actual amplitude spectrum of the filter is not equal to the desired or ideal amplitude spectrum (4.38). This point has already been studied in Chapter 1 where we examined the spectral artifacts that are introduced when a signal is truncated in time. In Figure (4.4) we display the impulse response of a filter of cut-off frequency $f_c = 50Hz$ for filter lengths $(2L+1)$ 21, and 41. We also display the associated amplitude response. It is easy to see that the filter truncation has introduced the so called *Gibbs phenomenon* (Oscillations).

One way to minimize truncation artifacts is by smoothing the truncated impulse response with a taper or window.

$$b_n^w = b_n.w_n$$

now b_n^w is the truncated impulse response after applying a taper function. The taper is used to minimize truncation effects at the end point of the impulse response; a popular taper is the *Hamming Window*

$$w_n = 0.54 - 0.45\, cos(2\pi(n-1)/(N-1))\,, n = 1:N$$

In figure (4.5) we analyze the effect of tapering the impulse response of the filter before computing the amplitude response. It is clear that the oscillations around the transition band have been eliminated. It is important to stress that tapering will also increase the width of the transition band; therefore filters that are too short might not quite reflect the characteristics of the desired amplitude response.

4.4.3 High Pass filters

Knowing how to compute low pass filters allows us to compute high pass filters. If the amplitude response of a low pass filter if given by $B^L(\omega)$ we can design a high pass filter with the same cut-off frequency using the following expression:

$$B^H(\omega) = 1 - B^L(\omega) \tag{4.46}$$

that suggests that one can compute the impulse response of the low pass filter an then transform it into a high pass filter using the following expression

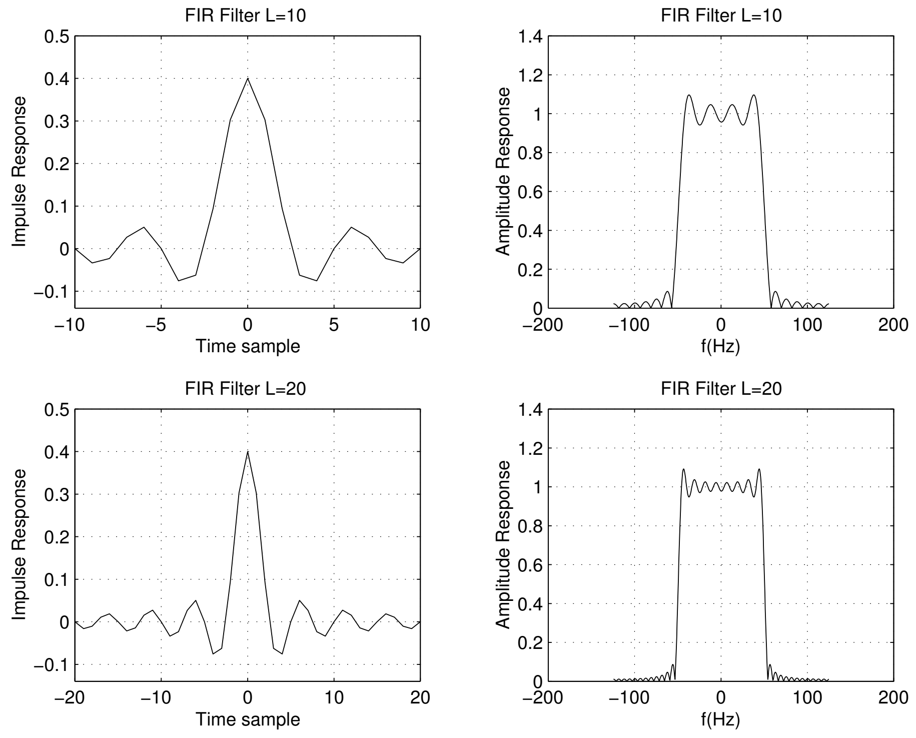


Figure 4.4: Impulse response of two finite length filters and the associated amplitude response. The filter were obtained by truncating the *ideal* infinite length impulse response sequence.

$$\begin{aligned} b_k^H &= \quad -b_k^L \qquad k \neq 0 \\ b_k^H &= \; 1 - b_k^L \qquad k = 0\,. \end{aligned} \tag{4.47}$$

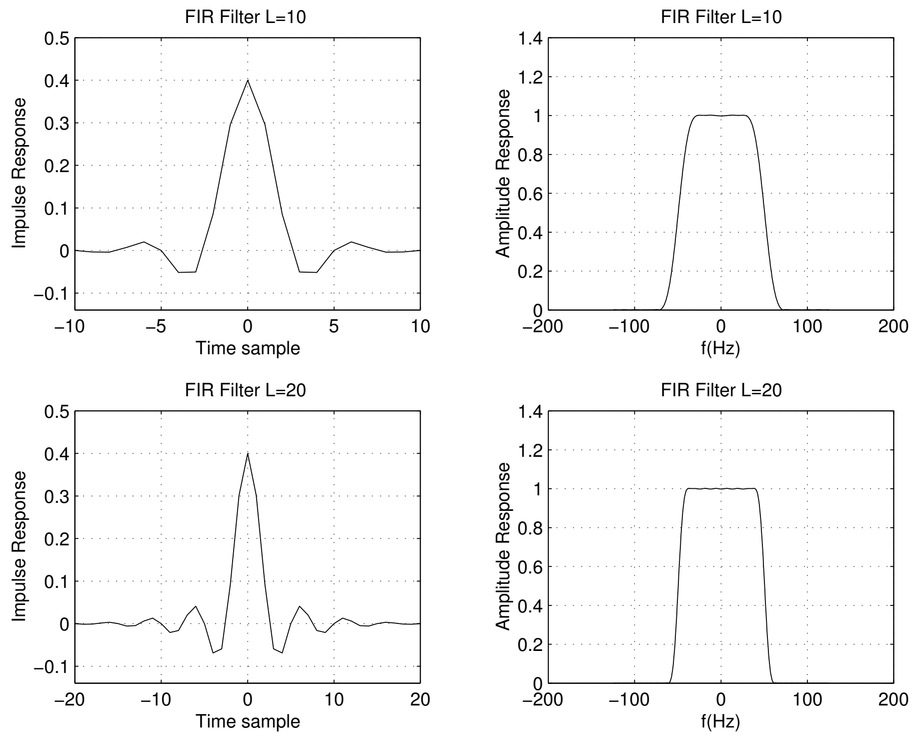


Figure 4.5: Impulse response of two finite length filters and the associated amplitude response. The filter were obtained by truncating the *ideal* infinite length impulse response sequence. In this case the truncated impulse response was *taper* with a Hamming window. Tapering helps to attenuate side-lobe artifacts (Gibbs phenomenon)

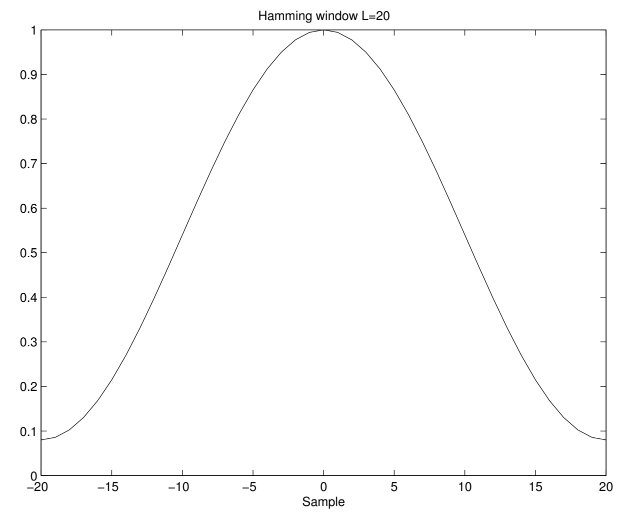


Figure 4.6: A Hamming taper (window) of length $2L + 1$.

Chapter 5

Deconvolution of reflectivity series

5.1 Modeling normal incidence seismograms

In this chapter, we study deconvolution again but now focus on estimating the reflectivity series. First, we present a model that describes reflections associated with plane waves impinging on horizontal layers. The plane wave is impinging at normal incidence and reflected at horizontal geological interfaces. The latter will lead to the so-called convolution model used in applied seismology.

5.1.1 Normal incidence

Consider a plane wave impinging at an angle of propagation $i = 0$ with respect to the normal (see Figure (5.1)). In this case, we have three waves:

- Incident wave: $\downarrow$ in medium 1
- Reflected wave: $\uparrow$ in medium 1
- Transmitted wave: $\downarrow$ in medium 2

Let us assume that the amount of incident wave is equal to 1, the amount of reflected wave is given by r, and the amount of the transmitted wave is denoted by t. At the boundary, the following condition should be satisfied (continuity of displacements)

$$1 + r = t$$

This equation has two unknowns; to compute the r and t, we need an extra equation. We will consider the conservation of energy. In the acoustic (vertical incidence case) conservation of energy leads to the following equation:

$$A_1 \times 1 + A_1 \times r^2 \; = \; A_2 \times t^2 \, .$$

The quantities A_1 and A_2 are called **admittances**

$$A_1 = (\rho_1 \, v_1)^{-1}$$

$$A_2 = (\rho_2 \, v_2)^{-1}$$

where ρ_1 and ρ_2 are the densities of the material above and below the interface and v_1 and v_2 the P-velocities, respectively. The inverse of the admittance is the **Acoustic Impedance**, so $I_1 = A_1^{-1}$ and $I_2 = A_2^{-1}$. After combining the equations of continuity of displacement and conservation of energy, we obtain the following expressions

$$r \; = \; \frac{I_2 - I_1}{I_2 + I_1} \text{Reflection coefficient} \tag{5.1}$$

$$t \; = \; \frac{2 I_1}{I_2 + I_1} \text{Transmission coefficient} \tag{5.2}$$

The above analysis is valid for an incident plane wave propagating downwards (Claerbout, 1976). Let us consider the case of an incident wave propagating upwards (Figure (5.2)).

- Incident wave: $\uparrow$ in medium 2
- Reflected wave: $\downarrow$ in medium 2
- Transmitted wave: $\uparrow$ in medium 1

In this case, the reflection and transmission coefficients are given by

$$r' = \frac{I_1 - I_2}{I_2 + I_1} \tag{5.3}$$

$$t' = \frac{2I_2}{I_2 + I_1} \tag{5.4}$$

From the above equations, it is clear that

$$r' = -r \tag{5.5}$$

5.1.2 Impulse response

Let us assume that we run a zero offset experiment in a stratified earth model composed of four layers plus a half-space of impedances given by I_1, I_2, I_3, I_4 and I_5. (Figure (5.3)). At $t = 0$, a delta-like source emits energy into the earth. The energy is transmitted and reflected from the layers. If we do not consider multiple reflections, our seismogram will have 4 arrivals (4 primary reflections).

To simplify the problem, I will show how to compute the amplitude of the wave recorded at the surface of the earth generated (reflected) at the interface 4. First, we have to compute the amplitude transmitted to each layer until reaching the layer number 4. This is given by the product of the transmission coefficients of each layer. In Figure (5.3) the transmission coefficients t are replaces by their equivalent expression $(1 + r)$.

The amplitude of the wave when it reaches the layer 4 is

$$1 \times t_1 \times t_2 \times t_3 = (1 + r_1)(1 + r_2)(1 + r3)$$

when the wave is reflected in the layer 4, the total amplitude at that point (the expression above) needs to be multiplied by the reflection coefficient of interface 4,

$$1 \times t_1 \times t_2 \times t_3 \times r_4 = (1 + r_1)(1 + r_2)(1 + r3)t_4$$

Note that now the reflected wave is propagating upwards. Therefore, the transmission coefficients are given by

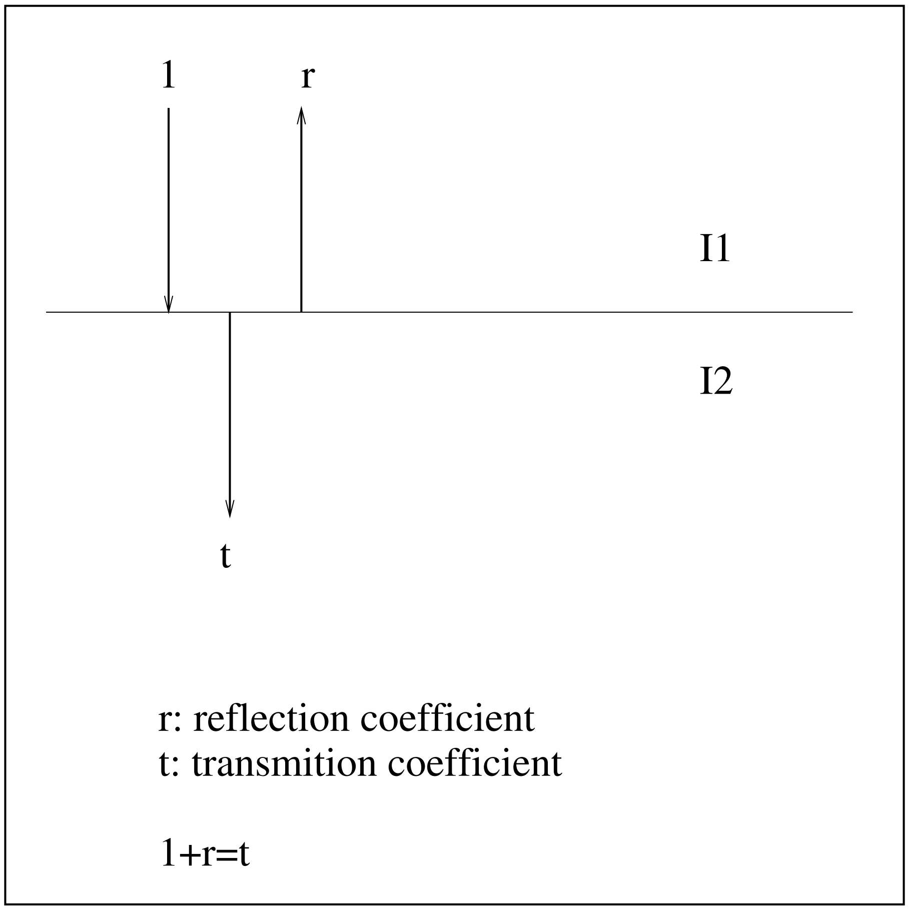


Figure 5.1: P-wave normal incidence. The incident wave propagates downwards.

$$1 + r' = 1 - r$$

The final amplitude after propagating the wave to the surface of the earth (this is what the receiver is measuring!) is given by

$$\underbrace{(1+r_1)(1+r_2)(1+r_3)}_{\text{Transmission} \downarrow} \times \underbrace{r_4}_{\text{Reflection}} \times \underbrace{(1-r_1)(1-r_2)(1-r_3)}_{\text{Transmission} \uparrow}$$

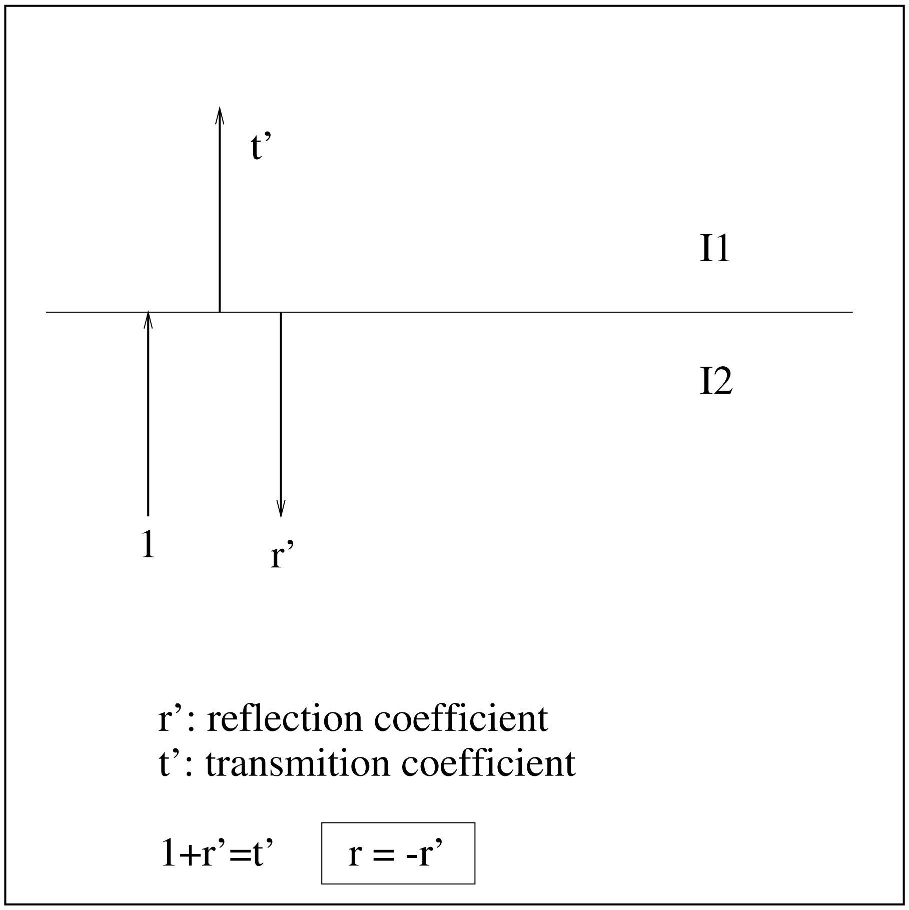


Figure 5.2: P-wave normal incidence. The incident wave propagates upwards.

The final expression for the amplitude of the wave reflected in the interface, 4 can be written down as follows

$$(1-r_1^2)(1-r_2^2)(1-r_3^2)r_4\,.$$

It is clear that reflections occur at all the layers

Amplitude of the reflection generated at the interface 1

$$a_1 = r_1$$

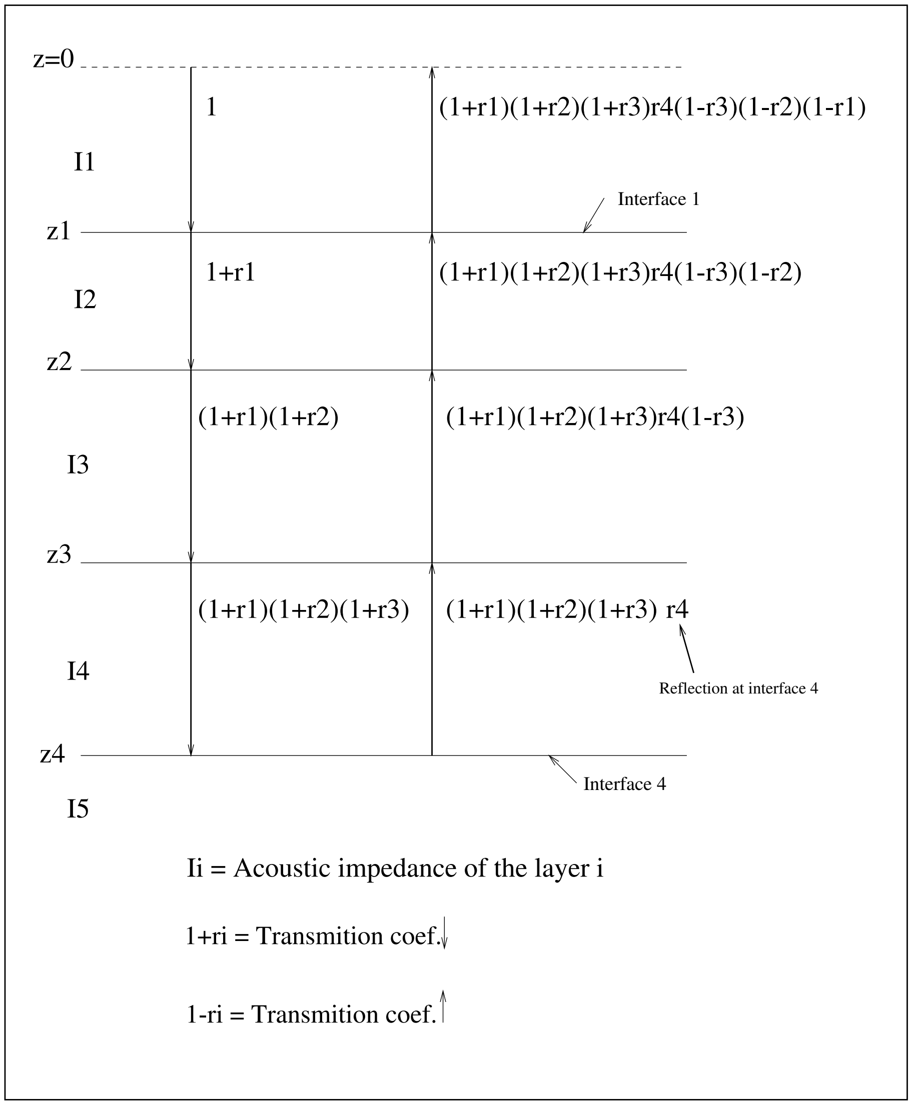

z=0
1
(1+r1)(1+r2)(1+r3)r4(1-r3)(1-r2)(1-r1)
I1
Interface 1
z1
1+r1
(1+r1)(1+r2)(1+r3)r4(1-r3)(1-r2)
I2
z2
(1+r1)(1+r2)
(1+r1)(1+r2)(1+r3)r4(1-r3)
I3
z3
(1+r1)(1+r2)(1+r3)
(1+r1)(1+r2)(1+r3) r4
I4
Reflection at interface 4
z4
Interface 4
I5
Ii = Acoustic impedance of the layer i
1+ri = Transmition coef.↓
1-ri = Transmition coef.↑

Figure 5.3: Amplitude of a wave plane wave propagating in a layered medium. Analysis of the wave reflected in the interface 4.

Amplitude of the reflection generated at the interface 2

$$a_2 = (1 - r_1^2) r_2$$

Amplitude of the reflection generated at the interface 3

$$a_3 \;=\; (1-r_1^2)(1-r_2^2)r_3$$

Amplitude of the reflection generated at the interface 4

$$a_4 \;=\; (1-r_1^2)(1-r_2^2)(1-r_3^2)r_4$$

We can write a general expression for the amplitude of a reflection generated at the k-th interface

$$a_1 \;= r_1$$

$$a_k = \prod_{i=1}^{k-1}(1-r_i^2)\, r_k \approx r_k \;\; k = 2, 3, 4, \ldots$$

How do we interpret these results? If we assume that the earth is excited with a delta function and neglect the presence of multiples, our zero-offset seismogram will be a collection of delta functions (spikes) at arrival times given by the two-way travel time to each interface. The strength of each arrival will be proportional to the amplitude a_k

However, having a source resembling a delta function is impossible in seismic exploration. The source signature is called a *wavelet*. The latter is a finite-length function denoted as $w(t)$. In this case, the seismogram is represented by a superposition of wavelets arriving at times t_k and amplitudes proportional to a.

In our model with 4 interfaces (Figure (5.3)) we will have 4 arrivals of amplitude a_1, a_2, a_3 and a_4. The seismogram can be expressed as follows

$$s(t) = a_1\, w(t-t_1) + a_2\, w(t-t_2) + a_3\, w(t-t_3) + a_4\, w(t-t_4) \qquad (5.6)$$

where t_1, t_2, t_3 and t_4 are the arrival times of each reflection [1]

Notice that if we neglect transmission effects, the amplitude a_i can be replaced by the reflection coefficient of r_i. Furthermore, we will assume

[1]Notice that $w(t-\tau)$ is $w(t)$ after being delayed τ seconds.

an earth model that consists of micro-layers of ”length” Δt. Hence, time becomes discrete and is given by $t = (n-1) * \Delta$, $n = 1, \ldots$. In this case, we can write the seismic trace model as a convolution between two-time series: a wavelet and the reflectivity sequence.

$$s_n = w_n * q_n \,. \tag{5.7}$$

For instance, in our 4-layer example,

$$q = (0,0,0,\ldots,0,r_1,0,0,\ldots,0,0,0,r_2,0,0,\ldots,0,0,r_3,0,0,\ldots,0,0,r_4,0,0,0)$$

Where non-zero reflectivity amplitudes are placed at the corresponding time samples n_i, $t_i = (n_i - 1)\Delta t$, $i = 1 \ldots 4$.

5.2 Deconvolution of reflectivity series

So far, we have discussed the problem of designing a deconvolution operator for a seismic wavelet. We have also examined a *toy problem* involving inverting wavelets of length 2 (dipoles) via series expansion and least-squares inversion.

In general, the convolutional model is a well accepted model to describe a seismic trace. In this model, we say that the seismic trace (zero-offset trace) can be written down as a convolution of two signals: a seismic wavelet (this is the source function) and the reflectivity series.

The reflectivity series is our *geological* unknown. In fact, reflectivity is a sequence of spikes that indicates the time position of the layers in the subsurface. Each spike's strength or amplitude is proportional to how much energy is reflected back to the receivers during the seismic experiment. Let us write the seismogram as a simple convolution between a wavelet w_n and a reflectivity sequence q_n

$$s_n = w_n * q_n \,. \tag{5.8}$$

In this simple model, we have neglected the noise. We will assume that deterministic noise (multiples and ground roll) has been attenuated, and therefore what is left is random noise

$$s_n = w_n * q_n + n_n \,. \tag{5.9}$$

It is clear from the above equation that one has a problem with one equation (one observable) and two unknowns (the wavelet and the reflectivity). Therefore, the seismic deconvolution problem involves the solution of two subproblems:

- **Wavelet Estimation**
- **Inverse filter design**

We refer to methods of estimating the seismic source as wavelet estimation methods. These statistical techniques explode some properties of the remaining unknown (the reflectivity). We also have deterministic processes based on the wave equation that can be adopted to estimate seismic sources in the marine case. These methods are beyond the scope of this course.

5.2.1 The autocorrelation sequence and the white reflectivity assumption

We have seen that the design of a Wiener or inverse filter of the wavelet involves the inversion of an autocorrelation matrix with Toeplitz structure. To clarify the problem, let us assume that we have a 3-point wavelet and we compute its autocorrelation matrix. We first write down the convolution matrix[2]:

$$\mathbf{C} = \begin{pmatrix} w_0 & 0 \\ w_1 & w_0 \\ w_2 & w_1 \\ 0 & w_2 \end{pmatrix} . \tag{5.10}$$

The autocorrelation matrix is given by

[2]This is the matrix you would have used to design a 2-point inverse filter

$$\mathbf{R} = \mathbf{C^T C} = \begin{pmatrix} r_0 & r_1 \\ r_1 & r_0 \end{pmatrix} . \tag{5.11}$$

Now we can try to write the autocorrelation coefficients in terms of the sample of the wavelet w_n. In this case, we have

$$r_0^w = w_0^2 + w_1^2 + w_2^2 \tag{5.12}$$

$$r_1^w = w_0 w_1 + w_1 w_2 \tag{5.13}$$

The first coefficient is the zero-lag correlation coefficient, this is also a measure of the energy of the wavelet. The second coefficient [3] r_1^w is the first lag of the correlation sequence.

The correlation coefficients can be written using the following expression:

$$r_j^w = \sum_k w_k w_{k+j} \,, \quad j = 0, \pm 1, \pm 2 \ldots . \tag{5.14}$$

In the inverse filter the matrix $\mathbf{R}$ is an $N \times N$ a matrix where N is the length of the filter, in this case we will need to compute the autocorrelation coefficients

$$r_j^w, j = 0, N-1 \,.$$

To design the inverse filter, we first need to know the wavelet. Unfortunately, the seismic wavelet is unknown. To solve this problem, we use the white reflectivity assumption. Under this assumption, the seismic reflectivity (the *geology*) is considered a zero-mean white process (Robinson and Treitel, 1980).

A zero-mean white process is an uncorrelated process; in other words if r_j^q is the autocorrelation function of the reflectivity, then

[3]Please, note that the supra-script w is used to stress that this is the autocorrelation of the wavelet

$$r_j^q = \begin{cases} P_q & j = 0 \\ 0 & j = \pm 1, \pm 2, \pm 3, \dots \end{cases} . \tag{5.15}$$

The autocorrelation measures the similarity of a time series with itself. The zero-lag coefficient measures the power of the signal P_q. The first coefficient (lag $j = 1$) measures the similarity of the signal with a one-sample shifted version of itself. If the reflectivity is a zero-mean white noise process, the following remarkable property is true

$$r_j^s = P_q \, r_j^w \, . \tag{5.16}$$

In other words: *the autocorrelation function of the trace is an estimator (within a scale factor) of the autocorrelation of the wavelet.* It is clear that now we can estimate the autocorrelation of the wavelet from the autocorrelation of our observable: the seismic trace.

We have managed to compute the autocorrelation function of the wavelet, but what about the wavelet. The Z-transform of the autocorrelation sequence of the wavelet can be used to calculate the seismic wavelet. In this case, we need to make a new assumption; we will assume that the wavelet is a minimum phase wavelet. Generally, this is a reasonable assumption to deal with sources generated by explosions (dynamite).

It is easy to show that the Z-transform of the autocorrelation the sequence can be decomposed as follows

$$R^w(z) = \sum_j r_j^w z^j = W(z) \, W(z^{-1}) \, . \tag{5.17}$$

The latter is valid for a real wavelet. In this case, the autocorrelation function provides information about the wavelet but cannot define the phase of the wavelet. After factorizing the above equation, one can select the zeros outside the unit circle (the minimum phase dipoles!!). In this way, we can recover the minimum phase wavelet consistent with the given spectrum $R^w(z)$.

Estimating a minimum phase wavelet from the autocorrelation is often called the spectral factorization problem. It can be solved using different techniques:

- Kolmogorov factorization (see my code in SeismicLab)
- Factorization via roots finding. Compute the roots of the Z-transform of the autocorrelation as indicated above, and keep the roots outside the unit circle to synthesize the minimum phase wavelet.
- Double inverse filter.

5.2.2 What do we do with the noise?

We start with our noisy seismogram in the time domain

$$s_n = w_n * q_n + n_n \,. \tag{5.18}$$

The deconvolution process aims to recover q_n from the data, s_n. To achieve this goal, a filter f_k must be computed such that $f_k * w_k = \delta_k$. Generally, of course, we can only compute an estimator of the filter $\hat{f}_k$, where $\hat{f}_k * w_k = a_k$, where a_k is called the averaging function. The latter resembles a delta function only in the ideal case. Applying $\hat{f}_k$ to both sides of equation (5.18), yields the estimated output of the deconvolution process

$$\begin{aligned} \hat{q}_k &= a_k * q_k + \hat{f}_k * n_k \\ &= q_k + (a_k - \delta_k) * q_k + \hat{f}_k * n_k \,. \end{aligned} \tag{5.19}$$

Since our main requirement is to estimate a reliable model $\hat{q}_t$ which is close to the actual reflectivity, it is important to design a filter such that the error terms in equation (5.19) are as small as possible. Or in other words, one seeks a solution with the following properties

$$a_k = w_k * f_k \approx \delta_k \,, \tag{5.20}$$

and

$$f_k * n_k \approx 0\,. \tag{5.21}$$

The last two expressions can also be written in matrix form

$$\mathbf{C}_w \mathbf{f} \approx \mathbf{d} \tag{5.22}$$

and

$$\mathbf{C}_n \mathbf{f} \approx \mathbf{0} \tag{5.23}$$

where $\mathbf{C}_w$ and $\mathbf{C}_n$ denote the convolution matrices for the wavelet and the noise, respectively. Both equations are honoured when we minimize the following objective function

$$J = ||\mathbf{C}_w \mathbf{f} - \mathbf{d}||^2 + \beta ||\mathbf{C}_n \mathbf{f}||^2\,, \tag{5.24}$$

where β is a tradeoff parameter. The second term in the last equation can be written as

$$||\mathbf{C}_n \mathbf{f}||^2 = \mathbf{f}^T \mathbf{C}_n^T \mathbf{C}_n \mathbf{f}\,, \tag{5.25}$$

where the matrix $\mathbf{C}_n^T \mathbf{C}_n$ is the noise autocorrelation matrix. If the noise is uncorrelated, we can replace $\mathbf{C}_n^T \mathbf{C}_n$ by

$$E[\mathbf{C}_n^T \mathbf{C}_n] = \sigma_n^2 \mathbf{I}\,. \tag{5.26}$$

where σ_n^2 is the variance of the noise. Now the objective function J is given by,

$$J = ||\mathbf{C}_w \mathbf{f} - \mathbf{d}||^2 + \mu ||\mathbf{f}||^2\,, \tag{5.27}$$

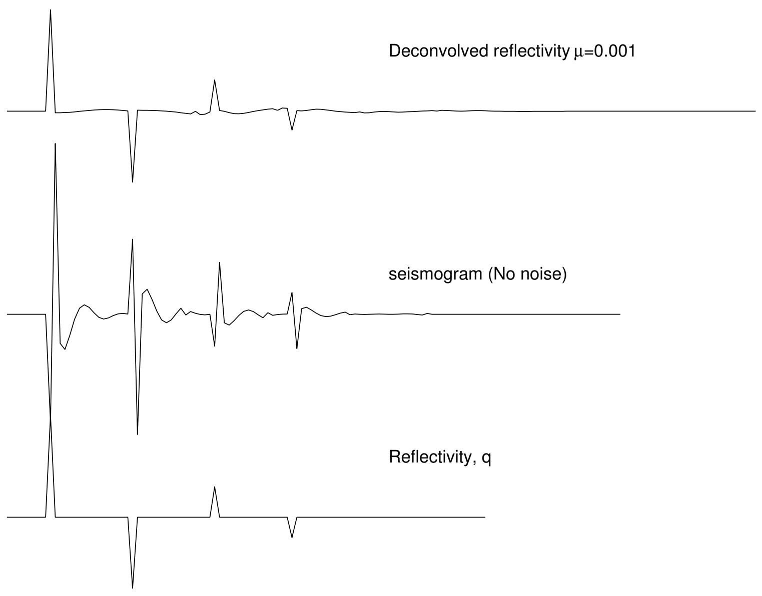


Figure 5.4: Deconvolution of a *clean* seismogram.

where $\mu = \sigma_n^2 \times \beta$. This is the objective function used to design the inverse filter, and the solution is given by

$$\mathbf{f} = (\mathbf{R}_w + \mu \mathbf{I})^{-1} \mathbf{C}_w^T \mathbf{d}\,. \qquad (5.28)$$

In Figures (5.4), (5.5) and (5.6) we test the performance of the least-squares inversion when dealing with noise-free and noisy data. It is clear that the pre-whitening parameter plays a crucial role in the deconvolution of noisy data.

5.2.3 Deconvolution in the frequency domain

A procedure similar to the one outlined in the previous section can be used to deconvolve data in the frequency domain. Taking the Discrete Fourier Transform (DFT) of equation (5.19) yields

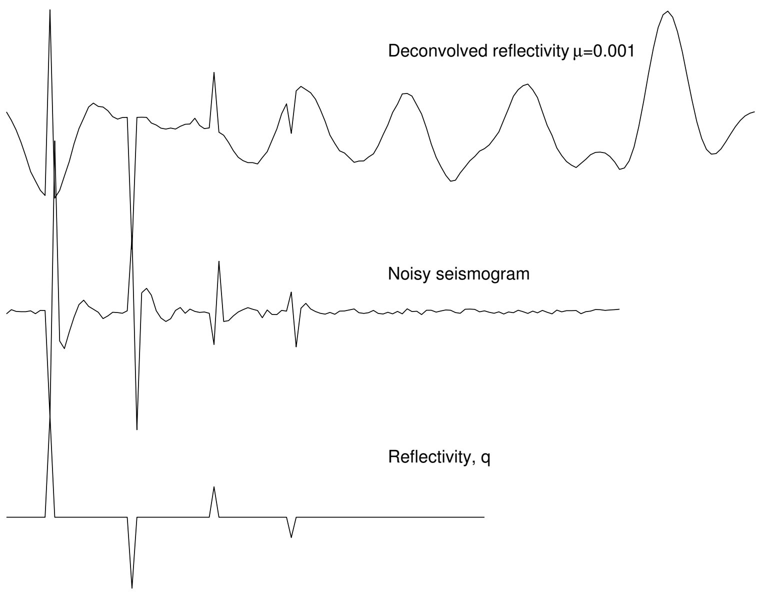


Figure 5.5: Deconvolution of a *noisy* seismogram. The tradeoff parameter is too small, then the resulting solution is unstable.

$$\hat{Q}_k = Q_k + (A_k - 1)Q_k + \hat{F}_k N_k . \tag{5.29}$$

Since a_k should be a good approximation to a delta function, it turns out that the filter should be designed to satisfy the following requirement

$$W_K F_k = A_k \approx 1 \qquad \forall k. \tag{5.30}$$

Furthermore, to maintain the noise at a small level, we also wish to minimize the

$$F_k \, N_k \approx 0 \qquad \forall k \, . \tag{5.31}$$

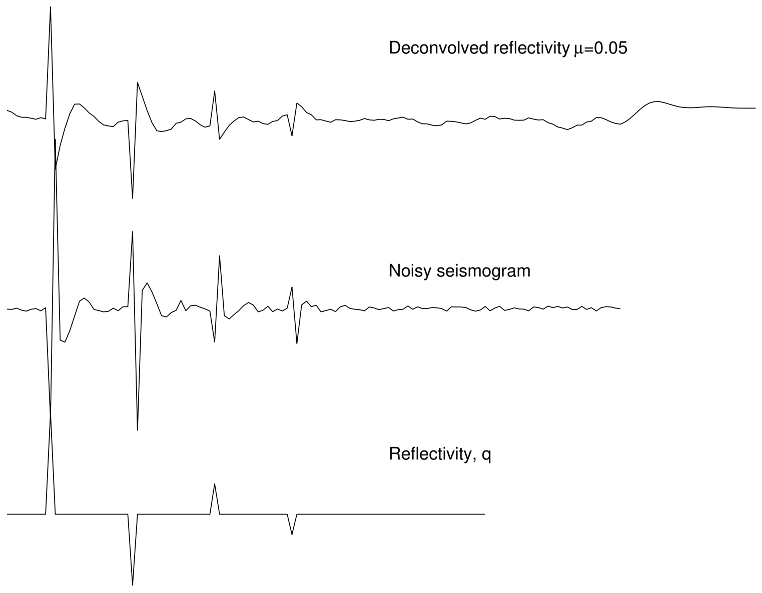


Figure 5.6: Deconvolution of a seismogram contaminated by additive noise. The tradeoff term has been adopted to stabilize the solution.

We can combine these two requirements into a single one. To achieve this, let us construct the following objective or cost function

$$J = \sum_k |A_k - 1|^2 + \alpha \sum_k |F_k N_k|^2. \tag{5.32}$$

Minimizing the objective function with respect to the filter coefficients leads to

$$\hat{F}_k = \frac{W_k^*}{|W_k|^2 + \alpha |N_k|^2}. \tag{5.33}$$

Finally, the reflectivity estimate is given by

$$\begin{aligned} \hat{Q}_k &= D_k \tfrac{W_k^*}{|W_k|^2+\alpha|N_k|^2} \\ &= D_k \tfrac{W_k^*}{|W_k|^2+\mu} \,. \end{aligned} \tag{5.34}$$

Since the noise has a flat spectrum ($|N_k|^2 = \sigma_n^2$) we can replace $\alpha|N_k|^2$ by another constant μ. An estimate of the variance of the reflectivity estimator in the frequency domain is given by

$$Var(\hat{Q}_k) = |\hat{F}_k|^2 {\sigma_N}^2. \tag{5.35}$$

after a few manipulations, we end up with

$$Var(\hat{Q}_k) = \frac{|W_k|^2 {\sigma_N}^2}{(|W_k|^2+\mu)^2}\,. \tag{5.36}$$

When $\mu = 0$ the variance $Var(\hat{Q}_k)$ can be too high at the frequencies k at which the wavelet power is small. Similarly, we can find an expression for the norm of the reflectivity estimator in the frequency domain

$$\begin{aligned} N = &\ \textstyle\sum_k |\hat{Q}_k|^2 \\ = &\ \tfrac{1}{{\sigma_N}^2} \textstyle\sum_k |S_k|^2 Var(\hat{Q}_k) \end{aligned} \tag{5.37}$$

The misfit function is

$$\begin{aligned} \Phi &= \textstyle\sum_k |S_k - W_k\hat{Q}_k|^2 \\ &= \tfrac{1}{{\sigma_N}^2} \textstyle\sum_k |S_k|^2 (\tfrac{\mu}{|W_k|^2+\mu})^2 \end{aligned} \tag{5.38}$$

Regularization error and noise magnification

If E_k denotes the deviation of the filter from the right inverse filter, defined by

$$E_k = 1 - \hat{F}_k W_k \tag{5.39}$$

we can write equation 5.34 as follows

$$\begin{aligned}\hat{Q}_k &= \hat{F}_k W_k Q_k + \hat{F}_k N_k \\ &= (1-E_k)Q_k + \tfrac{1-E_k}{W_k} N_k\end{aligned} \tag{5.40}$$

then, the difference between the actual reflectivity Q_k and the reflectivity estimate $\hat{Q}_k$ is given by

$$\hat{Q}_k - Q_k = \underbrace{-E_k Q_k}_{RE} + \underbrace{\frac{1-E_k}{W_k} N_k}_{NAE} \tag{5.41}$$

where RE stands for regularization error and NAE for noise amplification error. The NAE is independent of the data and can be expressed as a function of the wavelet:

$$\frac{W_k^*}{|W_k|^2 + \mu}\,.$$

It is clear that the more the filter resembles the inverse the wavelet W_k^{-1}, the larger this error will be. The RE introduces data-dependent degradation (i.e., ringing).

5.2.4 References

Robinson E. A. and S. Treitel, Geophysical Signal Analysis, 1980, Prentice Hall.

Clearbout, J., Fundamentals of Geophysical Data Processing, 1976, McGraw-Hill.

Chapter 6

Signal-to-noise-ratio enhancement via $f - x$ deconvolution

6.1 $f - x$ filters

Signal-to-noise-ratio enhancement in the $f - x$ (also called $\omega - x$) domain has been proposed by Canales (1994) as a method for random noise attenuation. The technique is widely accepted and used to process prestack and poststack seismic data from a reflection seismology experiment. The technique is instrumental in attenuating random noise; it is easy to implement and efficient in the computational sense. You will find this method often named $f - x$ deconvolution or $f - x$ decon.

Before developing the theory of $f - x$ deconvolution filters, a few background sentences are in order. Signal predictability has been extensively studied in the context of AR (Autoregressive) filters and harmonic retrieval via ARMA (Autoregressive Moving Average) models (see, for instance, Ulrych and Clayton, 1976). In general, AR and ARMA models are adopted for parametric spectral analysis. In this chapter, we will use them for the prediction of the seismic signal is space.

The idea is quite simple and can be summarized as follows. In the $f - x$ domain, events with linear moveout or quasilinear events manifest themselves as a superposition of harmonics. You have already studied in GEOPH 326 that reflections in common-mid-point (CMP) and common-shot gathers (CSG) have hyperbolic moveout. However, if you consider a small spatio-temporal window of seismic data in any domain (including CSG and CMP gathers), one can approximate seismic events by a superposition of constant dip signals, or in other words, events with linear moveout. Hence, $f-x$ filters are often run in overlapping spatio-temporal windows of seismic data to validate the linear moveout assumption under which these filters are designed.

If noise is taken into account, an optimal model to predict a superposition of harmonics is an ARMA model (AR: Autoregressive, MA:Moving average). However, given the the fact that ARMA models might not be very stable (they involve the solution of an eigenvalues problem), we will propose to replace the ARMA model with a long AR filter. In this case, the predictability is not optimal, but the problem can be easily solved using predictor error filters of the type, we have already analyzed in the context of deconvolution.

We will start this lecture by introducing the concept of predictability via a straightforward model composed of a superposition of harmonics in x.

6.1.1 The signal model

The signal model is based on the assumption that seismic data can be represented as a superposition of events with linear moveout. In general, a seismic section can be divided into overlapping windows where this assumption is valid.

One single event (dip)

We first consider a seismic section that consists of a single dip or single event with linear moveout

$$s(t,x) = w(t - t_0 - px)\,. \tag{6.1}$$

It is clear that the signal $s(t,x)$ represent an event with linear moveout in x because the wavelet of amplitude $w(t)$ is delayed an amount $t_0 + p\,x$ where p

is the dip (or local ray parameter) and x is distance. The frequency domain representation of $s(t,x)$ is given by[1]

$$S(f,x) = W(f)\, e^{-i2\pi f t_0} e^{-i2\pi f p x}\,, \tag{6.2}$$

We can simplify last equation by absorbing all terms that do not depend on space x into a single term

$$S(f,x) = A(f)\, e^{-i2\pi f p x}\,, \tag{6.3}$$

where $W(f)$ indicates the source spectrum, and $A(f) = W(f)e^{-i2\pi f t_0}$, with f the temporal frequency, x the spatial variable or offset and p the apparent slowness or dip. We will assume that the spatial variable x is regularly discretized according to $x = (k-1)\Delta x$, $k = 1, \dots N$ where N is the number of traces in the window of analysis. For any temporal frequency, f, we can write

$$S_n = A\, e^{-i\alpha n}, \qquad n = 1, \dots N \tag{6.4}$$

where $\alpha = 2\pi f p \Delta x$. The following recursion is obtained by combining S_n and S_{n-1}

$$S_n = a_1 S_{n-1}\,. \tag{6.5}$$

where $a_1 = exp(i\alpha)$. The last equation is a first order difference equation that allows us to recursively predict the signal along the spatial variable $x = (n-1)\,\Delta x$.

Superposition of L linear events

Now we assume a superposition of L events of different dip

$$s(t,x) = \sum_{k=1}^{L} \alpha_k\, w(t - t_{0k} - p_k\, x)\,. \tag{6.6}$$

In this case, we are also assuming that each events has an amplitude given by α_k. Last expression can be transformed to the $f-x$ domain

[1]Remember the time delay theorem: $f(t-\tau) \leftrightarrow F(\omega)^{-i\omega\tau}$.

$$S(f,x) = \sum_{k=1}^{L} \alpha_k W(f)\, e^{-i2\pi f t_{0k}} e^{-i2\pi f p_k\, x}\,, \tag{6.7}$$

where the later correspond to the superposition of L complex harmonics in x for a given frequency f. Similarly to the case where we analyze one dip ($L=1$), it can be shown that the superposition of L complex harmonics (L linear events in $x-t$) can be represented by a difference equation of order L

$$S_n = a_1 S_{n-1} + a_2 S_{n-2} + \dots a_p S_{n-L}\,. \tag{6.8}$$

where the coefficients $a_1, a_2 \dots a_L$ are related to the dip $p_1, p_2, p_3 \dots p_L$. The last equation can be written in prediction error form as follows

$$\sum_{k=0}^{L} g_k S_{n-k} = 0\,, \tag{6.9}$$

where the coefficients of the prediction error filter are related to the coefficients a_k in equation (4) by the following expressions

$$g_0 = 1, \quad g_k = -a_k, \quad k = 1, \dots L\,.$$

In the absence of noise, the prediction error filter g_k is a signal annihilator. In other words, the convolution of the g_k with S_k as expressed in equation 7.19 is equal to zero (annihilates the output). So far, we have been able to define a recursive expression to predict a noise-free superposition of complex harmonics. In real applications, however, additive noise will corrupt the data, hence we write

$$Y_n = S_n + E_n\,, \tag{6.10}$$

where E_n represents a white noise sequence and Y_n is the spatial signal corrupted by noise (Y_k is the measured signal in $f-x$ domain). The noise E_n is considered white noise in space. Substituting $S_{n-k} = Y_{n-k} - E_{n-k}$ into equation (7.19) leads to the following system of equations that defines the signal model in terms of the prediction error filter

$$\begin{aligned} \textstyle\sum_{k=0}^{L} g_k Y_{n-k} &= \textstyle\sum_{k=0}^{L} g_k E_{n-k} \\ &= e_n\,. \end{aligned} \tag{6.11}$$

The latter is model of the type called ARMA(L,L) in which the AR and MA components are identical. The signal e_n in equation (6.11) designates the non-white innovation sequence $\sum_{k=0}^{p} g_k E_{n-k}$. It is non-white because it the output of filtering g_k with white noise E_k.

6.1.2 $f - x$ deconvolution derivation via the AR model

Rather than trying to solve the ARMA equations, one can replace the ARMA model with a long AR (autoregressive) model. In other, words we simplify equation 6.11 into

$$Y_n - a_1 Y_{n-1} + a_2 Y_{n-2} \ldots + a_p Y_{n-L} = N_n \,. \tag{6.12}$$

Where a_k are the coefficients of the AR(L) model. In general, L is not equal to the number of dips in the data. The latter was valid for the ARMA model. The parameter L is the model's order also called the length of the $f - x$ decon filter which should be large enough to represent the original ARMA model. In other words, the expression given by 6.11 where the white noise E_n was convolved by the filter is simplified by assuming that the term $\sum_{k=0}^{p} g_k E_{n-k}$ is equivalent to a white noise innovation N_n. The determination of L involves using trial-and-error methods where we examine the output and adjust the filter to avoid signal leakage or by criteria based on statistical assumptions such the AIC method (Automatic information criterion also is known as Akaike information criterion).
The last equation can be written in matrix form as follows (assume $L = 3$),

$$\begin{pmatrix} y_1 & 0 & 0 \\ Y_2 & Y_1 & 0 \\ Y_3 & Y_2 & Y_1 \\ Y_4 & Y_3 & Y_2 \\ 0 & Y_4 & Y_3 \\ 0 & 0 & Y_4 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} - \begin{pmatrix} y_2 \\ Y_3 \\ Y_4 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} w_2 \\ N_3 \\ N_4 \\ 0 \\ 0 \end{pmatrix} \,. \tag{6.13}$$

We wave already seen a similar equation in the previous chapter when we analyze inverse filters. The last equation can be written as

$$\mathbf{Y}\mathbf{a} - \mathbf{d} = \mathbf{n} \tag{6.14}$$

The least-squares filter $\mathbf{a}$ is computed by minimizing the power of the innovation noise $\mathbf{N}$, in this case our cost or objective function is given by

$$J = ||\mathbf{Y}\,\mathbf{a} - \mathbf{d}||_2^2\,. \tag{6.15}$$

Taking derivatives of the cost function with respect to filter coefficients and equating the result to zero leads to

$$\mathbf{Y}^H\,\mathbf{Y}\,\mathbf{f} \;=\; \mathbf{Y}^H\,\mathbf{d} \tag{6.16}$$

Note that the matrix $\mathbf{Y}^H\,\mathbf{Y}$ is a Toeplitz form which can be efficiently solved using Levinson's recursion. Then, the estimated filter $\hat{\mathbf{a}}$ is given by

$$\hat{\mathbf{a}} \;=\; (\mathbf{Y}^H\,\mathbf{Y})^{-1}\,\mathbf{Y}^H\,\mathbf{d}\,. \tag{6.17}$$

Once the filter has been estimated, we apply it to the data vector $\mathbf{d}$ to obtain the *clean* data vector $\hat{\mathbf{d}}$

$$\hat{\mathbf{d}} \;=\; \mathbf{Y}\,\hat{\mathbf{a}}\,. \tag{6.18}$$

The estimate $\hat{\mathbf{d}}$ is the predicted data, the predicted noise sequence is given by

$$\hat{\mathbf{n}} \;=\; \hat{\mathbf{d}} \;-\; \mathbf{d}\,. \tag{6.19}$$

In general we need to regularize the filter by adding a small perturbation to the diagonal of the Toeplitz matrix,

$$\hat{\mathbf{f}} \;=\; (\mathbf{Y}^H\,\mathbf{Y} \;+\; \mu\mathbf{I})^{-1}\,\mathbf{Y}^H\,\mathbf{d}\,. \tag{6.20}$$

The process is called $f - x$ deconvolution because one can make an analogy with the deconvolution process.

6.1.3 The convolution matrix

We have adopted a convolution matrix to design our filter. But bear in mind that other data matrices can be used to estimate the data prediction filter. Canales (1984) original formulation uses the following model,

$$\begin{pmatrix} Y_1 & 0 & 0 \\ Y_2 & Y_1 & 0 \\ Y_3 & Y_2 & Y_1 \\ Y_4 & Y_3 & Y_2 \\ 0 & Y_4 & Y_3 \\ 0 & 0 & Y_4 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} Y_2 \\ Y_3 \\ Y_4 \\ 0 \\ 0 \end{pmatrix} . \tag{6.21}$$

Ulrych and Clayton (1976) proposed the transient-free convolution matrix. This is a matrix where zero extension is avoided. In other, words only available data is used to estimate the filter. In Canales' method missing data is treated as zeros. In our simple example ($L = 3$) the transient-free matrix formulation is given by

$$\begin{pmatrix} Y_3 & Y_2 & Y_1 \\ Y_4 & Y_3 & Y_2 \\ Y_5 & Y_4 & Y_3 \\ Y_6 & Y_5 & y_4 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} Y_4 \\ Y_5 \\ Y_6 \\ Y_7 \end{pmatrix} . \tag{6.22}$$

The solution of the above system gives a filter-free of truncation errors. However, the matrix $\mathbf{Y}^H \mathbf{Y}$ is not a Toepiltz form.

It is essential to mention that the above analysis only involved forward prediction filters. In other words we are trying to predict the future samples of the signal based upon past values of the signal. Now, talking about future doe snot make sense because we are working with spatial signal. By future I meant prediction Y_k from $Y_{k-1}, Y_{k-2}, \ldots$. A correct name for the latter is forward prediction. Similarly, one could have written equations where one can do backward prediction which means prediction Y_k from $Y_{k+1}, Y_{k+2}, \ldots$. In space, once the data has been acquired, one can write forward prediction and backward predcition equations. A more sophisticated scheme involves the simultaneous minimization of a forward and a backward prediction error. In this case, one can show that the system of equations in the transient-free matrix has the following aspect (we assume a filter of length L and a signal composed of N samples)

$$\begin{pmatrix} Y_L & \ldots & Y_1 \\ . & \ldots & . \\ . & \ldots & . \\ . & \ldots & . \\ Y_{N-1} & \ldots & Y_{N-L} \\ Y_2^* & \ldots & Y_{L+1}^* \\ . & \ldots & . \\ . & \ldots & . \\ . & \ldots & . \\ Y_{N-L+1}^* & \ldots & .Y_N^* \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ . \\ . \\ . \\ a_L \end{pmatrix} = \begin{pmatrix} Y_{L+1} \\ . \\ . \\ . \\ Y_N \\ Y_1^* \\ . \\ . \\ . \\ Y_{N-L}^* \end{pmatrix} . \tag{6.23}$$

6.1.4 Examples

In Figures 6.1 and 6.2 two synthetic windows where we examine the predictability of a single harmonic.

Two dimensional simulations are displayed in Figures 6.3 and 6.4. Notice that in spite the method being developed for signal with linear moveout, it can also be used for waveforms where the curvature of event is varying slowly as it is shown in Figure 6.4.

The algorithm to perform the $f-x$ noise attenuation is summarized as follows:

1. Transform the data into the $f - x$ domain

$$Data(t,x) \rightarrow Data(f,x)$$

2. for each frequency f solve the AR prediction problem outline in the preceding section to estimate the AR prediction filter.

3. Apply the filter to the data (convolution of the filter witht the data).

4. Transform back to $t - x$ domain

$$Data(f,x) \rightarrow Data(t,x)$$

5. end

In you are interested in knowing more about $f - x$ deconvolution, there is a nice review paper by Gülünay (2017) that you should read.

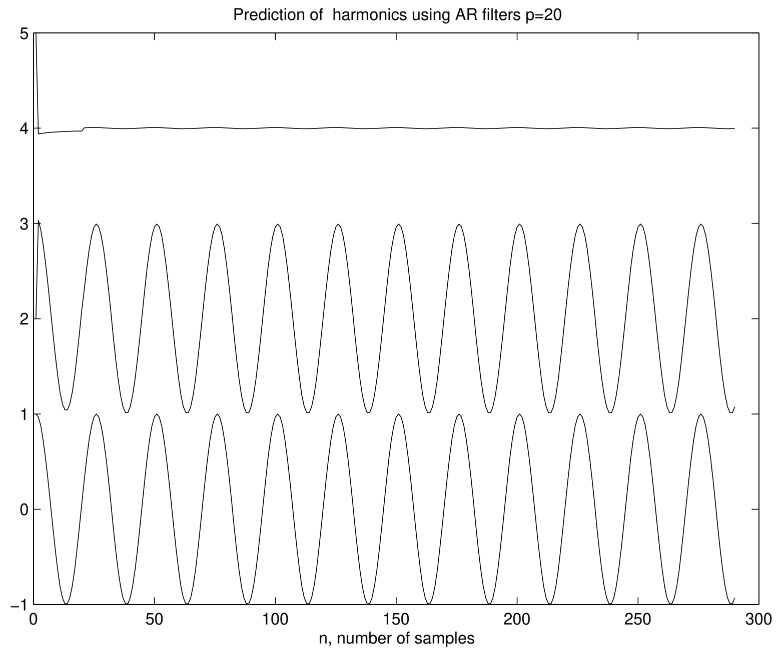


Figure 6.1: Prediction of a single harmonic (no noise) using AR filters. Bottom figure is the observed signal from where we compute the prediction filer a_k of length L. In this case $L = 2$ because a real sin requires two exponentials. Central signal is the predicted signal from the filer coefficients a_k. The signal at the top is the prediction error. Clearly the first sample of the signal cannot be predicted.

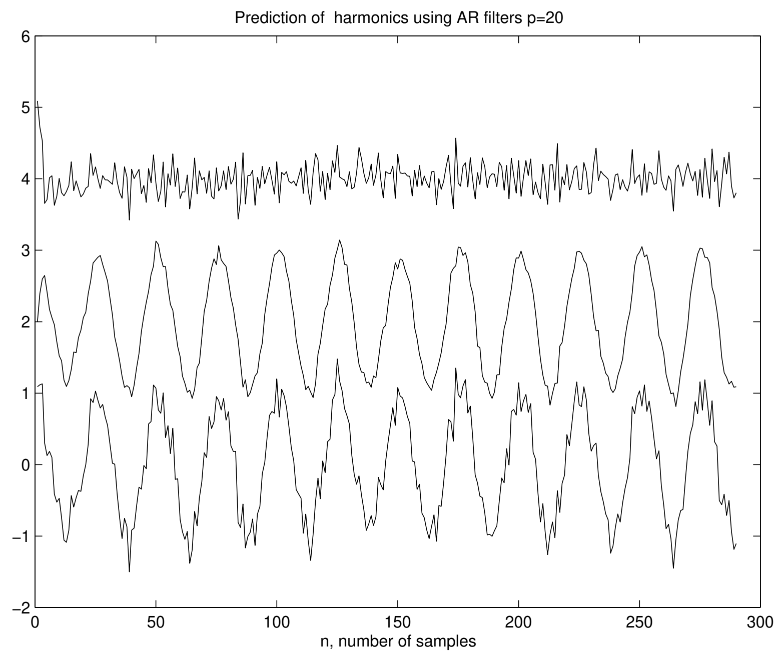


Figure 6.2: Prediction of a single harmonic ($\sigma_{noise} = 0.2$) using AR filters (linear prediction). The signal containing noise (Bottom signal) was used to estimate the prediction filter which is used to estimate the central signal. You can observe the noise has been partially removed. Finaly, the top signal is the predicted error or predicted noise which is the noise that has been extracted from the observed signal.

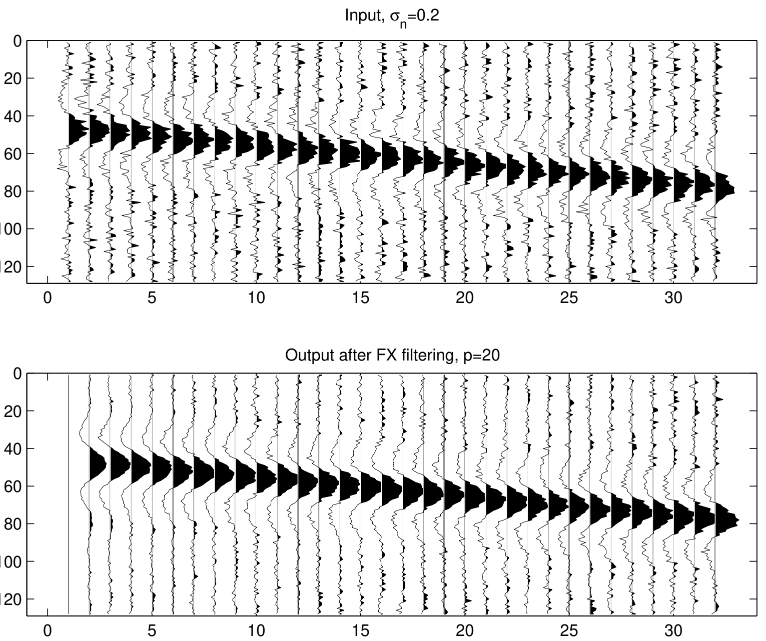


Figure 6.3: FX filtering of a single linear event immersed in noise.

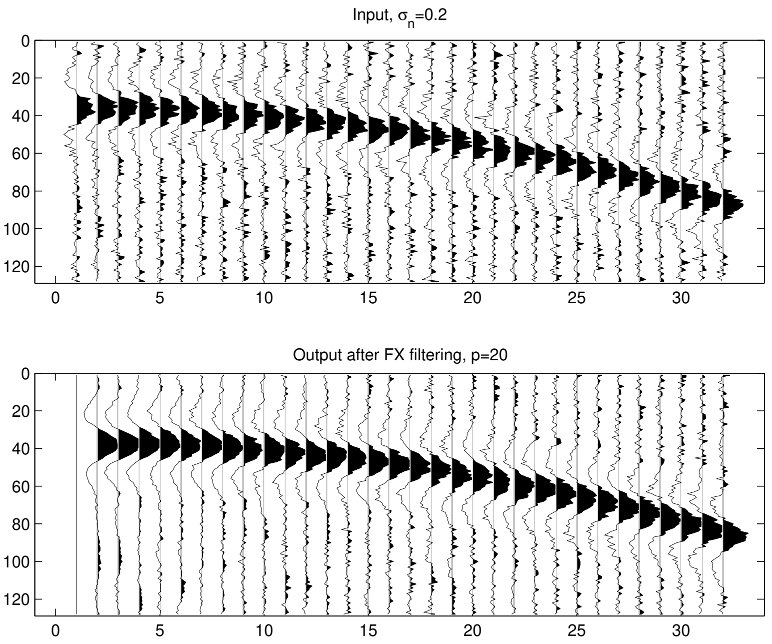


Figure 6.4: $f - x$ filtering of a single linear hyperbolic event immersed in noise. Notice that the curvature of the hyperbolic event is relatively small and, hence, the $f-x$ filter can model the signal and produce an output with attenuated noise.

6.1.5 References

Akaike, H., 1974, A new look at statistical model information, IEEE Trans. Auto. Control, **AC-19**, 716-723.

Canales, L. , 1984, Randon noise reduction: 54th Annual SEG Meeting, p.525-527.

Fahlman, G. G., and Ulrych, T. J., 1982, A new method of estimating the power spectrum of gapped data: Mon. Not. Roy. Astr. Soc., v.199, p.53-65.

Gülünay N, 2017, Signal leakage in $f-x$ deconvolution algorithms: Geophysics 82 (5), W31-W45.

Ulrych, T. J., and Clayton, R. W., 1976, Time series modeling and maximum entropy: Phys. of the Earth and Plan. Int., **12**, 188-200.

Walker, C., and Ulrych, T. J., 1983, Autoregressive recovery of the acoustic impedance: Geophysics, **48**, 1338-1350.

Chapter 7

The KL transform and eigenimages

In this chapter, we will discuss another technique to improve the information content of seismic data. The application of eigenimage analysis in seismology was proposed by Hemon and Mace (1978). In their approach, they use a particular linear transformation called the Karhunen-Loeè (KL) transformation. The KL transformation is also knows as the principal component transformation, the eigenvector transfomation or the Hotelling transofmation. Of particular relevance to the ensuing discussion is the excellent paper by Ready and Vintz (1973), which deals with information extraction and SNR improvement in multispectral imagery.

In 1983, the work of Hemon and Mace was extended by a group of researchers at the University of British Columbia in Canada which culminated in the work of Jones and Levy (1987).

In 1988 Freire and Ulrych applied the KL transformation in a somewhat different manner to the processing of vertical seismic profiling data. The actual approach which was adopted in this work was by means of singular value decomposition (SVD), which is another way of viewing the KL transformation (the relationship between the KL and SVD transformations is discussed in this chapter).

A seismic section which consists of M traces with N points per trace may be viewed as a data matrix $\mathbf{X}$ where each element x_{ij} represents the

i^{th} point of the j^{th} trace. A singular value decomposition (Lanczos, 1961), decomposes $\mathbf{X}$ into a weighted sum of orthogonal rank one matrices which have been designated by Andrews and Hunt (1977) as eigenimages of $\mathbf{X}$. A particularly useful aspect of the eigenimage decomposition is its application in the complex form. In this instance, if each trace is transformed into the analytic form, then the eigenimage processing of the complex data matrix allows both time and phase shifts to be considered which is of particular importance in the case of the correction of residual statics.

7.1 Mathematical framework

We consider the data matrix $\mathbf{X}$ to be composed of M traces with N data points per trace, the M traces forming the rows of $\mathbf{X}$. The SVD of $\mathbf{X}$ is given by, (Lanczos (1961)),

$$\mathbf{X} = \sum_{\mathbf{i=1}}^{\mathbf{r}} \sigma_{\mathbf{i}} \mathbf{u_i} \mathbf{v_i^T}. \tag{7.1}$$

where ^{T} indicates transpose, r is the rank of $\mathbf{X}$, $\mathbf{u}_i$ is the ith eigenvector of $\mathbf{XX^T}$, $\mathbf{v}_i$ is the ith eigenvector of $\mathbf{X^TX}$ and σ_i is the ith singular value of $\mathbf{X}$. The singular values σ_i can be shown to be the positive square roots of the eigenvalues of the matrices $\mathbf{XX^T}$ and $\mathbf{X^TX}$. These eigenvalues are always positive owing to the positive definite nature of the matrices $\mathbf{XX^T}$ and $\mathbf{X^TX}$. In matrix form equation (7.1) is written as

$$\mathbf{X} = \mathbf{U\Sigma V^T} \tag{7.2}$$

Andrews and Hunt (1977) designate the outer dot product $\mathbf{u}_i\mathbf{v}_i^T$ as the ith eigenimage of the matrix $\mathbf{X}$. Owing to the orthonormality of the eigenvectors, the eigenimages form an orthonormal basis which may be used to reconstruct $\mathbf{X}$ according to equation (7.1).

Suppose, for example, that $\mathbf{X}$ represents a seismic section and that all M traces are linearly independent. In this case $\mathbf{X}$ is of full rank M, all the σ_i are different from zero and a perfect reconstruction of $\mathbf{X}$ requires all eigenimages. On the other hand, in the case where all M traces are equal to within a scale factor, all traces are linearly dependent, $\mathbf{X}$ is of rank one and may be perfectly reconstructed by the first eigenimage $\sigma_1 \mathbf{u_1 v_1^T}$. In the general case, depending on the linear dependence which exists among the traces, $\mathbf{X}$ may be reconstructed from only the first few eigenimages. In this case, the data may be considered to be composed of traces which show a high degree of trace-to-trace correlation. Indeed, $\mathbf{XX^T}$ is, of course, a weighted estimate of the zero lag covariance matrix of the data $\mathbf{X}$ and the structure of this covariance matrix, particularly the distribution of the magnitudes of the corresponding eigenvalues, indicates the parsimony or otherwise of the eigenimage decomposition. If only $p, p < r$, eigenimages are used to approximate $\mathbf{X}$, a reconstruction error ϵ is given by

$$\epsilon = \sum_{k=p+1}^{r} \sigma_k^2 \,. \tag{7.3}$$

Freire and Ulrych (1988) defined band-pass $\mathbf{X_{BP}}$, low-pass $\mathbf{X_{LP}}$ and high-pass $\mathbf{X_{HP}}$ eigenimages in terms of the ranges of singular values used. The band-pass image is reconstructed by rejecting highly correlated as well as highly uncorrelated traces and is given by

$$\mathbf{X_{BP}} = \sum_{\mathbf{i=p}}^{\mathbf{q}} \sigma_\mathbf{i} \mathbf{u_i v_i^T} \,, \qquad \mathbf{1 < p \leq q < r} \,. \tag{7.4}$$

The summation for X_{LP} is from $i = 1$ to $p-1$ and for $\mathbf{X_{HP}}$ from $i = q+1$ to r. It may be simply shown that the percentage of the energy which is contained in a reconstructed image X_{BP} is given by E, where

$$E = \frac{\sum_{i=p}^{q} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2} \,. \tag{7.5}$$

The choice of p and q depends on the relative magnitudes of the singular values, which are a function of the input data. These parameters may, in general, be estimated from a plot of the eigenavalues $\lambda_i = \sigma_i^2$ as a function of the index i. In certain cases, an abrupt change in the eigenvalues is easily

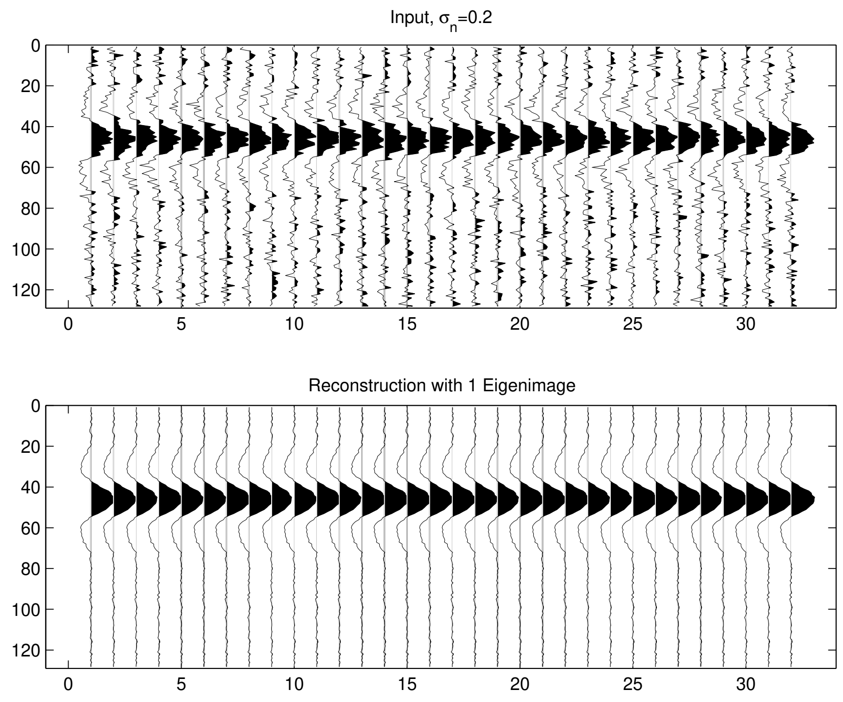


Figure 7.1: A flat event immersed in nose and the recostruction by means of the first eigenimage

recognised. In other cases, the change in eigenvalue magnitude is more gradual and care must be exercised in the choice of the appropriate index values.

In Figures 7.1 and 7.2 we illustrate the reconstruction fo a flat event immersed in noise using the first eigenimage of the data. In this example only the most energetic singular value was retained. When the data exhibit some type of moveout, one eigenimage is not sufficient to properly reconstruct the data. This can be observed in Figures 7.3 and 7.4.

As we have seen, decomposition of an image $\mathbf{X}$ into eigenimages is performed by means of the SVD of $\mathbf{X}$. Many authors also refer to this decomposition as the Karhunen-Loève or KL transformation. We believe however, that the SVD and KL approaches are not equivalent theoretically for image processing and, in order to avoid confusion, we suggest the adoption of the term eigenimage processing. Some clarification is in order.

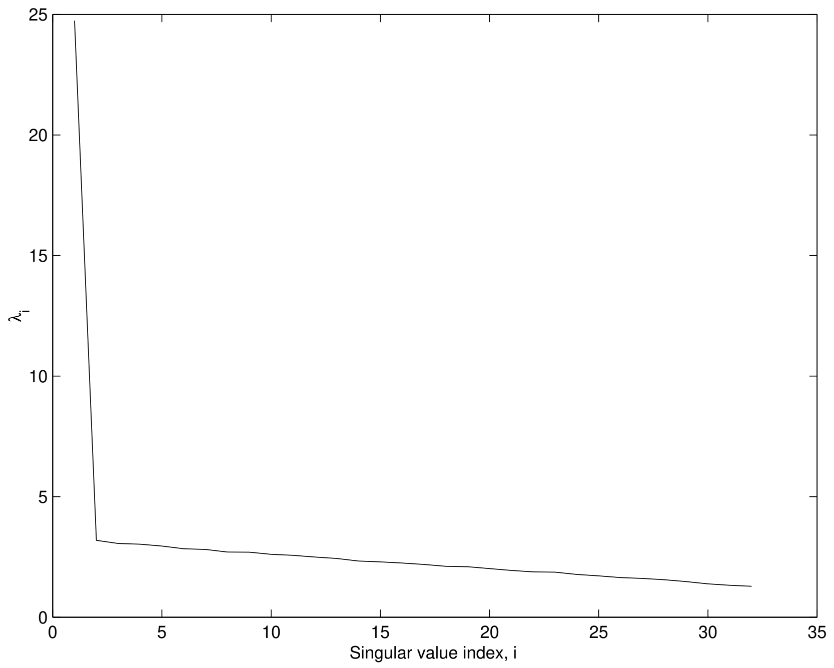


Figure 7.2: Spectrum of singular values for the data in Figure 7.1.

A wide sense stationary process $\xi(t)$ allows the expansion

$$\widehat{\xi}(t) = \sum_{n=1}^{\infty} c_n \psi_n(t) \qquad 0 < t < T \tag{7.6}$$

where $\psi_n(t)$ is a set of orthonormal functions in the interval $(0, T)$ and the coefficients c_n are random variables. The Fourier series is a special case of the expansion given by equation (7.6) and it can be shown that, in this case, $\xi(t) = \widehat{\xi}(t)$ for every t and the coefficients c_n are uncorrelated only when $\xi(t)$ is mean squared periodic. Otherwise, $\xi(t) = \widehat{\xi}(t)$ only for $|t| < T/2$ and the coefficients c_n are no longer uncorrelated. In order to guarantee that the c_n are uncorrelated and that $\xi(t) = \widehat{\xi}(t)$ for every t without the requirement of mean squared periodicity, it turns out that the $\psi_n(t)$ must be determined from the solution of the integral equation

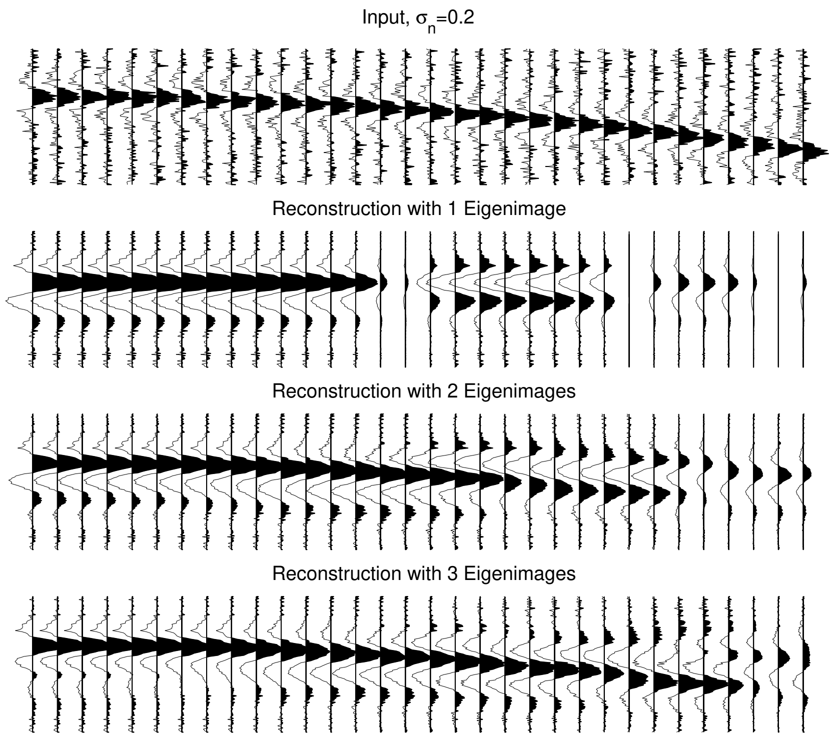


Figure 7.3: A Parabolic event immersed in nose and the reconstruction by means of the 1,2 and 3 eigenimages

$$\int_0^T R(t_1,t_2)\psi(t_2)dt_2 = \lambda\psi(t_1) \qquad 0 < t_1 < T \tag{7.7}$$

where $R(t_1,t_2)$ is the autocovariance of the process $\xi(t)$.

Substituting the eigenvectors which are the solutions of equation (7.7) into equation (7.6) gives the KL expansion of $\xi(t)$. An infinite number of basis functions is required to form a complete set. For a $N \times 1$ random vector $\mathbf{x}$ we may write equation (7.6) in terms of a linear combination of orthonormal basis vectors $\mathbf{w_i} = (w_{i1}, wi2, \ldots, wiN)^T$ as

$$x_k = \sum_{i=1}^{N} y_i w_{ik} \qquad k = 1, 2, \ldots, N \tag{7.8}$$

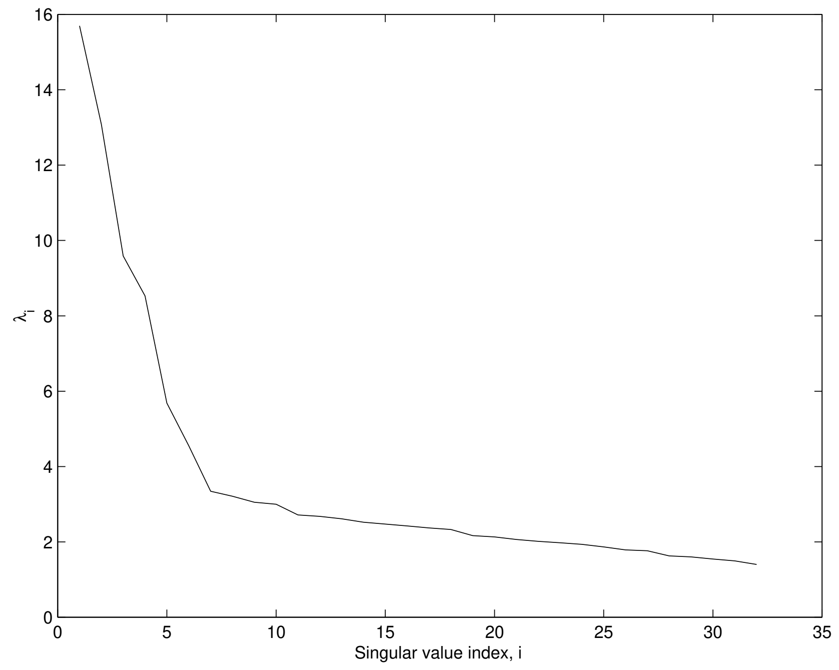


Figure 7.4: Spectrum of singular values for the data in Figure 7.3.

which is equivalent to

$$\mathbf{x} = \mathbf{W}\mathbf{y} \tag{7.9}$$

where $\mathbf{W} = (\mathbf{w_1}, \mathbf{w_2}, \ldots, \mathbf{w_N})$. Now only N basis vectors are required for completeness. The KL transformation or, as it is also often called, the KL transformation to principal components, is obtained as

$$\mathbf{y} = \mathbf{W^T}\mathbf{x} \tag{7.10}$$

where $\mathbf{W}$ is determined from the covariance matrix $\mathbf{C_x}$ of the process

$$\mathbf{C_x} = \mathbf{W}\mathbf{\Lambda}\mathbf{W^T} \tag{7.11}$$

Let us now turn our attention to the problem of the KL transformation for multivariate statistical analysis. In this case we consider M vectors $\mathbf{x_i}, \mathbf{i} = \mathbf{1}, \mathbf{M}$ arranged in a $M \times N$ data matrix $\mathbf{X}$. The M rows of the data matrix are viewed as M realizations of the stochastic process $\mathbf{x}$ and consequently the assumption is that all rows have the same row covariance matrix $\mathbf{C_r}$. The KL transform now becomes

$$\mathbf{Y} = \mathbf{W^T X} \tag{7.12}$$

An unbiased estimate of the row covariance matrix is given by

$$\hat{\mathbf{C}}_\mathbf{r} = \frac{1}{M-1} \sum_{i=1}^{M} \mathbf{x_i x_i^T} \tag{7.13}$$

assuming a zero mean process for convenience. Since the factor $M-1$ does not influence the eigenvectors, we can see from equation (12) and the definition of $\mathbf{U}$ that $\mathbf{W} = \mathbf{U}$. Consequently, we can rewrite equation (11) as

$$\mathbf{Y} = \mathbf{U^T X} \tag{7.14}$$

Substituting equation (7.1) into equation (7.14), we obtain

$$\mathbf{Y} = \mathbf{U}^T \mathbf{U} \Sigma \mathbf{V}^T = \Sigma \mathbf{V}^T \tag{7.15}$$

The principal components contained in the matrix $\mathbf{Y}$ may be viewed as the inner product of the eigenvectors of $\mathbf{XX^T}$ with the data, or as the weighted eigenvectors of $\mathbf{X^T X}$.

Since $\mathbf{X}$ may be reconstructed from the principal component matrix $\mathbf{Y}$ by the inverse KL transformation

$$\mathbf{X} = \mathbf{UY} \tag{7.16}$$

we may combine last two equations to obtain

$$\mathbf{X} = \mathbf{U \Sigma Y^T} \tag{7.17}$$

Last equation is identical with equation (7.1), showing that, providing we are considering a multivariate stochastic process, the SVD and the KL transformation are computationally equivalent.

7.2 Eigenimage analysis of common offset sections

We investigate the application of eigenimage analysis to common offset sections. Our principal goal is to show that often, common offset sections can be efficiently compressed using eigenimages. A subsidiary goal is to improve the S/N ratio of pre-stack data by eigenimage filtering of common offset sections.

We consider the data matrix X to be composed of n_x traces with n_t data points per trace, the n_x traces forming the columns of X. The Singular Value Decomposition (SVD) of X (Lanczos, 1961), is given by:

$$X = \sum_{i=1}^{r} \lambda_i \, u_i \, v_i^T \,, \tag{7.18}$$

where r indicates the rank of the matrix X, u_i is the i-th eigenvector of $X\,X^T$, v_i is the $i-$th eigenvector of $X^T\,X$ and λ_i is the i-th singular values of X. Andrew and Hunt (1977) called the outer product $u_i\,v_i^T$ the i-th eigenimage of the matrix X.

Suppose that X represents a seismic section and that all the n_x traces are linearly independent. In this case the matrix X is of full rank and all the singular values are different from zero. A perfect reconstruction of X requires all eigenimages. On the other hand, in the case where all n_x traces are equal to within a scale factor, all traces are linearly dependent, X is of rank one and may be perfectly recovered by the first eigenimage, $\lambda_1 u_1\,v_1^T$.

The eigenimage decomposition can be used to optimally extract laterally coherent waveforms. In general, common offset sections exhibit a good lateral coherence. Our approach in this paper is to first decompose the pre-stack data cube into common offset sections and then apply eigenimage analysis to compress each common offset section and improve the S/N ratio.

Our strategy is summarized as follows:

1. The pre-stack data cube is decomposed into common offset sections, in our examples we construct 10 common offset sections containing traces with offsets indicated in Table 1.

2. Each common offset section is decomposed into eigenimages. 3- Only the eigenvectors that correspond to the first p singular values are kept.

3. Equation (7.18) is used to reconstruct the common offset section. If the misfit is acceptable, we save the vectors u_i, v_i, λ_i, $i = 1 \dots p$.

It is interesting to note that the amount of data compression that can be achieved using this procedure is remarkably high. Using the SVD we can represent each common offset section by n_2 floats:

$$n_2 = p \times n_t + p \times n_x + p \,.$$

We define the compression ratio as follows

$$C = (n_1 - n_2)/n_2,$$

where $n_1 = n_x \times n_t$ is the total number of floats required to represent the common offset section, X.
In Table 1 we summarize the compression ratio for the ten common offset sections in which we have decomposed the data cube. In this example p corresponds to the number of singular values that account for 30% of the total power encountered in the spectrum of singular values. In Figure 7.51 we portray the spectra of singular values. We note that the eigen-decomposition is in terms of a few energetic singular values that correspond to coherent events in the common offset domain.

In Figures 7.6 and 7.7 we display the common offset section #2 after and before eigenimage filtering. Since the evenst are fairly flat, we can always retain the information content of the section in a few eigenimages. compression and S/N ratio enhancement

In Figures 7.8 and 7.9 we display a CDP after and before performing the eigenimage analysis in common offset domian. It is clear that we cannot

COS#	Offset [m]	p	$C = (n_1 - n_2)\,/\,n_2$
1	0-221	9	13.7
2	221-427	6	20.0
3	427-633	4	18.7
4	633-839	5	17.8
5	839-1045	4	23.7
6	1045-1250	5	21.2
7	1250-1456	6	18.2
8	1456-1662	6	14.4
9	1662-1868	6	15.2
10	1868-2780	7	13.0

Table 7.1: Compression ratios for 10 common offset sections. The variable p indicates the number of singular values used in the eigen-decomposition.

use eigenimages in the CDP domain, but after filtering in the common offset domain an sorting in CDPs we note that some high frequency noise at near offset traces was eliminated.

In summary, by sorting the data into common offset section we have been able to apply the eigenimage analysis on individual common offset traces. The pre-stack volume is reconstructing with a minimal distortion.

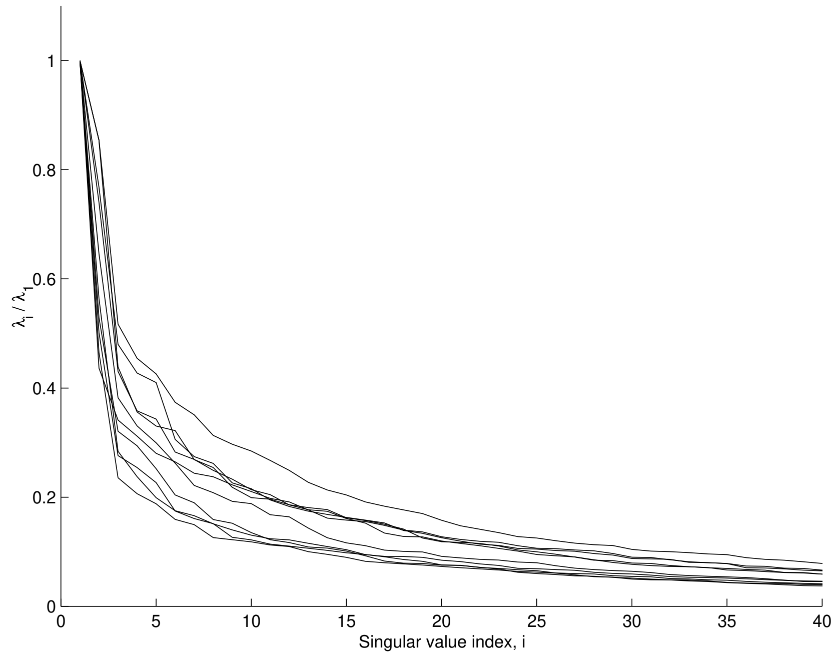


Figure 7.5: Spectra of singular values for the 10 common offset sections used to test the algorithm.

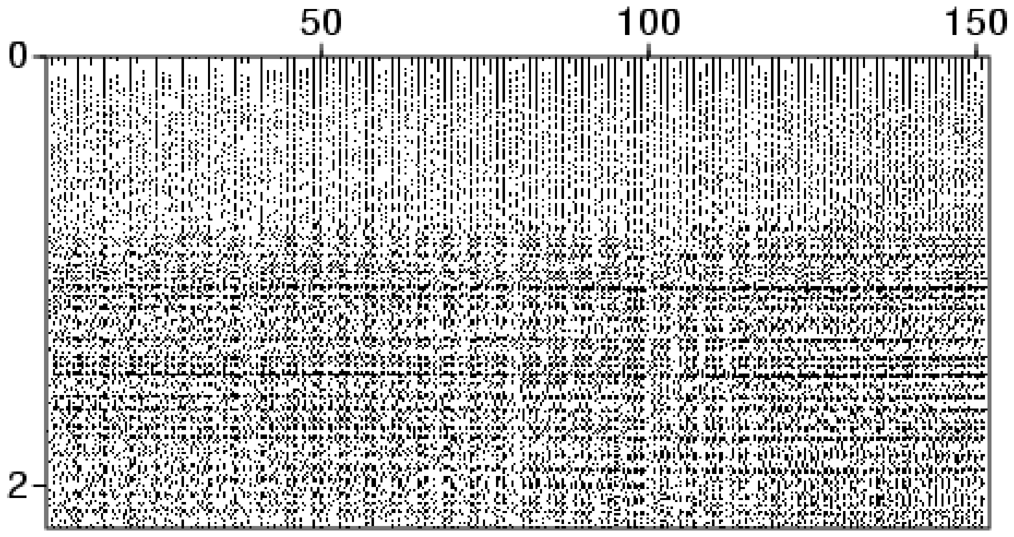


Figure 7.6: Common offset section #2.

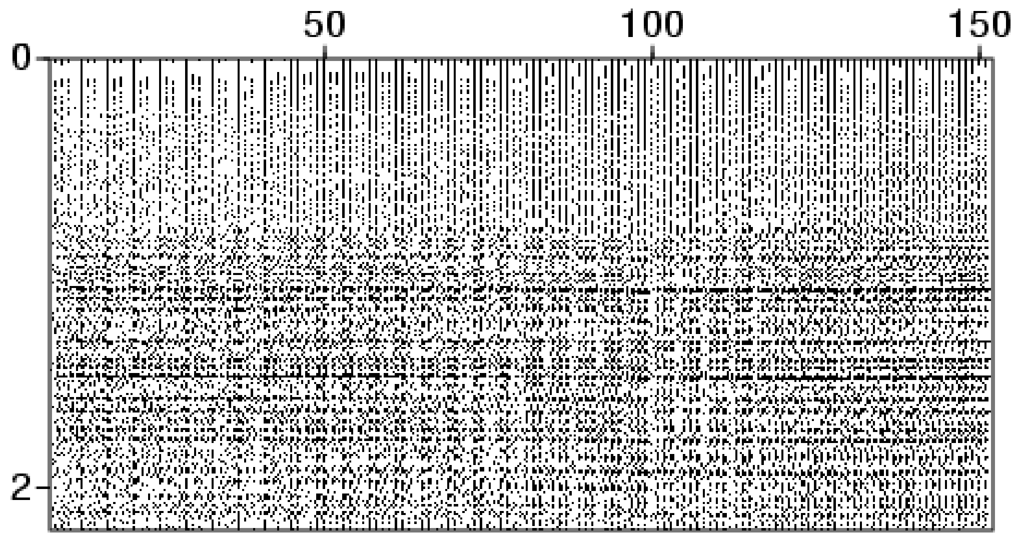


Figure 7.7: Common offset section #2 after eigenimage filtering

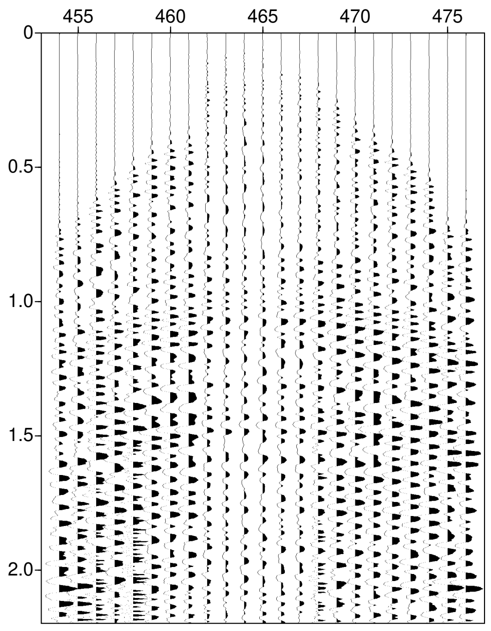


Figure 7.8: Original CDP.

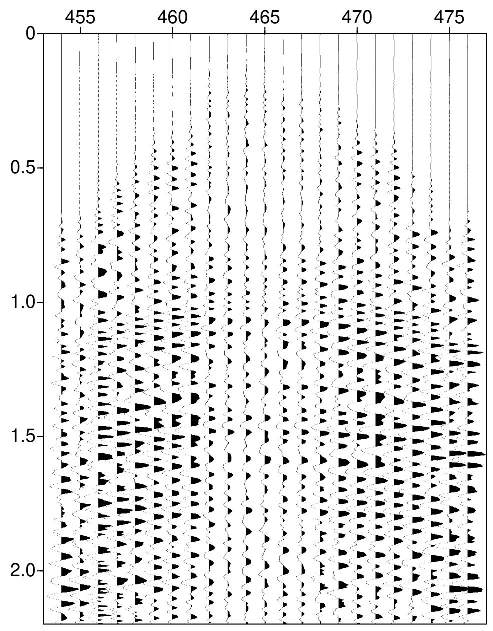


Figure 7.9: CDP after Eigenimage filtering in the common offset domain

7.2.1 Eigenimages and application to Velocity Analysis

Eigen-decomposition of seismic data (hyperbolic windows in CMP gathers) can be used to design coherence measures for high resolution velocity analysis. The idea is to replace the semblance measure by a norm that is a function of the eigenvalues of the covariance matrix of the gate of analysis. In this section we will derive a very simple algorithm that can be used to compute high resolution coherence measures for velocity analysis.

Techniques that exploit the eigen-structure of the covariance matrix have been borrowed from the field of array processing (Bienvenu and Kopp, 1983; Wax et al., 1984), and applied to velocity analysis by different researchers (Biondi and Kostov, 1989; Key and Smithson, 1990; Kirlin, 1992).

The seismic signal, in the presence of noise, at receiver i may be modeled using the following equation:

$$x_i(t) = s(t-\tau_i) + n_i(t) \qquad i = 1, N\,, \tag{7.19}$$

where $\tau_i = (t_0^2 + d_i^2/v^2)^{1/2} - t_0$ is the delay of the signal between the i-th receiver and a receiver having $d_0 = 0$. If a waveform is extracted along a hyperbolic path parametrized with velocity v, equation (7.19) may be rewritten as

$$x_i(t) = s(t) + n_i(t) \qquad i = 1, N\,, \tag{7.20}$$

noindent where, to avoid notational clutter, I used the same variable $x(t)$ to designate the delayed waveform (equation (6.19)) and the corrected waveform (equation (6.20)). The covariance matrix of the the signal is defined as:

$$R_{i,j}(t) = E[x_i(t)x_j(t)] \qquad i,j = 1, N\,, \tag{7.21}$$

where E denotes the expectation operator. If we assume the noise and signal to be uncorrelated the data covariance matrix becomes:

$$R_{i,j}(t) = R_{s\,i,j}(t) + \sigma_n^2(t)\delta_{i,j}\,, \tag{7.22}$$

where $R_{s\,i,j}(t)$ denotes the signal covariance matrix, and $\delta_{i,j} = 1$, if $i = j$ and $\delta_{i,j} = 0$, otherwise. Assuming a stationary source and a stationary noise process, we may drop the dependence on t. It is easy to verify that the eigenvalues of the covariance matrix become

$$\lambda_i = \lambda_{si} + \sigma_n^2 \qquad i = 1, 2, \dots, N\,, \tag{7.23}$$

where λ_{si} are the eigenvalues of the signal covariance matrix. Assuming that the signal is invariant across each trace, the signal covariance matrix is rank 1, and we can write the following relationships:

$$\begin{aligned} \lambda_{s1} &= N.P_s \\ \lambda_{si} &= 0 \qquad i = 2, ..., N\,, \end{aligned} \tag{7.24}$$

where $P_s = E[s(t)^2]$ denotes the signal power. Using equation (6.23), the eigenvalues of the data covariance matrix become

$$\begin{aligned} \lambda_1 &= N.P_s + \sigma_n^2 \\ \lambda_i &= \sigma_n^2 \quad i = 2, ..., N\,. \end{aligned} \tag{7.25}$$

For uncorrelated noise, the minimal $N-1$ eigenvalues of the data are equal to the variance of the noise. The largest eigenvalue is proportional to the power of energy of the coherent signal plus the variance of the noise.

In real situations, the eigen-spectrum is retrieved from an estimate of the data covariance matrix. If the stationary random processes $x_i(t)$ and $x_j(t)$ are ergodic the ensemble averages defined in equation (7.21) can be replaced by time averages (see for instance, Bendat and Piersol, 1971). The estimator of the covariance matrix becomes:

$$\hat{R}_{i,j} = \frac{1}{2M+1} \sum_{k=-M}^{M} x_i(k\Delta t) x_j(k\Delta t)\,. \tag{7.26}$$

Using the results given in equations (7.24) and (7.25) it is evident that an estimator of the noise variance is

$$\hat{\sigma}_n^2 = \frac{1}{N-1} \sum_{i=2}^{N} \hat{\lambda}_i\,. \tag{7.27}$$

Similarly, an estimator of the signal energy is given by

$$\hat{P}_s = \frac{\hat{\lambda}_1 - \hat{\sigma}_n^2}{N}\,, \tag{7.28}$$

and equations (7.27) and (7.28) can be combined into a single measure, the signal-to-noise-ratio:

$$\widehat{C} = \frac{1}{N}\frac{\hat{\lambda}_1 - \sum_{i=2}^{N}\hat{\lambda}_i/(N-1)}{\sum_{i=2}^{N}\hat{\lambda}_i/(N-1)}\,. \tag{7.29}$$

The coherence measure, $\hat{C}$, was devised assuming the presence of a signal and that the proper velocity is used to extract the waveform. In general the coherence ,$\hat{C}$, is computed for different gates and different trial velocities. It is convenient to explicitly emphasize the dependence of the coherence on these parameters by denoting $\hat{C}(t_0, v)$. When the gate of analysis contains only noise, the measure $\hat{C}(t_0, v)$ tends towards zero. When the trial velocity does not match the velocity of the reflection, it is not possible to decompose the eigen-structure of the data into signal and noise contributions. In this case, the covariance matrix has a complete set of eigenvalues different from zero; therefore it is not possible to recognize which part of the eigen-spectrum belongs to the noise and which belongs to the signal process.

Key and Smithson (1990) proposed another coherence measure based on a log-generalized likelihood ratio which tests the hypothesis of equality of eigenvalues,

$$\hat{W}_{ml} = M \log^N \left[\frac{(\sum_{i=1}^{N}\hat{\lambda}_i/N)^N}{\prod_{i=1}^{N}\hat{\lambda}_i}\right]\,. \tag{7.30}$$

In the absence of signal, $\lambda_i = \sigma_n^2$, $i = 1, N$ and hence $W_{ml} = 0$. In the presence of a single reflected signal, $\lambda_1 \neq 0$, $\lambda_i = 0$, $i = 2, N$ and $W_{ml} \to \infty$. Therefore, W_{ml} provides a strong discrimination between signal and noise. Key and Smithson (1990) combined equation (7.29) and (7.30) into a single measure, K_{ml}, given by the product:

$$\hat{K}_{ml} = \hat{W}_{ml}\,\hat{C}\,. \tag{7.31}$$

It is important to point out that only one eigenvalue, λ_1, is required to estimate the coherence measure, $\hat{C}$. Since

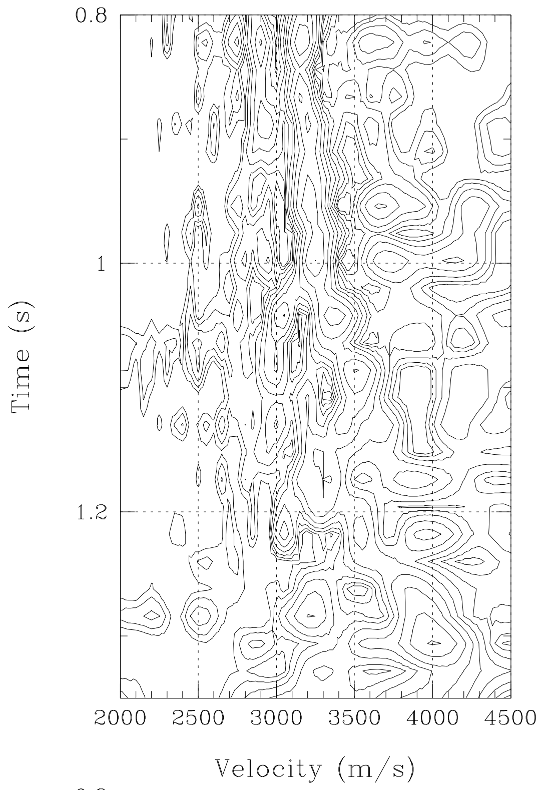
$$Trace(\hat{\mathbf{R}}) = \hat{\lambda}_1 + \hat{\lambda}_2 + \ldots + \hat{\lambda}_N \tag{7.32}$$

where

$$Trace(\hat{\mathbf{R}}) = \sum_{i=1}^{N} \hat{R}_{i\,i}\,. \tag{7.33}$$

It is easy to see from equations (6.32) and (6.33) that only $\hat{\lambda}_1$ is needed to compute the coherence measure, $\hat{C}$.

It is also important to mention that the velocity panel obtained via the SNR coherence measure can be further improved by adopting a bootstrap procedure (Sacchi, 1998). In this case, the seismic traces are randomly sampled to produce individual estimates of the coherence measure. From this information one can obtained an average coherence measure and a histogram (in fact a density kernel estimator) of the position of the peak that optimizes the coherence. The improve SNR coherence obtained with this techniques is portrayed in Figure (6.11).



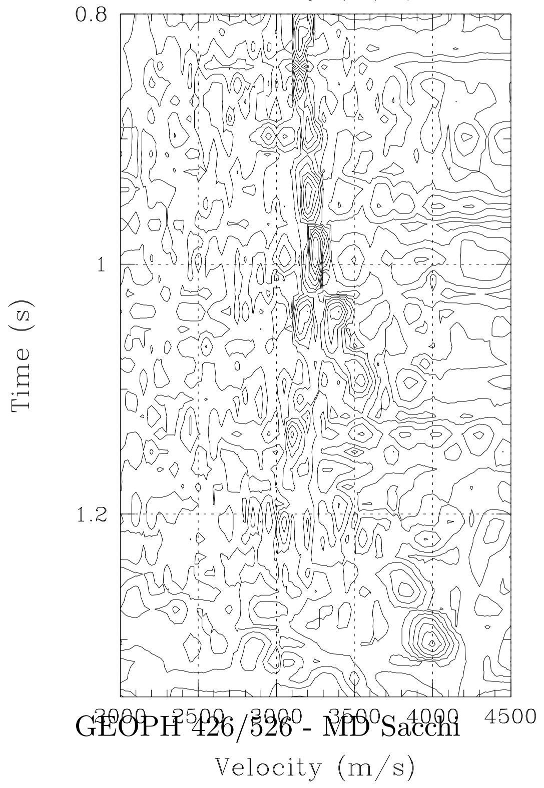


Figure 7.10: Left: Semblance of a CMP gather. Right: High resolution coherence analysis (SNR measure).

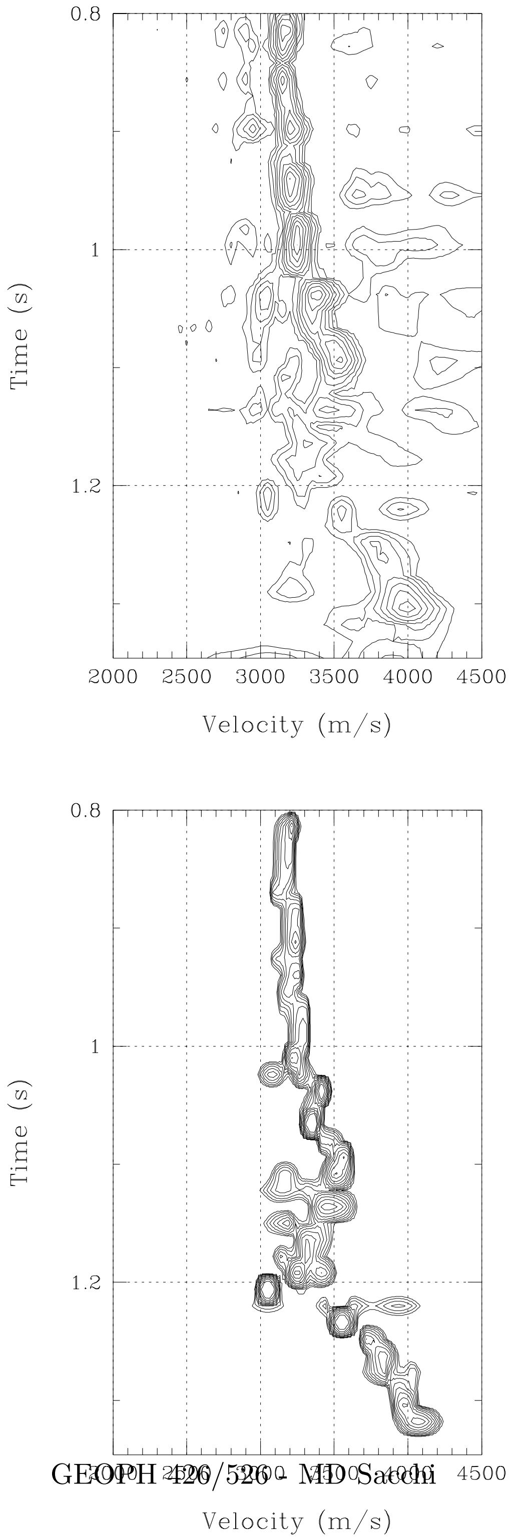


Figure 7.11: Left: Average SNR measure obtained via bootstrapping individual realizations. Right: Frequency distribution of the peak that maximizes the coherence after 50 bootstrap realizations.

7.3 A Matlab Code for Eigenimage Analysis

```
% A Code to filter data using the Eigenimage approach

% Generation of the model. Single
% event with parabolic moveout.

dt = 4./1000; w = ricker(20.,dt); nw=max(size(w));
nx = 32; nt = 128;
DATA = zeros(nx,nt);

for i=1:nx
 for j=1:nw
   c= fix(0.05*i*i);
  DATA(i,20+j+c) = w(j);
 end
end

% Add noise to the model
NOISE = 0.2 * randn(nx,nt);
DATA = DATA + NOISE;


[U S V] = svd(DATA);

% Reconstruction with 3 eigenimages

 p = 4;    % Keep 1,2,3.
 q = min(size(S));
 for i = p:q;
 S(i,i) = 0;
 end

% Filtered image
```

```
DATA = U*S*V';
```

7.3.1 References

Andews, H. C., and Hunt, B. R., 1977, *Digital image restoration*, Prentice-Hall, Signal Processing Series.

Bienvenu, G., and Kopp, L., 1983, Optimality of high resolution array processing using the eigensystem approach: IEEE, Trans. Acoust., Speech, Signal Processing., **ASSP-31**, 1235-1248.

Biondi, B. L., and Kostov, C., 1989, High-resolution velocity spectra using eigenstructure methods: Geophysics, **54**, 832-842.

Key, S. C., and Smithson, S. B., 1990, New approach to seismic-event detection and velocity determination: Geophysics, **55**, 1057-1069.

Kirlin, R. L., 1992, The relationship between the semblance and the eigenstructure velocity estimators, Geophysics: **57**, 1027-1033.

Freire, S.L.M and Ulrych, T.J., 1988, Application of singular value decomposition to vertical seismic profiling, Geophysics, **53**, 778-785.

Hemon, C.H. and Mace, D., 1978, Essai d'une application de la transformation de Karhunen-Loève au traitement sismique, Geophysical Prospecting, **26**, 600-626.

Jones, I.F. and Levy, S., 1987, Signal-to-noise ratio enhancement in multichannel seismic data via the Karhunen-Loève transform, Geophysical Prospecting, **35**, 12-32.

Lanczos, C., 1961, *Linear Differential operators*, D. Van Nostrand Co.

Sacchi, M.D., 1998, A bootstrap procedure for high-resolution velocity analysis: Geophysics, **65**, 1716-1725.

Chapter 8

Radon Transforms

In this chapter, we study the numerical implementation of the Radon transform. We will analyze the problem using the inverse problem formalism and study the problem of designing a high-resolution Parabolic Radon transforms for multiple suppression.

8.1 Slant Stacks

Radon transforms with linear integration paths are generally called slant stacks.

Different techniques have been devised to identify and/or filter linear events. Generally, they have the following common framework. First, they assume that linear events are recorded on an array with discrete and limited coverage. Secondly, they assume that the noise is uncorrelated with the signals. In geophysics, linear event identification has been an active field of research. Two classic examples are vertical seismic profiles (VSP) and slowness vector estimation in seismographic arrays for earthquake detection and location. In VSP processing, linear event detection estimation is used to identify and separate the principal components of the VSP data: the up-going and the down-going waves.

A general strategy for event identification-estimation involves the following approach. First, the data are transformed into a new domain where each component may be isolated. After masking the undesired parts, the data are mapped back to the original domain retaining only the desired information.

In seismic processing, the Radon transform is commonly known as the $\tau - p$ (τ denotes time and p ray parameter) or slant stack transform. The original idea developed by Radon in 1917 (Deans, 1983), has provided a basic framework for many problems of image reconstruction in physics, astronomy, medicine, optics, non-destructive testing and geophysics. In image processing, it is also called the Hough transform (Pratt, 1991), which may be regarded as a transformation of a line in Cartesian coordinate space to a point in polar coordinate space.

In geophysics, the properties of the Radon transform are examined by Phinney et al. (1981), Durrani and Bisset (1984) and Tatham (1984). Chapman (1981) developed exact formulas for a point source in Cartesian or spherical coordinates, and for a line source in cylindrical coordinates. The relationship between the Radon transform and the plane wave decomposition is also well-established (Stoffa et al., 1981; Treitel et al., 1982). Least-squares procedures to compute the Radon transform were investigated by Thorson and Claerbout (1985), Beylkin (1987) and Kostov (1990). These authors showed how to mitigate the smearing caused by the finite aperture. Recently, Zhou and Greenhalgh (1994) linked the least-squares solution to p-dependent Wiener filters. These researchers derived the slant stack formulas in the continuous domain, but the resulting algorithms are identical to those obtained by other researchers (Beylkin, 1987; Kostov, 1990).

The problem of computing the Radon transform may be posed in the frequency-space domain ($f - x$) to avoid the inversion or large matrices. This technique was adopted by Beylkin (1987), Kostov (1990), Foster and Mosher (1992), and recently by Zhou and Greenhalgh (1994). This allows us to solve several small problems in the band that comprises the signal. Some stability concerns arise when the problem is tackled in this manner. Particularly, a least-squares solution can be extremely unstable at low frequencies. Besides, it is interesting that slant stacks can also be computed in the time-space domain. Thorson and Claerbout (1985) and, recently Yilmaz and Tanner (1994) have presented high-resolution least-squares slant stack operators designed in the time-space domain. Their procedures use an iterative inversion scheme especially devised to solve large linear sparse operators. Thorson and Claerbout (1985) have also shown how to update each iteration the variances of the model to drive the solution to minimum entropy.

8.1.1 The slant stack operator (conventional definition)

Let $u(h,t)$ represent a seismic signal. Throughout this chapter the variable t designates the time and h the offset or range. For a the continuous array, we define the slant stack using the following transformation

$$v(p,\tau) = (\mathcal{L}u)(p,\tau) = \int_{-\infty}^{\infty} u(h, t = \tau + h\,p)dh\,. \tag{8.1}$$

Where p and τ denote the slope or ray parameter and the intercept time, respectively. $v(p,\tau)$ is used to designate the signal in the $\tau - p$ domain. The adjoint transform $\mathcal{L}^*$ is given by

$$\tilde{u}(h,t) = (\mathcal{L}^* v)(p,\tau) = \int_{-\infty}^{\infty} v(p, t = \tau - hp)dp\,. \tag{8.2}$$

In the frequency domain, the pair of transformations are given by,

$$V(p,\omega) = \int_{-\infty}^{\infty} U(h,\omega) e^{i\omega ph} dh, \tag{8.3}$$

$$\tilde{U}(h,\omega) = \int_{-\infty}^{\infty} V(p,\omega) e^{-i\omega ph} dp, \tag{8.4}$$

substituting (2.3) into (2.4) yields

$$\tilde{U}(h,\omega) = \int_{-\infty}^{\infty} U(h',\omega) \int_{-\infty}^{\infty} e^{-i\omega p(h-h')} dp\, dh' \tag{8.5}$$

which may be written as follows

$$\tilde{U}(h,\omega) = U(h,\omega) * \rho(h,\omega)\,, \tag{8.6}$$

where $*$ denotes convolution and the function ρ is given by

$$\rho(h,\omega) = \int_{-\infty}^{\infty} e^{-i\omega ph} dp\,, \tag{8.7}$$

making the substitution $z = -\omega p$ equation (8.7) becomes

$$\rho(h,\omega) = \int_{-\infty}^{\infty} \frac{1}{|\omega|} e^{ihz} dz = \frac{2\pi}{|\omega|}\delta(h)\,. \tag{8.8}$$

The convolution operator is a delta function with respect to the variable h. Using the property of the δ function,

$$\begin{aligned} \tilde{U}(h,\omega) = & \tfrac{2\pi}{|\omega|} U(h,\omega) * \delta(h) \\ = & \tfrac{2\pi}{|\omega|} U(h,\omega) \end{aligned} \tag{8.9}$$

the inversion formula becomes,

$$U(h,\omega) = \frac{|\omega|}{2\pi}\tilde{U}(h,\omega)\,. \tag{8.10}$$

The inverse is computed in two steps. First, the adjoint is used to evaluate $\tilde{U}(h,\omega)$. Then, $\tilde{U}(h,\omega)$ is multiplied by the ρ filter frequency response. The conventional slant stack pair in the frequency domain results in,

$$\begin{aligned} V(p,\omega) &= \int_{-\infty}^{\infty} U(h,\omega) e^{i\omega ph} dh, \\ U(h,\omega) &= \frac{|\omega|}{2\pi}\int_{-\infty}^{\infty} V(p,\omega) e^{-i\omega ph} dp\,. \end{aligned} \tag{8.11}$$

Now, consider that the range of p is a finite interval, $p \in [-P, P]$. This case leads to the following ρ filter,

$$\rho(h,\omega) = \int_{-P}^{P} e^{-i\omega ph} dp = 2P\frac{\sin(\omega P h)}{\omega P h}\,. \tag{8.12}$$

Substituting (8.12) in (8.6),

$$\tilde{U}(h,w) = 2P\int_{-\infty}^{\infty} U(h',\omega)\frac{\sin(\omega P(h-h'))}{\omega P(h-h')}dh'. \tag{8.13}$$

The data may be recovered after solving a deconvolution problem. Spatial deconvolution is required since the infinite range of the variable p is truncated to a finite range. The wavenumber domain response of the ρ filter has the following expression:

$$\begin{aligned}\rho(k,\omega) &= \int_{-\infty}^{\infty}\rho(h,\omega)e^{ikh}dh \\ &= \int_{-\infty}^{\infty}\int_{-P}^{P}e^{-i(\omega p-k)h}dhdp \\ &= \int_{-P}^{P}\delta(\omega p-k)dp \\ &= \frac{1}{\omega}\int_{-\omega P}^{\omega P}\delta(k'-k)dk' \qquad (8.14)\\ &= \begin{cases}\frac{1}{\omega} & k\le|\omega P| \\ 0\ , & \text{otherwise}\end{cases}\end{aligned}$$

According to the last equation, the spatial deconvolution will be unstable if the wavenumbers in the data lie outside the range $[-\omega P,\omega P]$. Equation (2.14) also shows that the deconvolution is unstable at low frequencies.

8.1.2 The inverse slant stack operator

The definition of the forward slant stack operator and its adjoint maybe changed to construct another slant stack pair,

$$u(h,t) = (\mathcal{L}^*v)(p,\tau) = \int_{-\infty}^{\infty}v(p,t=\tau-hp)dp \tag{8.15}$$

$$\tilde{v}(p,\tau) = (\mathcal{L}u)(h,t) = \int_{-\infty}^{\infty}u(h,t=\tau+hp)dh, \tag{8.16}$$

the pair of transformations can be posed in the frequency-offset domain,

$$U(h,\omega) = \int_{-\infty}^{\infty} V(p,\omega) e^{-i\omega p h} dp, \tag{8.17}$$

$$\tilde{V}(p,\omega) = \int_{-\infty}^{\infty} U(h,\omega) e^{i\omega p h} dh \,. \tag{8.18}$$

Substituting, (8.17) into (8.18) yields,

$$\tilde{V}(p,\omega) = \int_{-\infty}^{\infty} V(p',\omega) \int_{-\infty}^{\infty} e^{-i\omega h(p-p')} dh dp' \,, \tag{8.19}$$

where now, the convolution is with respect to the variable p, and the convolutional operator is given by

$$\gamma(p,\omega) = \int_{-\infty}^{\infty} \frac{1}{|\omega|} e^{ihp} dh = \frac{2\pi}{|\omega|} \delta(p) \,. \tag{8.20}$$

The γ filter is a delta function with respect to the variable p. Therefore, equation (8.19) becomes,

$$\tilde{V}(p,\omega) = \frac{2\pi}{|\omega|} V(p,\omega) \tag{8.21}$$

or equivalently

$$V(p,\omega) = \frac{|\omega|}{2\pi} \tilde{V}(p,\omega) \,. \tag{8.22}$$

From the above derivation, it is clear that the ρ and the γ filters have the same frequency response. Finally, the slant stack pair becomes,

$$\begin{aligned} V(p,\omega) &= \frac{|\omega|}{2\pi} \int_{-\infty}^{\infty} U(h,\omega) e^{i\omega p h} dh, \\ U(h,\omega) &= \int_{-\infty}^{\infty} V(p,\omega) e^{-i\omega p} dp \,. \end{aligned} \tag{8.23}$$

Assuming that $h \in [-H, H]$ (finite aperture), the γ filter has the following structure

$$\gamma(p,\omega) = \int_{-H}^{H} e^{i\omega p h} dh = 2H\frac{\sin(\omega H p)}{\omega H p}\,. \tag{8.24}$$

Hence, $V(p,\omega)$ may be calculated by solving the following the integral equation,

$$\tilde{V}(p,\omega) = 2H\int_{-\infty}^{\infty} V(p',\omega)\frac{\sin(\omega H(p-p'))}{\omega H(p-p')}dp'\,. \tag{8.25}$$

After a comparison of the slant stacks pairs, equations (8.11) and (8.23), a deconvolution procedure is required in both cases. In the conventional slant stack transform, deconvolution is necessary to recover the data from the $\tau - p$ space. In the inverse slant stack operator the deconvolution process is required to estimate the $\tau - p$ space. The truncation effect of the variable p may be alleviated by choosing the proper region of support of the transform. The truncation of the variable h is associated with the transform's resolution and cannot be alleviated. Generally, both the variables h and p are truncated. Thus, deconvolution should be carried out in both the forward and inverse transform (Zhou and Greenhalgh, 1994). However, the range of p may be chosen so that most of the energy in the signal lies within this range.

8.1.3 The sampling theorem for slant stacks

Assuming that the wavefield is evenly sampled according to $U(n\Delta h,\omega), n = 0, \pm 1, \pm 2, \ldots$, the relationship between the $\tau - p$ and the $h - t$ spaces is given by

$$U(n\Delta h,\omega) = \frac{|\omega|}{2\pi}\int_{-\infty}^{\infty} V(p,\omega)e^{-i\omega p n\Delta h}dp\,, \tag{8.26}$$

where $V(p,\omega)$ denotes the slant stack corresponding to a continuous wavefield $U(h,\omega)$. The integration domain can be decomposed into small subdomains as follows,

$$\begin{aligned} U(n\Delta h,\omega) &= \frac{|\omega|}{2\pi}\sum_{k=-\infty}^{\infty}\int_{-(2k-1)\frac{\pi}{\omega\Delta h}}^{(2k+1)\frac{\pi}{\omega\Delta h}} V(p,\omega)e^{-i\omega pn\Delta h}dp \\ &= \frac{|\omega|}{2\pi}\sum_{k=-\infty}^{\infty}\int_{-\frac{\pi}{\omega\Delta h}}^{\frac{\pi}{\omega\Delta h}} V(p+2k\frac{\pi}{\omega\Delta h},\omega)e^{-i\omega(p+2k\frac{\pi}{\omega\Delta h})n\Delta h}dp \end{aligned} \tag{8.27}$$

since $e^{-i2\pi nk} = 1 \ \forall n\, k$, the last equation may be written in the following form

$$U(n\Delta h,\omega) = \frac{|\omega|}{2\pi}\int_{-\frac{\pi}{\omega\Delta h}}^{\frac{\pi}{\omega\Delta h}} V_d(p,\omega)e^{-i\omega pn\Delta h}dp, \tag{8.28}$$

where the relationship between the slant stack of the continuous signal and the one corresponding to the sampled wavefield, $V_d(p,\omega)$, is given by

$$V_d(p,\omega) = \sum_{k=-\infty}^{\infty} V(p+2k\frac{\pi}{\omega\Delta h},\omega). \tag{8.29}$$

Thus, the discrete signal has a $\omega - p$ representation with support in the $p \in [-\frac{\pi}{\omega\Delta h},\frac{\pi}{\omega\Delta h}]$ range. The components with slope $p-2\frac{\pi}{\omega\Delta h}, p+2\frac{\pi}{\omega\Delta h}, p-4\frac{\pi}{\omega\Delta h}, p+4\frac{\pi}{\omega\Delta h},\ldots$ will appear to have slope p and every slope outside the range $(-\frac{\pi}{\omega\Delta h},\frac{\pi}{\omega\Delta h})$ will have an alias inside this range. If the continuous signal has all the components inside that range, the aliased components do not exist, and therefore, we can write $V_d(p,\omega) = V(p,\omega)$. It is clear from the above discussion that spatial sampling must be chosen to avoid the aliasing effect. If $P = P_{max} = -P_{min}$, the following relationship guarantees the absence of aliasing,

$$\Delta h \leq \frac{1}{2Pf_{max}}\,, \tag{8.30}$$

where $f_{max} = \omega_{max}/2\pi$ is the maximum temporal frequency of the seismic signal. The product $P\,f_{max}$ is also the maximum wavenumber. Similarly, if Δh is given, the maximum ray parameter that can be retrieved ed without alias is given by

$$P_{max} = \frac{1}{2\Delta h f_{max}} \,. \tag{8.31}$$

For a non-symmetric slant stack, $P_{max} \neq -P_{min}$, equation (8.30) is modified as follows (Turner, 1990),

$$\Delta h \leq \frac{1}{P' f_{max}} \,, \tag{8.32}$$

where $P' = |P_{max} - P_{min}|$.

8.2 Discrete slant stacks

Discrete versions of equations of the continuous Radon pair are obtained by replacing integrals with summations and imposing finite limits. First, assume that the seismogram contains $N = L_f - L_n$ traces, where the indices L_f and L_n denote far and near offset traces, respectively.

$$v(p,\tau) = (\mathcal{L}u)(p,\tau) = \sum_{l=L_n}^{L_f} u(h_l, \tau + h_l p)\Delta h_l, \tag{8.33}$$

where $\Delta h_l = (h_{l+1} - h_l)$ for $l = L_n, \dots, L_f - 1$. Similarly, we approximate the continuous Radon transform by the following expression

$$\tilde{u}(h,t) = (\mathcal{L}^* v)(\tau, p) = \sum_{j=J_{min}}^{J_{max}} v(h, t - hp)\Delta p_j \tag{8.34}$$

where $\Delta p_j = (p_{j+1} - p_j)$ for $j = J_{min}, \dots, J_{max} - 1$. Taking the Fourier transform of the above equations yields

$$V(p,f) = \sum_{l=L_n}^{L_f} U(h_l, f) e^{2\pi i f h_l p} \Delta h_l \tag{8.35}$$

$$\tilde{U}(h,f) = \sum_{j=J_{min}}^{J_{max}} V(p,f) e^{-2\pi i f h p_j} \Delta p_j \,. \tag{8.36}$$

Using matrix notation, it is possible to rewrite the slant stack and its adjoint as follows (f is omitted to avoid notational clutter),

$$\mathbf{m} = \mathbf{L}^H \mathbf{d} \tag{8.37}$$

$$\tilde{\mathbf{d}} = \mathbf{L}\mathbf{m} \tag{8.38}$$

The operators $\mathbf{L}$ and $\mathbf{L^H}$ form an adjoint pair. The matrix $\mathbf{L}$ is the forward operator, and $\mathbf{L}^*$ denotes the adjoint operator. The vector $\mathbf{m}$ indicates the Radon space $V(p,f)$ at discrete values of p and fix frequency f, whereas the vector $\mathbf{d}$ indicates the data $U(h,f)$ at discrete values of h and fix frequency f.

8.2.1 The discrete slant stack operator (conventional definition)

The slant stack operator, equation (8.37), maps the $t-x$ space into the $\tau-p$ domain; the adjoint, equation (8.38), maps the $\tau-p$ domain into the $t-x$ domain. It is clear that since $\mathbf{L}$ is non-orthogonal $\mathbf{L}$ and $\mathbf{L}^H$ do not constitute an inverse pair. Given $\mathbf{m} = \mathbf{L}\mathbf{d}$, the problem is how to recover $\mathbf{d}$. A relationship between $\mathbf{d}$ and $\tilde{\mathbf{d}}$ is obtained after substituting (8.37) into (8.38)

$$\tilde{\mathbf{d}} = \mathbf{L}\,\mathbf{L^H}\,\mathbf{d}\,. \tag{8.39}$$

Equation (2.41) is uniquely invertible in $f \in B$ provided that $\det(\mathbf{L}\,\mathbf{L^H}) \neq \mathbf{0}$ in the band B,

$$\begin{aligned} \mathbf{d} = & \; (\mathbf{L}\,\mathbf{L}^H)^{-1}\tilde{\mathbf{d}} \\ = & \; \mathbf{G}^{-1}\tilde{\mathbf{d}}\,. \end{aligned} \tag{8.40}$$

The $N \times N$ matrix $\mathbf{G} = \mathbf{L}\,\mathbf{L}^H$ represents a discrete version of the ρ filter. The pair of transformations which map a signal from $f - h$ to $f - p$ and vice-versa is given by

$$\begin{aligned} \mathbf{m} = & \; \mathbf{L}\mathbf{d} \\ \mathbf{d} = & \; \mathbf{G}^{-1}\mathbf{L}\mathbf{m}\,. \end{aligned} \tag{8.41}$$

The vector **m** always exists since it is obtained using simple mapping. Both expressions constitute an inverse pair when the inverse of **G** exits. The forward and inverse pair do not permit us to model the signal when additive noise is present adequately. If the data are contaminated with noise, the noise is mapped to the Radon domain.

8.2.2 The least squares solution

Assume that the data is the result of applying a Radon operator (slant stack) to a **m**.

$$\mathbf{d} = \mathbf{L}\mathbf{m} \tag{8.42}$$

The idea is to find **m** such that the following objective function is minimized (Yilmaz, 1994),

$$J = ||\mathbf{d} - \mathbf{L}\mathbf{m}||^2 \tag{8.43}$$

The solution to this problem is the least squares solution

$$\mathbf{m} = (\mathbf{L}^H\mathbf{L})^{-1}\mathbf{L}^H\mathbf{d} \tag{8.44}$$

The inverse needs to be stabilized using a damping parameter.

$$\mathbf{m} = (\mathbf{L}^H \mathbf{L} + \mu \mathbf{I})^{-1} \mathbf{L}^H \mathbf{d} \tag{8.45}$$

In general, this is the approach that is used to compute slant stacks and parabolic Radon transform. Other techniques to improve the resolution of these operators were proposed by Sacchi and Ulrych (1995).

8.2.3 Example

In Figure (8.2), we display 3 panels. The first panel is the ideal $\tau - p$ signal; the second is the $\tau - p$ signal transformed to $offset - time$. These are two linear events with a positive and negative slope. The third panel (left) is the inverted $\tau - p$ signal using least-squares. Artifacts have been created in the inverted $\tau - p$. The alias generates these artifacts. As we have already seen f_{max}, the maximum ray parameter and the spatial sampling must satisfy a Nyquist condition (equation (7.30)). This condition is not satisfied; therefore, the $\tau - p$ domain exhibits aliasing.

In Figure (8.2)I muted the $\tau - p$ domain, eliminating all the contributions where $p > 0$. The muted $\tau - p$ domain is used to reconstruct the data; this is displayed in Figure (**??**) [Left]. This procedure can be used to discriminate down-going and up-going wavefields in Vertical Seismic Profiles (VSP).

Figures (8.2.3) and (8.4) displayed a simulation similar to the one described above, but now the original signal in $t - offset$ has spectral components that are contained in the $5 - 35$Hz band. In other words, I have eliminated the alias artifacts.

8.3 Parabolic Radon Transform (Hampson, 1986)

This is a simple modification to the slant stack instead of integrating along curves of the form

$$t = \tau + ph$$

we use curves of the type

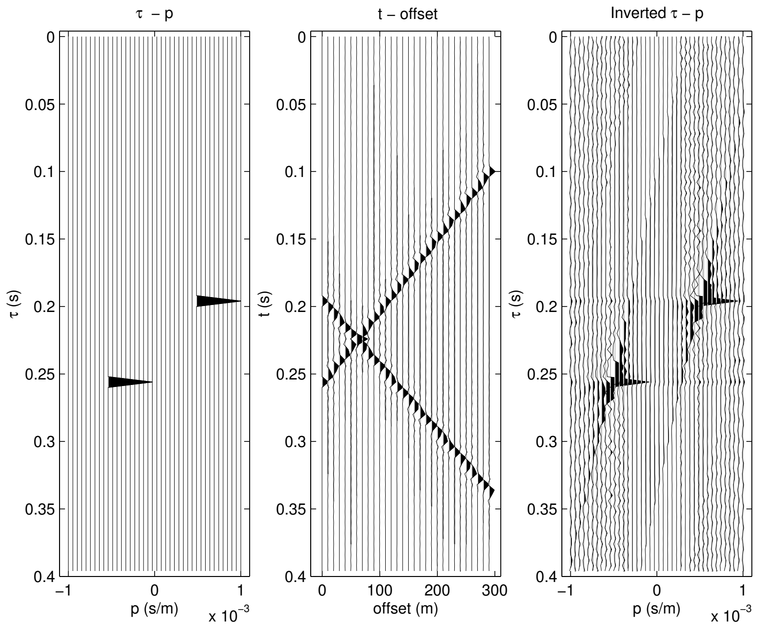


Figure 8.1: Left: Ideal $\tau - p$ panel. Center: Data generated by forward transforming the ideal $\tau - p$ panel. Right: Inverted $\tau - p$ panel.

$$t = \tau + qh^2$$

.

This is an excellent approximation to process data containing hyperbolic events after NMO correction Parabolic Radon Transform is utilized to remove multiple reflections. After NMO correction, the moveout of the primaries is zero. The residual move out of the multiples follows in a first-order approximation, a parabolic moveout. The transform isolates multiples from primaries to mute (filter) them out.

Assume you have two events in a CMP (common midpoint gather): a primary and a multiple. Let we assume that the intercept time of these events T_0 is the same.

The travel-time curve for the primary is given by:

$$T_p = \sqrt{(T_0^2 + h^2/v_p^2)} \tag{8.46}$$

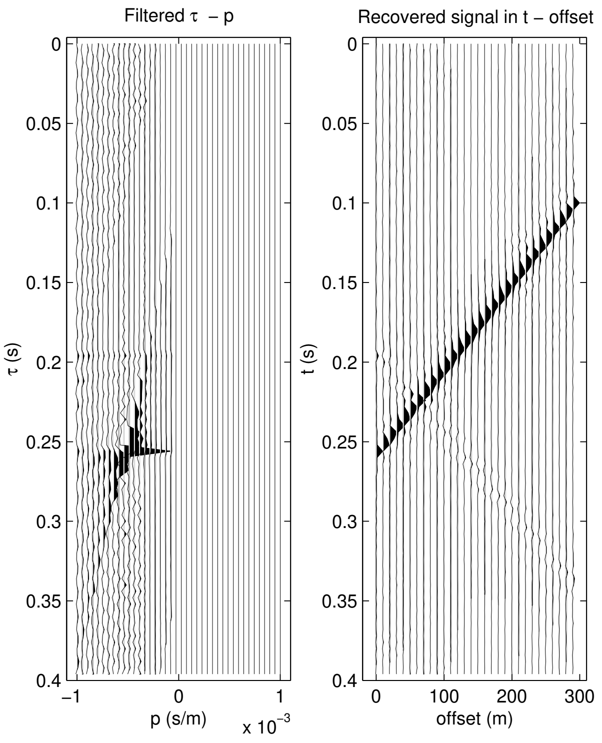


Figure 8.2: Left: Inverted $\tau - p$ panel after muting. Right: Data reconstructed by forward modelling the inverted/muted $\tau - p$ panel.

,

and the traveltime for the multiple is given by:

$$T_m = \sqrt{(T_0^2 + h^2/v_m^2)}\,. \tag{8.47}$$

If the multiple is generated by a shallow layer or by the water column we can consider $v_p > v_m$.

Now suppose that we apply NMO correction to the complete data set with the NMO law that uses the primary velocity. The NMO correction entails applying the following time shift to the data

$$\Delta T_{NMO} = T_0 - \sqrt{(T_0^2 + h^2/v_{NMO}^2)}\,, \tag{8.48}$$

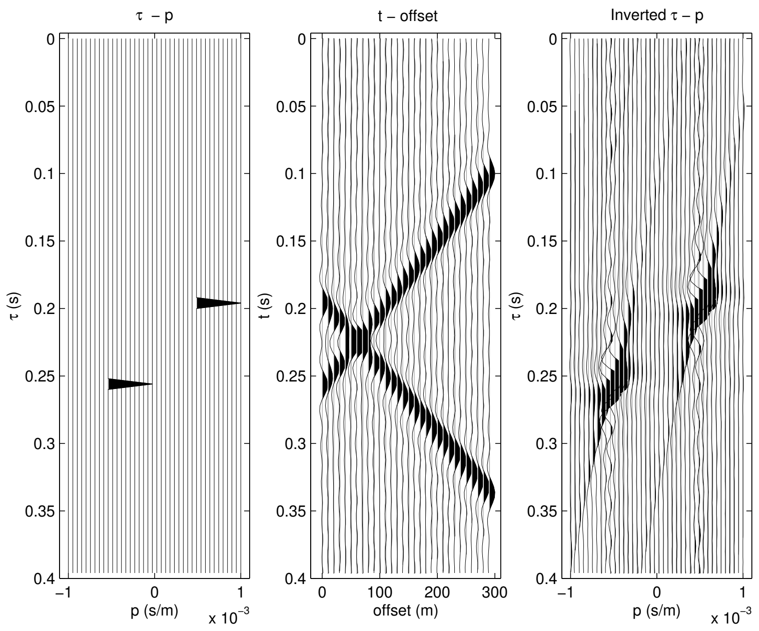


Figure 8.3: Left: Ideal $\tau - p$ panel. Center: Data generated by forward transforming and band-limiting the ideal $\tau - p$ panel. Band-limiting is needed to eliminate alias. Right: Inverted $\tau - p$ panel.

therefore, the time of the primary after NMO is

$$T_p(After) = T_p + \Delta T_{NMO} \tag{8.49}$$

It is clear that is the NMO velocity is the velocity of the primary, the time of the primary becomes

$$T_p(After) = T_0 \,. \tag{8.50}$$

In other words, the primary has the same time for all offsets (a flat event). What is the time of the multiple after NMO?. Let us try to compute it,

$$T_m(After) = T_m + T_0 - \sqrt{(T_0^2 + h^2/v_{NMO}^2)}\,, \tag{8.51}$$

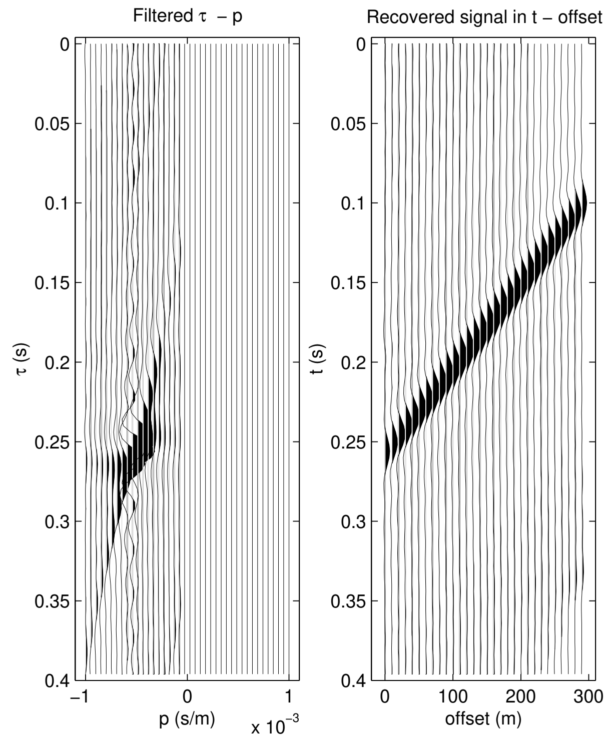


Figure 8.4: Left: Inverted $\tau - p$ panel after muting. Right: Data reconstructed by forward modelling the inverted/muted $\tau - p$ panel. Note that the alias artifacts have disappeared.

or after replacing T_m

$$T_m(After) = T_0 + \sqrt{(T_0^2 + h^2/v_m^2)} - \sqrt{(T_0^2 + h^2/v_{NMO}^2)} \tag{8.52}$$

The two square roots in the above equation can be expanded in the Taylor series (Keeping only up to the second-order term) we have

$$T_m(After) \approx T_0 + \frac{1}{2T_0\, v_m^2} h^2 - \frac{1}{2T_0\, v_{NMO}^2} h^2 \tag{8.53}$$

which can be re-written as

$$T_m(After) \approx T_0 + qh^2 \tag{8.54}$$

where

$$q = \frac{1}{2\,T_0}(\frac{1}{v^2} - \frac{1}{v_{NMO}^2})\,. \tag{8.55}$$

A transform with a parabolic integration path can be constructed by simply interchanging in the original slant stack h by h^2. Some people prefer to parameterize the parabola in terms of the residual moveout time at far offset,

$$t = \tau + q\frac{h^2}{h_{max}^2}\,,$$

then it is clear that the parameter q is nothing else than the moveout in seconds at the far offset trace. In Figures (8.56) and (8.57) we portrayed a primary and a multiple before and after parabolic Radon transform filtering. In this example, q is residual moveout at far offset.

8.4 High-resolution Parabolic Radon Transform

The high-resolution Parabolic Radon transform proposed by Sacchi and Ulrych (1995) entails utilizing a regularization technique that leads to an operator that does not exhibit a Toeplitz structure. In the original formulation of the high-resolution Radon transform, the Radon operator is inverted via Cholesky decomposition. This is quite expensive compared to the classical least squares Radon transform that uses the Levinson recursion to invert a Toeplitz form.

We can propose a method to achieve high resolution at a computational cost of the order of the conventional parabolic least-squares Radon transform. This feature makes our new algorithm quite attractive for processing large data sets.

The Parabolic Radon transform is a widely accepted technique for multiple removal (Hampson, 1986). The method can be implemented in the

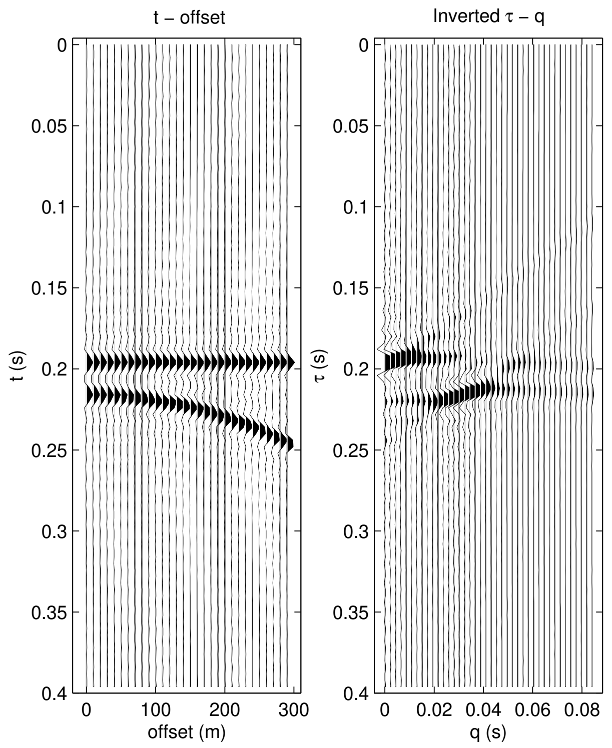


Figure 8.5: Left:A Primary and a multiple after NMO correction. Inverted $\tau - q$ panel.

frequency domain via a fast algorithm that exploits the Toeplitz structure of the least-squares Radon operator (Kostov, 1990; Darche, 1990). Recently, Sacchi and Ulrych (1995) proposed a high-resolution algorithm to increment the transform's ability to distinguish events with similar moveout curves. This algorithm is based on a procedure that attempts to find a sparse representation of the parabolic Radon domain's reflections. A similar algorithm has been proposed by Cary (1998). In this case, the Radon panel is constrained to be sparse in both the Radon parameter and the intercept time. The high-resolution parabolic Radon transform can be used to isolate multiples interferences within a few milliseconds of residual moveout at far offset. This is a problem frequently encountered when dealing with multiple short-period reflections generated by carbonate targets in the Western

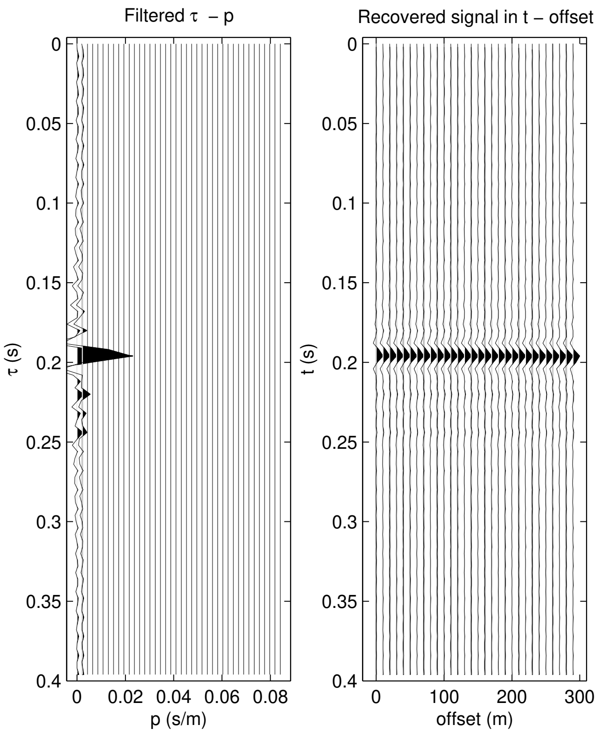


Figure 8.6: Left: Inverted $\tau - q$ panel after muting. Right: Data reconstructed by forward modelling the inverted/muted $\tau - q$ panel. In this example, the multiple has been eliminated by muting in the $\tau - q$ domain.

Canadian Basin (Hunt et al., 1996).

One of the advantages of the high-resolution parabolic Radon transform is that the focusing power of the transform is considerably increased with respect to the classical least-squares parabolic Radon transform. Unfortunately, the high resolution parabolic Radon transform leads to the inversion of an operator that is Hermitian but does not exhibit a Toeplitz structure. The resulting Hermitian operator is inverted using Cholesky decomposition. The Cholesky method for solving Hermitian linear systems of equations requires operations that is proportional to M^3, where M is the dimension of the Hermitian operator.

8.4.1 Least-squares Parabolic Radon transform

Common midpoint (CMP) gathers after normal moveout (NMO) correction can be modelled as a superposition of events with parabolic moveout:

$$d(x_j, t) = \sum_{k=1}^{M} m(q_k, \tau = t - q_k\, x_j^2)\,, j = 1, N\,, \tag{8.56}$$

where $d(x_j, t)$ denotes the CMP gather, x_j the offset, $m(q_k, \tau)$ is the Radon panel, q_k the discrete Radon parameter and τ the intercept time. The data consist of N seismic traces which do not need to be regularly sampled. The Radon parameter is uniformly discretized according to $q_k = q_0 + \Delta q\,(k - 1)$, $k = 1, \ldots, M$.
Equation (8.56) is essentially a decomposition of the CMP gather in terms of parabolic events distributed in the plane τ, q. It is computationally more convenient to rewrite the last equation in the frequency-offset domain. Taking Fourier to transform with respect to the temporal variable t, we arrive at the following expression

$$d(x_j, f) = \sum_{k=1}^{M} m(q_k, f)\, e^{i2\pi f q_k x_j^2}\,, j = 1, \ldots, N\,. \tag{8.57}$$

The calculations can be carried out independently for each frequency f. Equation (8.57) can be written in matrix form as follows:

$$\mathbf{d}(f) = \mathbf{L}(f)\,\mathbf{m}(f)\,. \tag{8.58}$$

To avoid notational clutter, we will drop the frequency dependency in equation (8.58) and write $\mathbf{d} = \mathbf{L}\,\mathbf{m}$.

The least-squares Radon operator is estimated by minimizing the following cost function.

$$J = ||\mathbf{d} - \mathbf{L}\,\mathbf{m}\,||^2 + \mu||\mathbf{m}||^2\,. \tag{8.59}$$

The regularization term $\mu||\mathbf{m}||^2$ is used to control the roughness of the solution. It can be shown that this term is one of the major sources of amplitude smearing in the Radon panel (Sacchi and Ulrych, 1995).

Taking derivatives of J with respect to $\mathbf{m}$ and equating them to zero yields

$$\begin{aligned}(\mathbf{L}^H\mathbf{L} + \mu\mathbf{I})\mathbf{m} &= \mathbf{L}^H\mathbf{d} \\ &= \mathbf{m}_{adj}\,.\end{aligned} \tag{8.60}$$

In the last equation $\mathbf{m}_{adj}$ denotes the low-resolution Radon transform obtained using the adjoint or transpose operator $\mathbf{L}^H$. The least-squares solution becomes

$$\begin{aligned}\mathbf{m} &= (\mathbf{L}^H\mathbf{L} + \mu\mathbf{I})^{-1}\mathbf{m}_{adj} \\ &= (\mathbf{R} + \mu\mathbf{I})^{-1}\mathbf{m}_{adj}\,.\end{aligned} \tag{8.61}$$

At this point, some observations are in order. First, it is clear that $\mathbf{R} = \mathbf{L}^H\mathbf{L} + \mu\mathbf{I}$ is a Toeplitz form (Kostov, 1990), with elements are given by

$$\{\mathbf{R} + \mu\mathbf{I}\}_{l,m} = \sum_{k=1}^{N} e^{-i2\pi f\Delta q(l-m)x_k^2} + \mu\delta_{l,m}\,. \tag{8.62}$$

Solving this equation using the Levinson recursion requires approximately $4M^2+7M$ operations and storage of only the first row of the Toeplitz matrix (Marple, 1987). This feature yields a very efficient algorithm to compute the parabolic Radon transform.

8.4.2 High-resolution parabolic Radon transform

In the high resolution, parabolic Radon transform, the vector $\mathbf{m}$ is retrieved by solving the following equation:

$$(\mathbf{R} + \mathbf{W}^H\mathbf{W})\mathbf{m} = \mathbf{m}_{adj}\,. \tag{8.63}$$

The matrix $\mathbf{W}$ is a diagonal matrix with elements that depend on $\mathbf{m}$ (Sacchi and Ulrych, 1995). This leads to an iterative algorithm where $\mathbf{W}$ is bootstrapped from the result of a previous iteration. In general, the iterative procedure is not required if we can design $\mathbf{W}$ from a priori information. The matrix of weights $\mathbf{W}$ is a diagonal matrix with elements given by

$$\{\mathbf{W}\}_{l,m} = w_l\,\delta_{l,m},\;\; l,m = 1,\ldots,M\,. \tag{8.64}$$

The elements of the diagonal form $\mathbf{R}+\mathbf{W}^H\,\mathbf{W}$ become:

$$\{\mathbf{R}+\mathbf{W}^H\,\mathbf{W}\}_{l,m} = \sum_{k=1}^{N} e^{-i2\pi f \Delta q (l-m) x_k^2} \;+\; w_l^2 \delta_{l,m}\,. \qquad (8.65)$$

It is clear that the addition of a diagonal matrix with non-constant e elements has destroyed the Toeplitz structure of the operator. The above matrix can be inverted by the Cholesky method in operations proportional to M^3. From the computational point of view, it is more convenient to compute the Radon transform using a constant diagonal regularization (equation (8.60)). However, if we want to estimate a high-resolution Radon operator, the regularization term must be a diagonal form with non-constant elements. The elements of $\mathbf{W}$ are used to emphasize the Radon parameters q_k that need to be constrained to be zero. The matrix $\mathbf{W}$ is bootstrapped from the data iteratively. The procedure, as mentioned above, is described in Sacchi and Ulrych (1995).

In our synthetic example, the elements of the diagonal matrix $\mathbf{W}^H\mathbf{W}$ are given by

$$w_k^2 = \begin{cases} 100. & if \;\; q_k \notin Q \\ 0.0001 & if \;\; q_k \in Q\,, \end{cases} \qquad (8.66)$$

where Q indicates the set of parameters q_k where the reflections are localized. These weights can be interpreted as the inverse of variance in model space. If w_l^2 is large, $1/w_l^2$ is small, and therefore, the algorithm will constrain the areas of no reflections in the τ, q space to be zero. It is clear that the resolution is enhanced by inhibiting the creation of smearing in the Radon panel.

8.4.3 Conjugate gradients and circulant matrices

To solve equation (7.63) we adopt the method of conjugate gradients, which is summarized below.

We want to solve $(\mathbf{R}+\mathbf{D})\mathbf{m} = \mathbf{m}_{adj}$, where $\mathbf{D} = \mathbf{W}^H\mathbf{W}$.

Start with an initial solution $\mathbf{m_0}$, set $\mathbf{p}_0 = \mathbf{r}_0 = \mathbf{m}_{adj} - (\mathbf{R}+\mathbf{D})\mathbf{m}_0$, $\alpha_{i+1} = (\mathbf{r}_i, \mathbf{r}_i)/(\mathbf{p}_i, (\mathbf{R}+\mathbf{D})\mathbf{p}_i)$

$\mathbf{m}_{i+1} = \mathbf{m}_i + \alpha_{i+1}\mathbf{p}_i$ $\mathbf{r}_{i+1} = \mathbf{r}_i - \alpha_{i+1}(\mathbf{R}+\mathbf{D})\mathbf{p}_i$
$\beta_{i+1} = (\mathbf{r}_{i+1}, \mathbf{r}_{i+1})/(\mathbf{r}_i, \mathbf{r}_i)$
$\mathbf{p}_{i+1} = \mathbf{r}_{i+1} + \beta_{i+1}\mathbf{p}_i$
where $i = 0, 1, 2, \dots K$ denotes the iteration number.

The cost of the conjugate gradients algorithm is dominated by multiplying a matrix by a vector. In general, matrix times vector multiplication is an $O(M^2)$ process. In our problem, we will use the Toeplitz structure of $\mathbf{R}$ to find a fast manner to compute the operation as mentioned earlier.

The product $(\mathbf{R}+\mathbf{D})\mathbf{x}$ can be decomposed into two products: $\mathbf{Rx} + \mathbf{Dx}$. The first product can be efficiently computed using the Fast Fourier Transform (FFT), the second product involves only $2M$ operations (M products plus M additions) and does not substantially increase the computational cost of the inversion.

The first product, $\mathbf{y} = \mathbf{Rx}$, is evaluated by augmenting the system as follows:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{y}' \end{bmatrix} = \mathbf{R}_{aug} \begin{bmatrix} \mathbf{x} \\ \mathbf{0} \end{bmatrix}, \tag{8.68}$$

where $\mathbf{R}_{aug}$ is the original Toeplitz matrix after being properly folded to become a circulant matrix (Strang, 1986; Schonewille and Duijndam, 1998). The right-hand side can be computed by multiplying the Fourier transform of the first row of $\mathbf{R}_{aug}$ by the Fourier transform of vector $[\mathbf{x}, \mathbf{0}]^T$, and taking the inverse Fourier transform of this product. Now our matrix times vector operation takes $O(M' \log M')$ operations where M' is the size of the augmented matrix ($M' = 2M$). We have found that the conjugate gradients algorithm convergences after a few iterations ($K \approx M/5$). Therefore, the inversion becomes an $O(K\, M'\, Log(M'))$ process. This is more efficient than the direct inversion of equation (7.63) by the Cholesky method.

8.4.4 Example

In Table 8.1 we present a comparison of CPU times in seconds for 3 different algorithms. The times in Table 1 correspond to the total computational cost for 512 frequencies. These simulations were performed on an SGI Origin 2000.

In both cases, we have 4 parabolic events which were mapped to the Radon domain u sing the following algorithms:

1. **Lev**: Classical least-squares parabolic Radon transforms are implemented via the Levinson recursion (valid for constant damping).
2. **Chol**: High-resolution Radon transform implemented via the Cholesky decomposition.
3. **CG+FFT**: High resolution parabolic Using the FFT, radon transforms are implemented via conjugate gradients plus matrix times vector multiplication.

It is clear that the new algorithm can achieve high resolution at a computational cost comparable to the one of the classical least squares Radon transform computed with the Levinson recursive solution.

In Figure (7.1), we portray the results obtained for the 256 $\times$ 256 simulation. Note that the differences between the high resolution Radon transforms computed with the Cholesky decomposition and the proposed algorithm are minimal.

$N \times M$	**Lev**	**Chol**	**CG+FFT**
128×128	2	6	3
256×256	8	42	12

Table 8.1: CPU times in seconds for the 3 algorithms tested in this study. N denotes the number of traces and M the number of q parameters.

8.5 Programs for Slant Stack and Parabolic Radon Transforms

The following two programs are MATLAB implementations of the Radon transform in $f-x$ and $f-p$. The inverse transform is solved using Least-

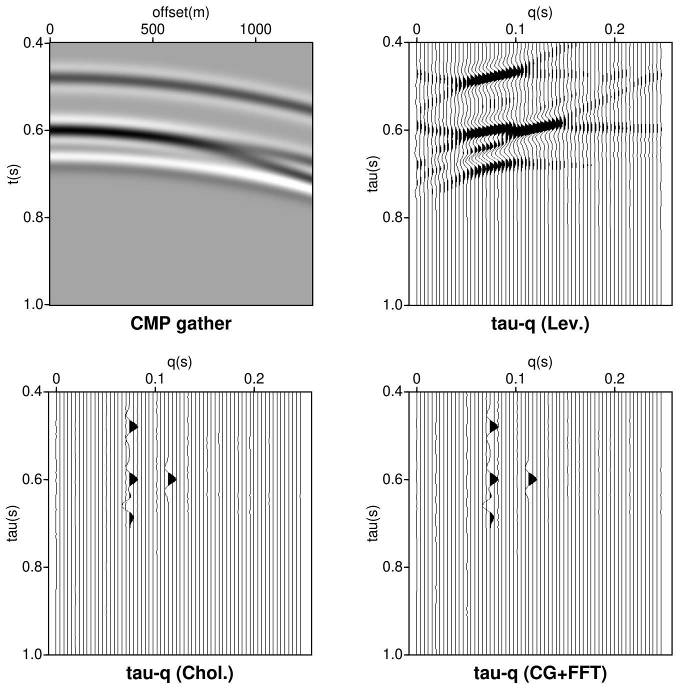


Figure 8.7: A synthetic CMP gather composed of 4 parabolic events is used to test 3 different algorithms to compute the Radon transform. **Lev.** indicates the classical solution using least squares with a constant damping term; the Levinson algorithm is used to invert the resulting Toeplitz form. **Chol.** indicates the high-resolution solution using non-constant damping (8)). This solution is computed through the Cholesky decomposition. **CG+FFT** indicates the proposed fast algorithm to compute the high-resolution Radon transform. In this example, the size of the Radon operator is 256×256. CPU times in seconds are given in Table 7.1

squares. The high-resolution implementation using circulant matrices is a little bit more tricky and requires more then a few lines of Matlab.

Forward Transform

Operator to compute the forward linear and parabolic Radon transform.

```
function [d]=for_taup(m,dt,h,q,N,flow,fhigh);
%INV_TAUP  An inverse Radon transforms. Given the seismic data, this
%       function computes the Radon panel by inverting the Radon operator
%
% [d] = for_taup(m,dt,h,q,N);
%
%
% IN  m: the Radon panel (d(nt,nq)
%    dt: sampling in sec
%    h(nh) offset or position of traces in mts
%    q(nq) ray parameters to retrieve or curvature
%       of the parabola if N=2
%    N:1 Linear tau-p
%    :2 Parabolic tau-p
%    flow, fhig: min and max freq. in Hz
%
% OUT d: the data
%
%
% SeismicLab
% Version 1
%
% written by M.D.Sacchi, last modified December 10, 1998.
% sacchi@phys.ualberta.ca
%
% Copyright (C) 1998 Signal Analysis and Imaging Group
%           Department of Physics
%           The University of Alberta
%

nt= max(size(m));
nh = max(size(h));


M = fft(m,[],1);
```

```
D = zeros(nt,nh);
i = sqrt(-1);

ilow = floor(flow*dt*nt)+1; if ilow<1; ilow=1;end;
ihigh = floor(fhigh*dt*nt)+1;
if ihigh>floor(nt/2)+1; ihigh=floor(nt/2)+1;end

for if=ilow:ihigh
f = 2.*pi*(if-1)/nt/dt;
L = exp(i*f*(h.^N)'*q);
x = M(if,:)';
y = L * x;
D(if,:) = y';
D(nt+2-if,:) = conj(y)';
end
D(nt/2+1,:) = zeros(1,nh);
d = real(ifft(D,[],1));

return;
```

Inverse transform

Operator to compute the LS inverse Radon transform. Notice that this is an "academic" implementation. A fast implementation involves replacing `inv` by a fast solver (i.e., Levinson's recursion).

```
function [m] = inv_taup(d,dt,h,q,N,flow,fhigh,mu);
%INV_TAUP  An inverse Radon transform. Given the seismic data,
%       this subroutine computes
%       the Radon panel by inverting the Radon operator
%
% [m] = inv_taup(d,dt,h,q,N,flow,fhigh,mu)
%
% IN  d: seismic traces (d(nt,nh)
%     dt: sampling in sec
%     h(nh) offset or position of traces in mts
%     q(nq) ray parameters to retrieve or curvature
%       of the parabola if N=2
%    N:1 Linear tau-p
%    :2 Parabolic tau-p
%    flow: freq. where the inversion starts in HZ (> 0Hz)
%    fhigh: freq. where the inversion ends in HZ  (> Nyquist)
%    mu: regularization parameter
%
% OUT m: the linear or parabolic tau-p panel
%
%
% SeismicLab
% Version 1
%
% written by M.D.Sacchi, last modified December 10, 1998.
% sacchi@phys.ualberta.ca
%
% Copyright (C) 1998 Signal Analysis and Imaging Group
%           Department of Physics
%           The University of Alberta
%

nt= max(size(d));
nq = max(size(q));
nh = max(size(h));

D = fft(d,[],1);
M = zeros(nt,nq);
i = sqrt(-1);

ilow = floor(flow*dt*nt)+1; if ilow<1; ilow=1;end;
ihigh = floor(fhigh*dt*nt)+1;
if ihigh>floor(nt/2)+1; ihigh=floor(nt/2)+1;end

for if=ilow:ihigh
f = 2.*pi*(if-1)/nt/dt;
L = exp(i*f*(h.^N)'*q);
```

```
y = D(if,:)';
x = L'*y;

MATRIX = L'*L;
tr=real(trace(MATRIX));
  Q =mu*tr*eye(nq);
  x = inv(MATRIX+Q) *L'* y;
M(if,:) = x';
M(nt+2-if,:) = conj(x)';
end
M(nt/2+1,:) = zeros(1,nq);
m = real(ifft(M,[],1));
return
```

8.6 Time variant velocity stacks (Hyperbolic time-domain Radon Transform)

We will discuss in this section the computation of time-variant operators that can be used as an alternative to the parabolic Radon transform.

The parabolic Radon transform is a time-invariant operator, therefore it can be implemented in the frequency domain. This trick permits one to solve several small problems, one at each frequency, instead of a large problem involving all the time-offset-velocity samples at the same time.

It is clear that in the case of the parabolic Radon transform time invariance is achieved by means of an approximation. It might happen that the parabolic the approximation is not properly satisfied and consequently, travel times (especially at far offsets) need to be properly modelled.

In this part of the course, we will focus our attention to the computational aspects of Hyperbolic Radon operators.

We have already mentioned that the data in the CDP domain can be modelled as a superposition of hyperbolas. User this assumption a hyperbolic stack operator can be used to map hyperbolas (reflections) into $time - velocity$ pairs. In other words, our operator is used to map data from $offset - time$ to $velocity - time$ space. In the new space, we can identify multiple reflections and filter them out. We can also use this type of operator to reduce random noise and to enhance the overall aspect of the seismic reflection, which might be hidden by a strong ground-roll (in a CSG) or any other type of deterministic noise.

The time-variant velocity-stack operator is defined in terms of summation along Dix hyperbolas, $m(\tau, v)$ is used to designate the velocity-stack and $d(t, h)$ the CMP gather:

$$d(t,h) = \int m(\tau = \sqrt{t^2 - h^2/v^2}, v)\, dv\,, \tag{8.69}$$

where h is source-receiver offset, t is two-way travel-time, v is the rms velocity, and τ is two-way vertical travel-time. After discretization and lexicographic arrangement, equation (8.69) can be written as

$$d = Lm\,. \tag{8.70}$$

The vectors d and m have $nt \times nh$ and $n\tau \times nv$ elements, respectively. The dimension of the operator L is $(nt \times nh) \times (nt \times n\tau)$. The forward or modelling operator, L, picks a wavelet in velocity space and produces a hyperbola in data space. The transpose operator L^T is a simple NMO followed by a stacking operator.
To find the inverse operator, we consider the problem

$$Minimize\ \{\phi = ||Lm - d||_2^2\}$$

Differentiating ϕ with respect to m yield the least-squares solution

$$\hat{m} = (L^T L)^{-1} L^T d = (L^T L)^{-1} m_0\,, \tag{8.71}$$

where for simplicity, we have assumed that L is full rank. In equation (8.71) m_0 is the low-resolution velocity stack computed by means of the adjoint or transpose operator (Sacchi and Ulrych, 1995). The velocity stack computed after inversion, $\hat{m}$, possesses more resolution that m_0. Unfortunately, the computation of $\hat{m}$ involves the inversion of $L^T\, L$. Assuming a typical CMP gather of 48 channels and 1000 samples per trace. In addition, suppose that 48 traces and 1000 samples were used to discretize the velocity, v, and the intercept time, τ, respectively. In this case, $L^T L$ has dimension 24000×24000. Direct methods cannot be applied to this type of problem.

8.6.1 The conjugate gradients algorithm

The trick here is to use a semi-iterative technique to find an approximate solution to our problem. The advantage of the CG algorithm is the matrix L does not need to be stored. L is not a matrix but an operation performed on a vector. To apply the CG algorithm, we must first define the operations L and L^T.

It is clear that L is an operator that picks a wavelet in the $\tau - v$ and produces a hyperbola in $t - h$. The operator L^T (the adjoint or transpose operator) does the opposite; it gathers information in $t-h$ along a hyperbolic path and collapses this information into a point in $\tau - v$.

Let us assume that we have a code capable of performing the following operations (as I have already mentioned L and L' do not need to be matrices)

$$y = Lx \quad x' = L^T y \,.$$

To solve the problem $||Lx - y||^2$ with an initial solution x_0, we use the following Conjugate Gradients (CG) algorithm:

```
Set initial values: $r = y - L\,x_0\,,\; g = L^T\,r\,,\; s = g$
for i = 1:$ITERMAX$ {
  $ss = Ls\,,\;\; \delta = ||ss||^2$
  $\alpha = \gamma/\Delta$
  $x = x + \alpha\, s$
  $r = r - \alpha\, ss$
  $g = L^T\, r$
  $s = g + \beta\, s$
```

- }

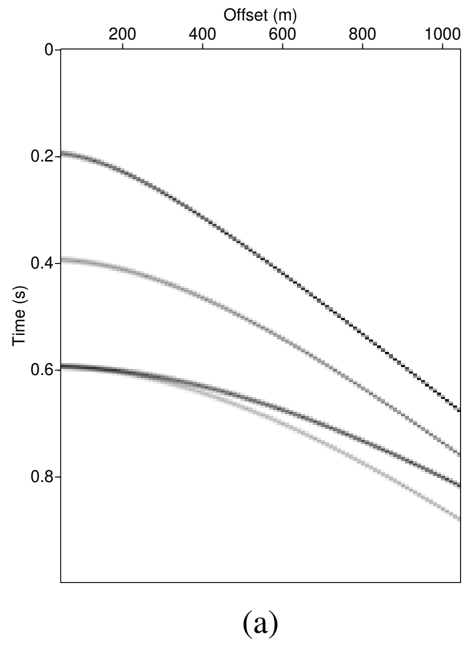
The CG algorithm will find the least squares solution for the over-determined problem in N iterations where N is the total number of observations. In the under-determined problem, the CG converges to the minimum norm solution. The technique gives the exact answer for exact arithmetics, but of course, round-off errors will affect the convergence of the algorithm. This is why the CG is often referred as a semi-iterative technique.

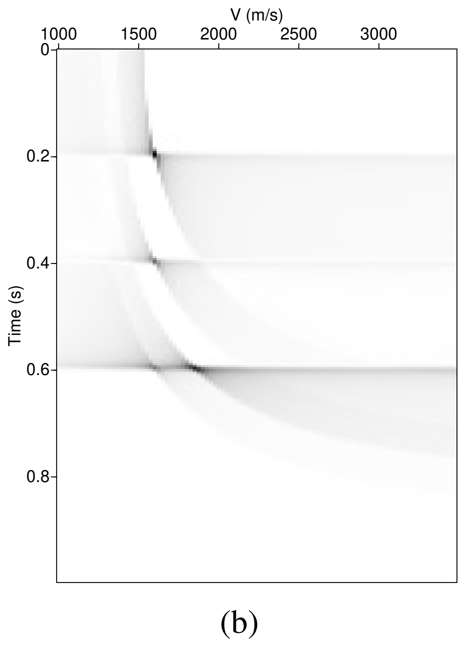
In the computation of the velocity stacks, we will use on a few iterations. How many iterations?. We can say that we will use enough iterations to properly model the hyperbolic events. The CG method allows us to explore our solution efficiently by stopping the algorithm at any number of iteration and then, if the solution is not optimal, we can re-start the algorithm until a satisfactory misfit is obtained.

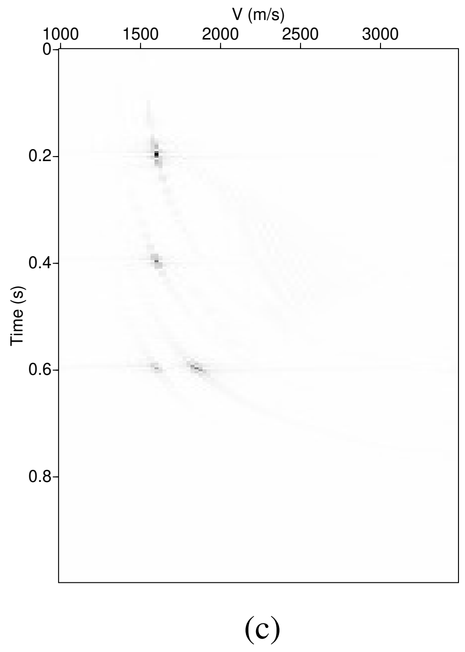
8.6.2 Example

We will analyze the performance of the CG with synthetic and real data examples. In Figure (8.8a), we portray a synthetic CMP gather. The model comprises 2 primaries reflections of 1500m/s (water column) and a primary of 1700 m/s at 0.65 s. In Figure (8.8b), we portray the velocity gather obtained using the adjoint operator. This gather does not offer enough resolution to properly identify and separate the multiple events at 0.65 s from the primary. In Figure (8.8c), we portray the velocity obtained after inverting the data using the CG algorithm. Figure (8.8d) is the primary obtained after muting the velocity gather.

In Figure (8.9), I displayed the velocity gather obtained via the CG algorithm after amplitude clipping. In this panel, we also portray the artifacts that arise from finite aperture and sampling (alias).







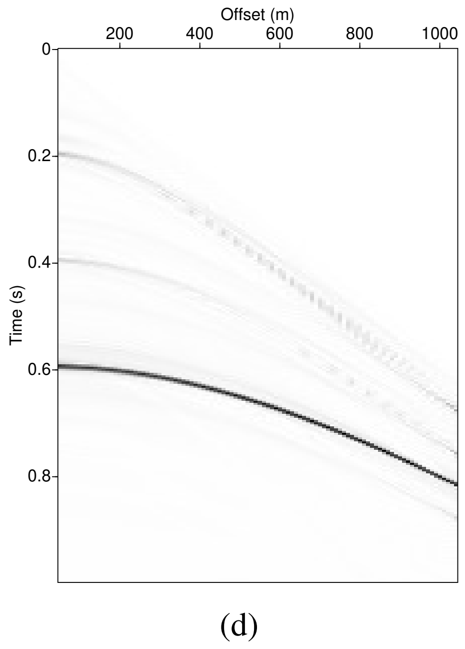


Figure 8.8: (a) Synthetic data. (b) Velocity gather obtained using the adjoint operator. (c) The velocity gather computed using the least-squares inversion. (d) Recovered data (primary) obtained after the de-multiple process.

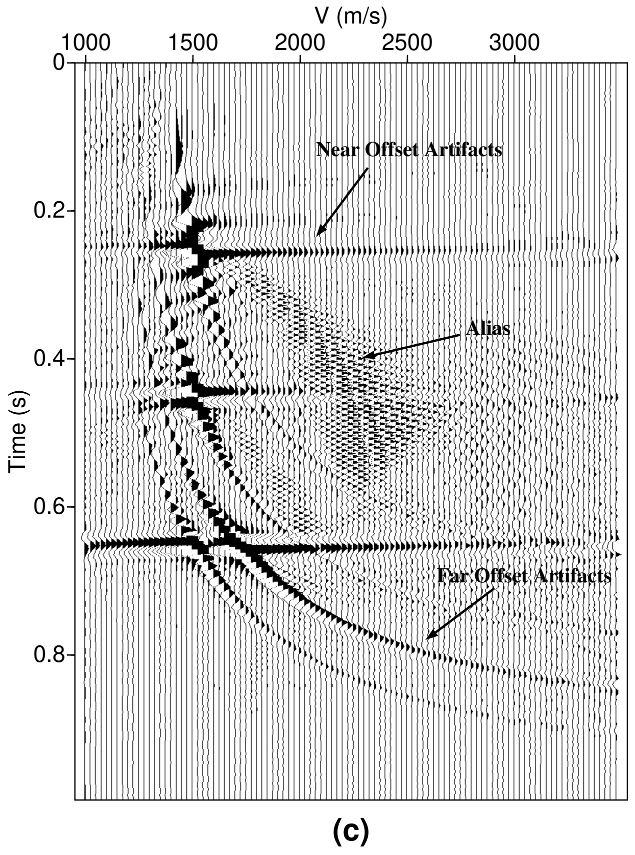


Figure 8.9: Clipped version of Figure (8.8)b showing finite aperture and sampling (alias) artifacts.

The following subroutine permits to compute forward and adjoint Hyperbolic Radon operators. I have included a linear interpolation step. See how do I define the adjoint of interpolation. It is important to stress that the forward and adjoint pairs must pass the dot product test. Otherwise, the CG inversion will not work. Once you have your forward and adjoint operators encapsulated in a subroutine it's quite simple to put together a CG inversion code.

```
      subroutine Hyperbolic_Radon(dt,nt,v,nv,h,nh,m,d,c)
c
c  Compute velocity panels  when c = 'a' (Adjoint Hyper. Radon)
c  Compute CMP gathers when      c = 'f' (Forward Hyper. Radon)
c                                c is character * 1
c INPUT
c  dt : sampling in sec
c  nt : number of time samples (also number of tau samples)
c  v(nv) : axis of the Radon panel (velocity in m/s). It can be
c          changed by 1/vel^2 or moveout at far offset
c  h(nh) : offset in meters
c          h(1) is offset of trace 1, h(2) is offset of trace 2....
c
c INPUT/OUTPUT
c  d(nh,nt) : cmp or super-cmp  input if c = 'a'
c  m(nv,nt) : Radon panel       output if c = 'a'
c
c  d(nh,nt) : cmp or super-cmp  output if c = 'f'
c  m(nv,nt) : Radon panel       input if c = 'f'
c

      real        d(300,2000), m(300,2000),h(300)
      real        v(300)

      character * 1 c

      if(c.eq.'a') call clean(m,nv,nt)  ! initialize m with zeros
      if(c.eq.'f') call clean(d,nh,nt)  ! initialize d with zeros

      do ih =1,nh
       do iv=1,nv
        do itau=1,nt

         ttt=(itau-1)*dt
         time=sqrt(ttt**2+(h(ih)/v(iv))**2)
         it1 = int(time/dt)
         a = time/dt - float(it1)       !Coeff. of the linear interp.
         it2 = it1 + 1

      if(it1.lt.nt.and.it1.ge.1) then
       if(c.eq.'a') m(iv,itau) = m(iv,itau)+(1.-a)*d(ih,it1)+a*d(ih,it2)
       if(c.eq.'f') d(ih,it1)  = d(ih,it1)+(1.-a)*m(iv,itau)
       if(c.eq.'f') d(ih,it2)  = d(ih,it2)+    a *m(iv,itau)
      endif

        enddo    ! end offset loop
       enddo    ! end velocity loop
      enddo    ! end tau loop

      return
      end
```

8.7 High Resolution Radon Transform

We can construct a solution m that consists of a few isolated spikes in velocity space. This is what we often call a sparse solution. We have already outlined a procedure to compute sparse solutions using the Parabolic Radon transform. In that case, the spareness constraint was used to invert the Radon operator in the frequency domain. When using hyperbolic Radon transforms, the sparseness constraint has to be imposed in the $\tau - v$ domain. In general, one can use any measure of sparseness (we have seen various norms that can be used to retrieve sparse models when dealing with impedance inversion in Chapter 4). Let's assume we use a Cauchy-like norm (Sacchi and Ulrych, 1995). In this case, we minimize

$$J = ||Lm - d||_2^2 + \mu \sum_k \ln(1 + m_k^2/b) \tag{8.72}$$

where m_k indicates an element of $m(\tau, v)$ after lexicographic arrangement (transformation of a matrix into a vector). The parameters μ and b are the hyper-parameters of the problem.

Taking derivatives of J with respect to m_k and equation them to zero leads to the following system

$$L^T\,L\,m - L^T\,d + Q\,m = 0 \tag{8.73}$$

where Q is a diagonal matrix with elements given by

$$Q_i = \frac{2\,\mu}{b + m_i^2}$$

It is clear that the system needs to be solved in an iterative manner (Q depends on the unknown model m). We can rewrite our solution as follows:

$$m^k = (L^T\,L + Q^{k-1})^{-1}\,L^T d\,. \tag{8.74}$$

where k indicates the iteration. The matrix of weights Q is computed from the result of the previous iteration. In general, one solve the problem for a given matrix of weights Q using CG, then after enough iteration to reach convergence, Q is updated and a CG is run again to solve the linear problem. The procedure is continues until we find the minimum of the cost

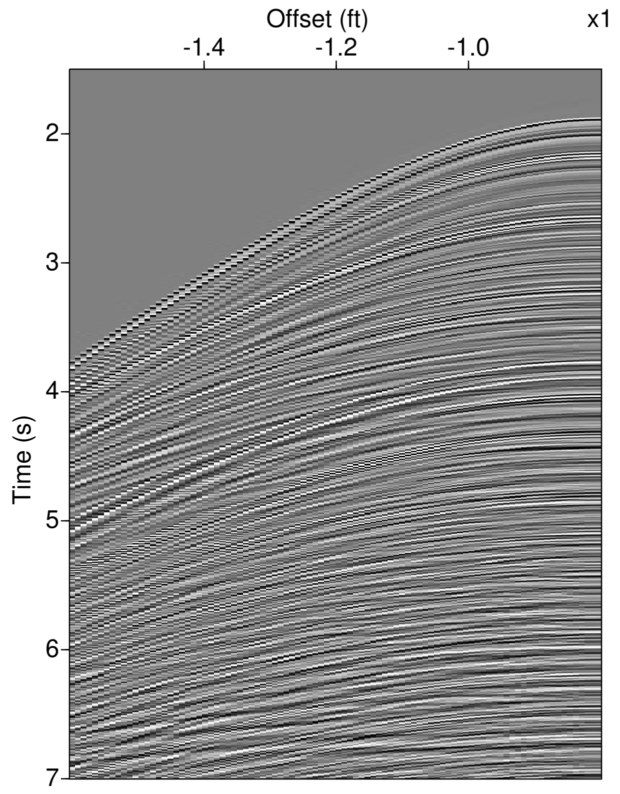


Figure 8.10: CMP gather # 1000 from a data set from the Gulf of Mexico.

function J. This algorithm can be very expensive, and in general a good felling for the μ and b is required to reach a sparse solution. When working with real data the parameters needed for the inversion (μ, b, number of iterations) are estimated by trial an error from a single CMP gather, the same parameters are used to invert the rest of the CMPs in the seismic volume.

In Figures (8.10), (8.11), (8.12), and (8.13) we test the high resolution hyperbolic Radon transform with a data set from the Gulf of Mexico (data provided by Western Geophysical to test multiple attenuation codes).

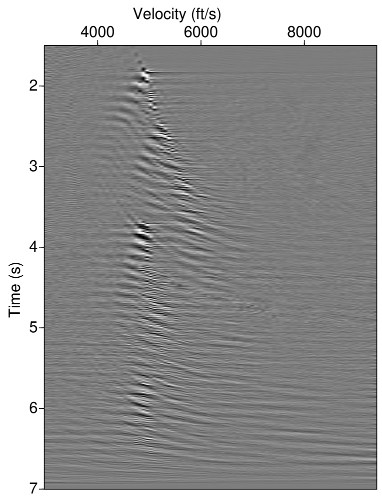


Figure 8.11: Velocity panel obtained by inversion of the Hyperbolic Radon transform using least-squares. CMP gather # 1000 from a data set from the Gulf of Mexico.

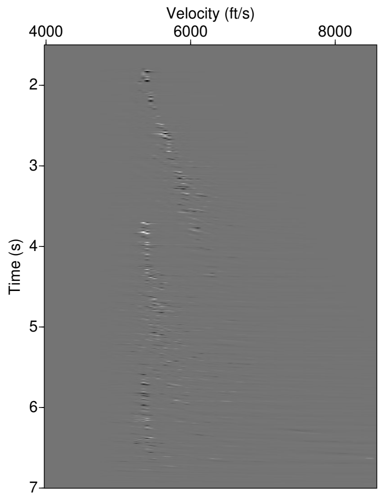


Figure 8.12: Velocity panel obtained by inversion of the Hyperbolic Radon transform using sparse inversion. CMP gather # 1000 from a data set from the Gulf of Mexico.

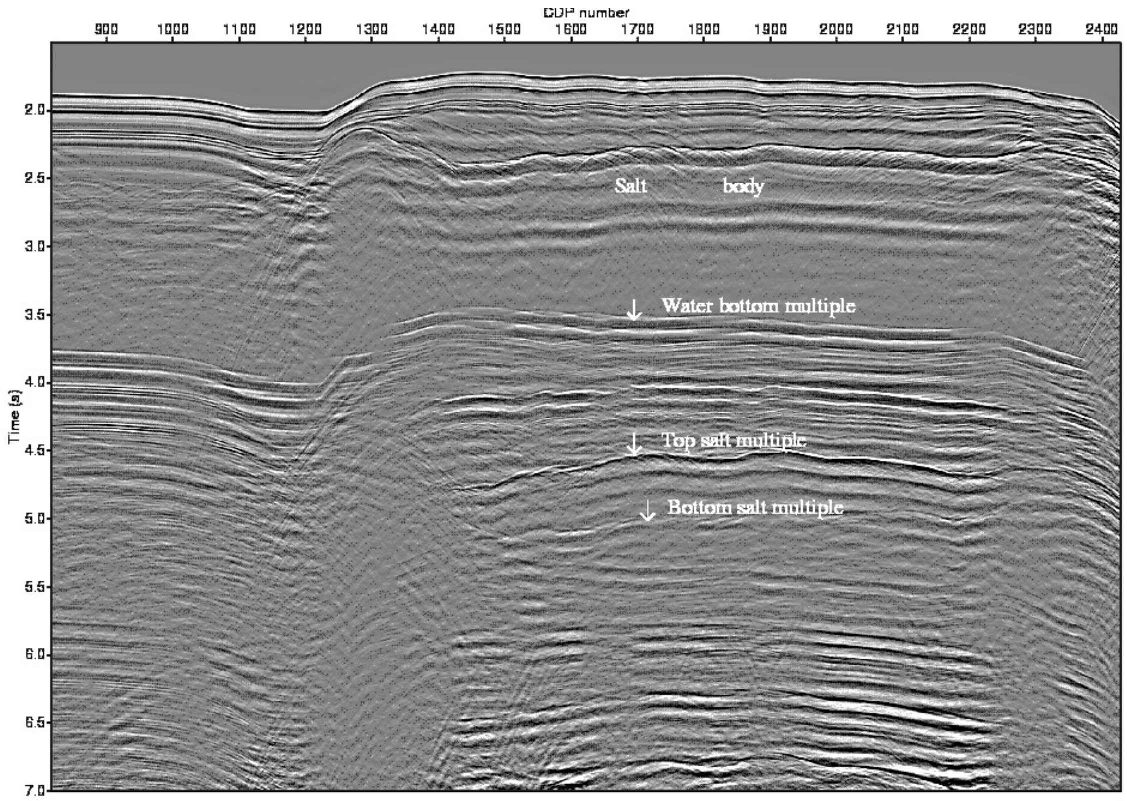


Figure 8.13: Stack section of the Gulf of Mexico data set before multiple removal.

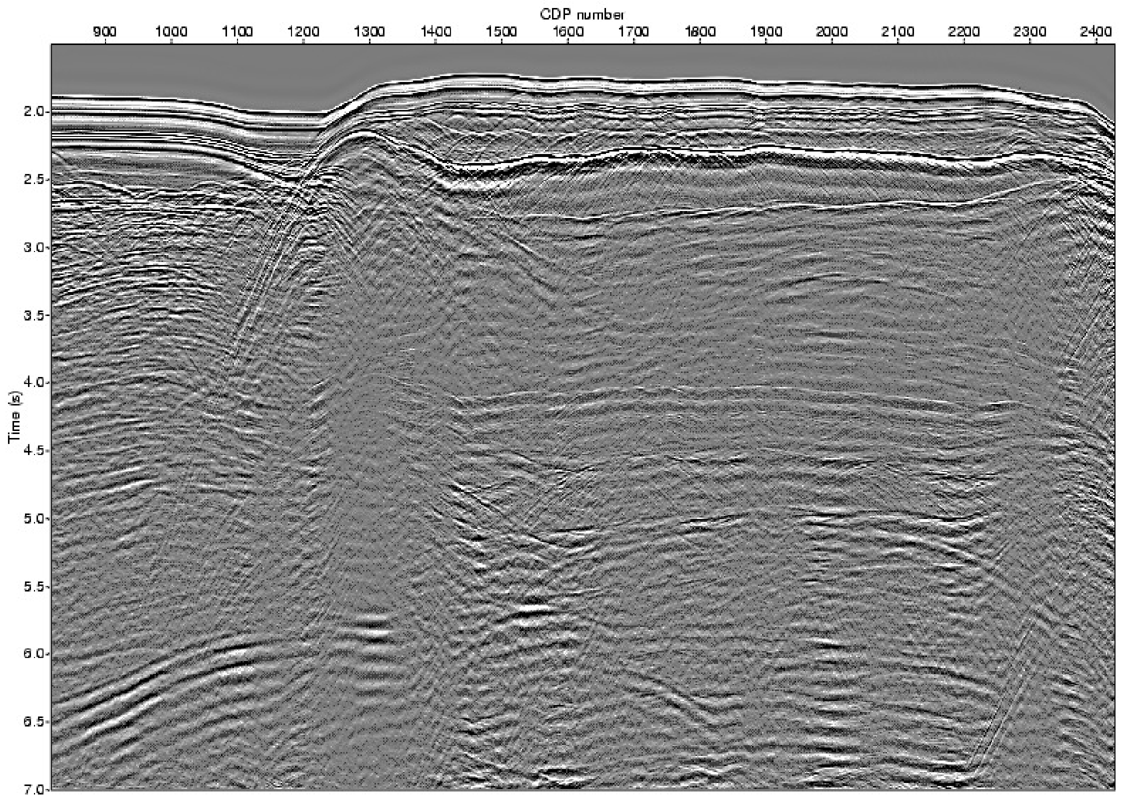


Figure 8.14: Stack section after multiple removal.

8.8 Interpolation problems

The parabolic and hyperbolic Radon transform can be used to interpolate CMP gathers. This is quite simple and entails mapping back the velocity stack to a data space using a new geometry. To interpolate pre-stack data in receiver-source space (or midpoint-offset) a more sophisticated approach is required. Various research group in the area of signal analysis have proposed algorithms to interpolate 1D data. These algorithms assume that the data are band-limited. One can extend these ideas to the problem of reconstructing pre-stack data. In geophysics Duijndam et. al (1999) and Hindriks et. al (1997) have introduced a least-squares algorithm to invert the fourier transform of the data. We will review some basic features of these algorithms and introduce a regularization term that enables us to recover large gaps in our pre-stack data set.

We define the discrete 2-D inverse Fourier transformation in source and receiver coordinates as

$$u(x_s, x_r, \omega) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} U(k_s(m), k_r(n), \omega) e^{jk_s(m)x_s} e^{jk_r(n)x_r}, \quad (8.75)$$

where x_s and x_r are the spatial variables along source and receiver coordinates, k_s and k_r are the corresponding wave-numbers and ω is the temporal frequency. Equation (8.75) gives rise to a linear system equations

$$u = AU \quad (8.76)$$

where

$$A_{mn} = \frac{1}{MN} e^{jk_s(m)x_s} e^{jk_r(n)x_r}, \quad (8.77)$$

u and U denote the known data and unknown coefficients of the DFT, respectively.

Therefore, the interpolation problem can be posed as finding, from the incomplete data, the 2D-DFT (U) by solving

$$u = AU + n \quad (8.78)$$

where n denotes the noise in the data. A unique solution may be obtained by minimizing the following expression

$$J = \| AU - u \|_2^2 + \epsilon \| U \|_2^2 \qquad (8.79)$$

and the solution can be shown to take the form:

$$\hat{U} = (A^T A + \epsilon I)^{-1} A^T u, \qquad (8.80)$$

where T denotes the transpose of a matrix.

Next, we derive a similar result but using a weighted DFT-domain norm introduced in the previous section. In this case the function to be minimized is

$$J = \| AU - u \|_2^2 + \epsilon \| U \|_P^2 \qquad (8.81)$$

The solution takes the form:

$$\hat{U} = (A^T A + DI)^{-1} A^T u, \qquad (8.82)$$

where D is a diagonal matrix with diagonal elements corresponding to $\frac{\epsilon}{|P(k_s,k_r)|^2}$ and $|P(k_s, k_r)|^2$ is a vector that contains the amplitude spectrum of U in lexicographic form. Ideally, one should know the amplitude spectrum of the data. Unfortunately, U is the unknown of our problem. The latter can be overcome by defining an initial D in terms of the DFT of the irregularly sampled data $A^T u$ and smoothing the result to attenuate the artifacts introduced by the irregularity of u (Ning and Nikias, 1990).

The scheme can be summarized as follows:

- Start with an initial $\hat{U}$.
- Compute $D = S(\hat{U}^* \hat{U})$, where S is a smoothing filter.
- Solve $\hat{U} = (A^T A + DI)^{-1} A^T u$ using Conjugate Gradients.
- Iterate until convergence.

An example of reconstruction is demonstrated on 15 synthetic shot gathers. Figure 8.15 shows six of the shot gathers with the shots #3 and #7 removed. The reconstruction is performed using the minimum weighted norm method with adaptive weights. Figure 8.16 shows the reconstructed shot gathers (only six shots are shown). The missing shots have been completely reconstructed.

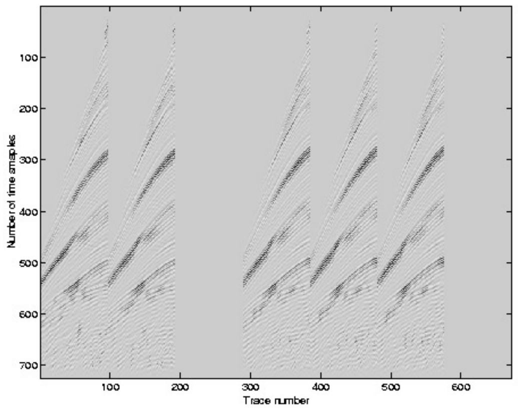


Figure 8.15: Six shot gathers with shots #3 and #7 removed.

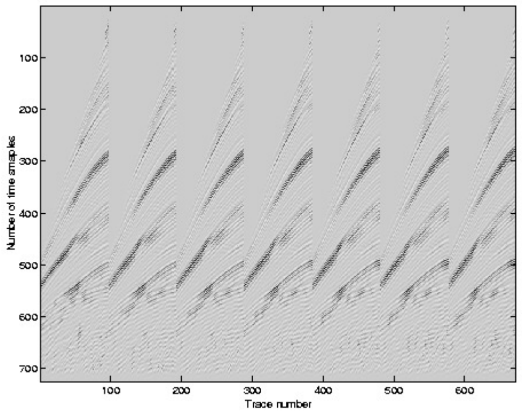


Figure 8.16: Reconstructed shot gathers using the minimum weighted norm algorithm.

8.9 References

Radon

Beylkin, G., 1987, Discrete radon transform: IEEE Trans. Acoust., Speech, Signal Processing., **ASSP-35**, 162-172.

Cary, P., W., 1998, The simplest discrete Radon transform, 68th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1999-20 02.

Chapman, C. H., 1981, Generalized Radon transforms and slant stacks: Geophys. J. Roy. Astr. Soc., **54**, 481-518.

Deans, S. R., 1983, The Radon transform and some of its applications: J. Wil ey and Sons, Inc.

Durrani, T. S., and Bisset, D., 1984, The Radon transform and its properties : Geophysics, **49**, 1180-1187.

Foster, J. D., and Mosher, C. C., 1992, Suppression of multiples reflection using the Radon transform: Geophysics, **57**, 386-395.

Hampson, D., 1986, Inverse velocity stacking for multiple elimination: J. Can. Soc. Expl. Geophys., **22**, 44-55.

Hunt, L., Cary, P., and Upham, W., 1996, An improved Radon Transform for short period multiple attenuation: CSEG 23rd Annual Mtg., Expanded Abstracts, 58-59.

Kostov, C., 1990, Toeplitz structure in slant-stack inversion: 60th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1618-1621.

Phinney, R. A., Chowdhury, K. R., and Frazer, L. N., 1981 Transformation and analysis of record sections: J. Geophys. Res., **86**, 359-377.

Pratt, W. K., 1991, Digital image processing: John Wiley and Sons, Inc

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

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

Yilmaz, Ö., 1989, Velocity-stack processing: Geophys. Prosp., **37**, 357-382.

Yilmaz, O., and Taner, M. T., 1994, Discrete plane-wave decomposition by least-mean-square-error method: Geophysics, **59**, 973-982.

Zhou, B., and Greenhalgh, S. A., 1994, Linear and parabolic $\tau - p$ revisited: Geophysics, **59**, 1133-1149.

FT Interpolation

Duijndam, A.J.W., Schonewille, M., and Hindriks, K., 1999, Reconstruction of seismic signals, irregularly sampled along on spatial coordinate: Geophysics, 64, 524-538.

Hindriks, K. O. H., Duijndam, A. J. W. and Schonewille, M. A., 1997, Reconstruction of two-dimensional irregularly sampled wavefields: Annual Meeting Abstracts, Society Of Exploration Geophysicists, 1163-1166.

Ning, T. and Nikias, C.L., 1990, Power Spectrum Estimation with Randomly Spaced Correlation Samples: IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 38, no. 6, 991-997.

Sacchi, M.D. and Ulrych, T.J., 1998, Interpolation and extrapolation using a high-resolution discrete Fourier transform: IEEE Trans. Signal Processing, vol. 46, no. 1, 31-38.