Chapter 7 *Pulse Compression*

Range resolution for a given radar can be significantly improved by using very short pulses. Unfortunately, utilizing short pulses decreases the average transmitted power, which can hinder the radar's normal modes of operation, particularly for multi-function and surveillance radars. Since the average transmitted power is directly linked to the receiver SNR, it is often desirable to increase the pulse width (i.e., increase the average transmitted power) while simultaneously maintaining adequate range resolution. This can be made possible by using pulse compression techniques. Pulse compression allows us to achieve the average transmitted power of a relatively long pulse, while obtaining the range resolution corresponding to a short pulse. In this chapter, we will analyze analog and digital pulse compression techniques.

Two analog pulse compression techniques are discussed in this chapter. The first technique is known as "correlation processing" which is dominantly used for narrow band and some medium band radar operations. The second technique is called "stretch processing" and is normally used for extremely wide band radar operations. Digital pulse compression will also be briefly presented.

7.1. Time-Bandwidth Product

Consider a radar system that employs a matched filter receiver. Let the matched filter receiver bandwidth be denoted as B. Then, the noise power available within the matched filter bandwidth is given by

$$N_i = 2 \frac{N_0}{2} B \tag{7.1}$$

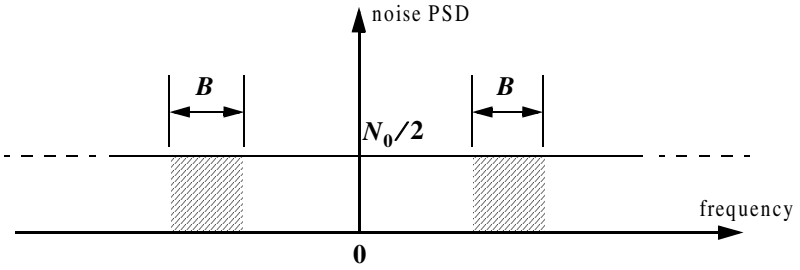

Figure 7.1. Input noise power.

where the factor of two is used to account for both negative and positive frequency bands, as illustrated in Fig. 7.1. The average input signal power over a pulse duration τ' is

$$S_i = \frac{E}{\tau'} \tag{7.2}$$

E is the signal energy. Consequently, the matched filter input SNR is given by

$$(SNR)_i = \frac{S_i}{N_i} = \frac{E}{N_0 B\tau'} \tag{7.3}$$

Using Eqs. (6.18) (from Chapter 6) and (7.3), one may compute the output peak instantaneous SNR to the input SNR ratio as

$$\frac{SNR(t_0)}{(SNR)_i} = 2B\tau' \tag{7.4}$$

The quantity $B\tau'$ is referred to as the "time-bandwidth product" for a given waveform, or its corresponding matched filter. The factor $B\tau'$ by which the output SNR is increased over that at the input is called the matched filter gain, or simply the compression gain.

In general, the time-bandwidth product of an unmodulated pulse approaches unity. The time-bandwidth product of a pulse can be made much greater than unity by using frequency or phase modulation. If the radar receiver transfer function is perfectly matched to that of the input waveform, then the compression gain is equal to $B\tau'$. Clearly, the compression gain becomes smaller than $B\tau'$ as the spectrum of the matched filter deviates from that of the input signal.

7.2. Radar Equation with Pulse Compression

The radar equation for a pulsed radar can be written as

$$SNR = \frac{P_t \tau' G^2 \lambda^2 \sigma}{(4\pi)^3 R^4 k T_e F L} \tag{7.5}$$

where P_t is peak power, τ' is pulse width, G is antenna gain, σ is target RCS, R is range, k is Boltzman's constant, T_e is effective noise temperature, F is noise figure, and L is total radar losses.

Pulse compression radars transmit relatively long pulses (with modulation) and process the radar echo into very short pulses (compressed). One can view the transmitted pulse to be composed of a series of very short subpulses (duty is 100%), where the width of each subpulse is equal to the desired compressed pulse width. Denote the compressed pulse width as τ_c. Thus, for an individual subpulse, Eq. (7.5) can be written as

$$(SNR)_{\tau_c} = \frac{P_t \tau_c G^2 \lambda^2 \sigma}{(4\pi)^3 R^4 k T_e F L} \tag{7.6}$$

The SNR for the uncompressed pulse is then derived from Eq. (7.6) as

$$SNR = \frac{P_t (\tau' = n\tau_c) G^2 \lambda^2 \sigma}{(4\pi)^3 R^4 k T_e F L} \tag{7.7}$$

where n is the number of subpulses. Equation (7.7) is denoted as the radar equation with pulse compression.

Observation of Eqs. (7.5) and (7.7) indicates the following (note that both equations have the same form): For a given set of radar parameters, and as long as the transmitted pulse remains unchanged, then the SNR is also unchanged regardless of the signal bandwidth. More precisely, when pulse compression is used, the detection range is maintained while the range resolution is drastically improved by keeping the pulse width unchanged and by increasing the bandwidth. Remember that range resolution is proportional to the inverse of the signal bandwidth,

$$\Delta R = c/2B \tag{7.8}$$

7.3. Analog Pulse Compression

Correlation and stretch pulse compression techniques are discussed in this section. Two MATLAB programs which execute digital implementation of both techniques (using the FFT) are also presented.

7.3.1. Correlation Processor

In this case, pulse compression is accomplished by adding frequency modulation to a long pulse at transmission, and by using a matched filter receiver in order to compress the received signal. As an example, we saw in Chapter 6 that using LFM within a rectangular pulse compresses the matched filter output by a factor $\xi = B\tau'$, which is directly proportional to the pulse width and bandwidth. Thus, by using long pulses and wideband LFM modulation we can achieve large compression ratios. This form of pulse compression is known as "correlation processing."

Fig. 7.2 illustrates the advantage of pulse compression. In this example, an LFM waveform is used. Two targets with RCS $\sigma_1 = 1m^2$ and $\sigma_2 = 0.5m^2$ are detected. The two targets are not separated enough in time to be resolved. Fig. 7.2a shows the composite echo signal from those targets. Clearly, the target returns overlap and, thus, they are not resolved. However, after pulse compression the two pulses are completely separated and are resolved as two targets. In fact, when using LFM, returns from neighboring targets are resolved as long as they are separated, in time, by τ_{n1}, the compressed pulse width.

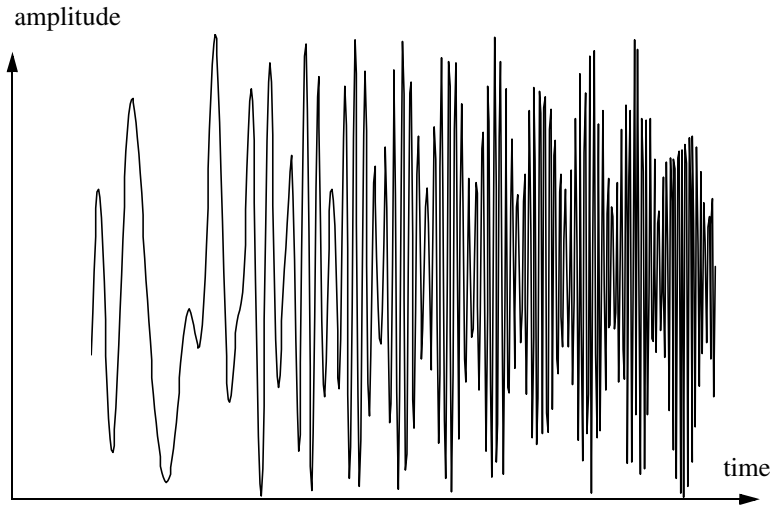

Figure 7.2a. Composite echo signal for two unresolved targets.

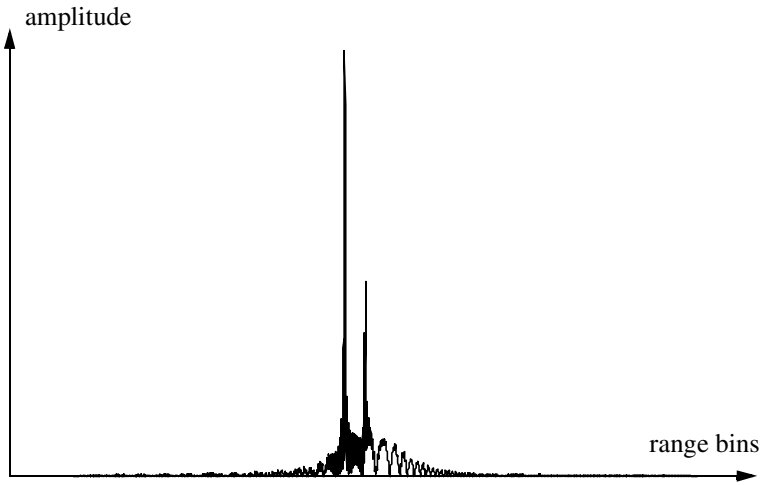

Figure 7.2b. Composite echo signal corresponding to Fig. 7.2a, after pulse compression.

Radar operations (search, track, etc.) are usually carried out over a specified range window, referred to as the receive window and defined by the difference between the radar maximum and minimum range. Returns from all targets within the receive window are collected and passed through a matched filter circuitry to perform pulse compression. One implementation of such analog processors is the Surface Acoustic Wave (SAW) devices. Because of the recent advances in digital computer development, the correlation processor is often performed digitally using the FFT. This digital implementation is called Fast Convolution Processing (FCP) and can be implemented at base-band. The fast convolution process is illustrated in Fig. 7.3

Since the matched filter is a linear time invariant system, its output can be described mathematically by the convolution between its input and its impulse response,

$$y(t) = s(t) \bullet h(t) \tag{7.9}$$

where $s(t)$ is the input signal, $h(t)$ is the matched filter impulse response (replica), and the $\bullet$ operator symbolically represents convolution. From the Fourier transform properties,

$$FFT\{s(t) \bullet h(t)\} = S(f) \cdot H(f) \tag{7.10}$$

And when both signals are sampled properly, the compressed signal $y(t)$ can be computed from

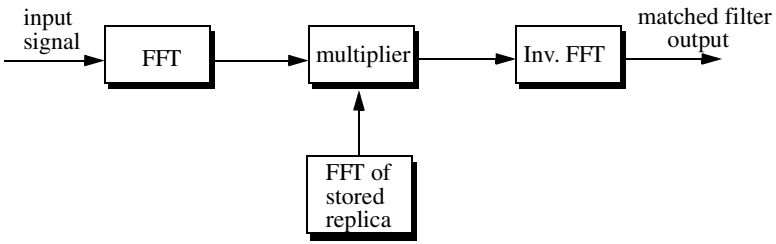

Figure 7.3. Computing the matched filter output using an FFT.

$$y = FFT^{-1}\{S \cdot H\} \tag{7.11}$$

where FFT^{-1} is the inverse FFT. When using pulse compression, it is desirable to use modulation schemes that can accomplish a maximum pulse compression ratio, and can significantly reduce the side lobe levels of the compressed waveform. For the LFM case the first side lobe is approximately $13.4dB$ below the main peak, and for most radar applications this may not be sufficient. In practice, high side lobe levels are not preferable because noise and/or jammers located at the side lobes may interfere with target returns in the main lobe.

Weighting functions (windows) can be used on the compressed pulse spectrum in order to reduce the side lobe levels. The cost associated with such an approach is a loss in the main lobe resolution, and a reduction in the peak value (i.e., loss in the SNR), as illustrated in Fig. 7.4. Weighting the time domain transmitted or received signal instead of the compressed pulse spectrum will theoretically achieve the same goal. However, this approach is rarely used, since amplitude modulating the transmitted waveform introduces extra burdens on the transmitter.

Consider a radar system that utilizes a correlation processor receiver (i.e., matched filter). The receive window in meters is defined by

$$R_{rec} = R_{max} - R_{min} \tag{7.12}$$

where R_{max} and R_{min}, respectively, define the maximum and minimum range over which the radar performs detection. Typically R_{rec} is limited to the extent of the target complex. The normalized complex transmitted signal has the form

$$s(t) = \exp\left(j2\pi\left(f_0 t + \frac{\mu}{2}t^2\right)\right) \qquad 0 \le t \le \tau' \tag{7.13}$$

τ' is the pulse width, $\mu = B/\tau'$, and B is the bandwidth. Note that this definition of the LFM pulse is different from that in Chapter 6. Earlier, f_0 denoted the chirp center frequency and in Eq. (7.13) it denotes the chirp start frequency.

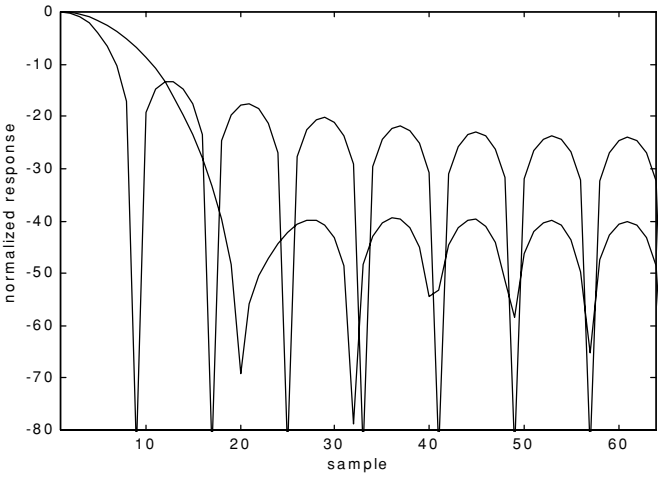

Figure 7.4. Reducing the first sidelobe to -42 dB doubles the main lobe width.

The radar echo signal is similar to the transmitted one with the exception of a time delay and an amplitude change that correspond to the target RCS. Consider a target at range R_1. The echo received by the radar from this target is

$$s_r(t) = a_1 \exp\left(j2\pi\left(f_0(t-\tau_1) + \frac{\mu}{2}(t-\tau_1)^2\right)\right) \tag{7.14}$$

where a_1 is proportional to target RCS, antenna gain, and range attenuation. The time delay τ_1 is given by

$$\tau_1 = 2R_1/c \tag{7.15}$$

The first step of the processing consists of removing the frequency f_0. This is accomplished by mixing $s_r(t)$ with a reference signal whose phase is $2\pi f_0 t$. The phase of the resultant signal, after low pass filtering, is then given by

$$\psi(t) = 2\pi\left(-f_0\tau_i + \frac{\mu}{2}(t-\tau_i)^2\right) \tag{7.16}$$

and the instantaneous frequency is

$$f_i(t) = \frac{1}{2\pi}\frac{d}{dt}\psi(t) = \mu(t-\tau_i) = \frac{B}{\tau'}\left(t - \frac{2R_1}{c}\right) \tag{7.17}$$

The quadrature components are

$$\begin{pmatrix} x_I(t) \\ x_Q(t) \end{pmatrix} = \begin{pmatrix} \cos\psi(t) \\ \sin\psi(t) \end{pmatrix} \tag{7.18}$$

Sampling the quadrature components is performed next. The number of samples, N, must be chosen so that foldover (ambiguity) in the spectrum is avoided. For this purpose, the sampling frequency, f_s (based on the Nyquist sampling rate), must be

$$f_s \geq 2B \tag{7.19}$$

and the sampling interval is

$$\Delta t \leq 1/2B \tag{7.20}$$

Using Eq. (7.17) it can be shown that (the proof is left as an exercise) the frequency resolution of the FFT is

$$\Delta f = 1/\tau' \tag{7.21}$$

The minimum required number of samples is

$$N = \frac{1}{\Delta f \Delta t} = \frac{\tau'}{\Delta t} \tag{7.22}$$

Equating Eqs. (7.20) and (7.22) yields

$$N \geq 2B\tau' \tag{7.23}$$

Consequently, a total of $2B\tau'$ real samples, or $B\tau'$ complex samples, is sufficient to completely describe an LFM waveform of duration τ' and bandwidth B. For example, an LFM signal of duration $\tau = 20\ \mu s$ and bandwidth $B = 5\ MHz$ requires 200 real samples to determine the input signal (100 samples for the I-channel and 100 samples for the Q-channel).

For better implementation of the FFT N is extended by zero padding, to the next power of two. Thus, the total number of samples, for some positive integer m, is

$$N_{FFT} = 2^m \geq N \tag{7.24}$$

The final steps of the FCP processing include: (1) taking the FFT of the sampled sequence; (2) multiplying the frequency domain sequence of the signal with the FFT of the matched filter impulse response; and (3) performing the inverse FFT of the composite frequency domain sequence in order to generate the time domain compressed pulse (HRR profile). Of course, weighting, antenna gain, and range attenuation compensation must also be performed.

Assume that I targets at ranges R_1, R_2, and so forth are within the receive window. From superposition, the phase of the down converted signal is

$$\psi(t) = \sum_{i=1}^{I} 2\pi\left(-f_0\tau_i + \frac{\mu}{2}(t-\tau_i)^2\right) \qquad \textbf{(7.25)}$$

The times $\{\tau_i = (2R_i/c);\ i = 1, 2, \ldots, I\}$ represent the two-way time delays, where τ_1 coincides with the start of the receive window.

MATLAB Function "matched_filter.m"

The function *"matched_filter.m"* performs fast convolution processing. It is given in Listing 7.1 in Section 7.5. The syntax is as follows:

[y] = matched_filter(nscat, taup, f0, b, rmin, rrec, scat_range, scat_rcs, win)

where

Symbol	Description	Units	Status
nscat	*number of point scatterers within the received window*	*none*	*input*
rmin	*minimum range of receive window*	*Km*	*input*
rrec	*receive window size*	*m*	*input*
taup	*uncompressed pulse width*	*seconds*	*input*
f0	*chirp start frequency*	*Hz*	*input*
b	*chirp bandwidth*	*Hz*	*input*
scat_range	*vector of scatterers range*	*Km*	*input*
scat_rsc	*vector of scatterers RCS*	m^2	*input*
win	*0 = no window* *1 = Hamming* *2 = Kaiser with parameter pi* *3 = Chebychev - sidelobes at -60dB*	*none*	*input*
y	*compressed output*	*volts*	*output*

The user can access this function either by a MATLAB function call, or by executing the MATLAB program *"matched_filter_driver.m"* which utilizes MATLAB based GUI. The outputs of this function are the complex array y and plots of the uncompressed and compressed signal versus relative. This function utilizes the function "power_integer_2.m" which implements Eq. (7.24):

```
function n = power_integer_2 (x)
m = 0.;
for j = 1:30
  m = m + 1.;
  delta = x - 2.^m;
  if(delta < 0.)
    n = m;
    return
  else
  end
end
```

As an example, consider the case where

nscat	*2*	*b*	*16 MHz*
rmin	*150 Km*	*scat_range*	*rmin in Km + {0, 50} meters*
rrec	*200 m*	*scat_rsc*	*{1, 1} m^2*
taup	*0.005 ms*	*win*	*2 (Kaiser)*
f0	*14 MHz*		

Note that the compressed pulsed range resolution, without using a window, is $\Delta R = 9.3m$. Figs. 7.5 and 7.6, respectively, show the uncompressed and compressed echo signal corresponding to this example.

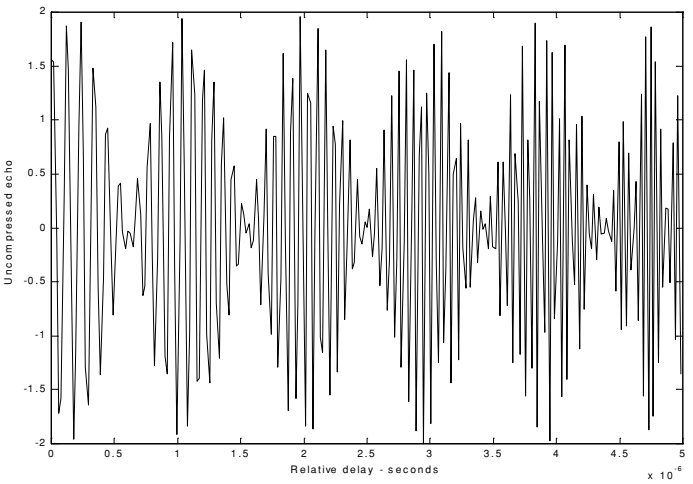

Figure 7.5. Uncompressed echo signal. Scatterers are unresolved.

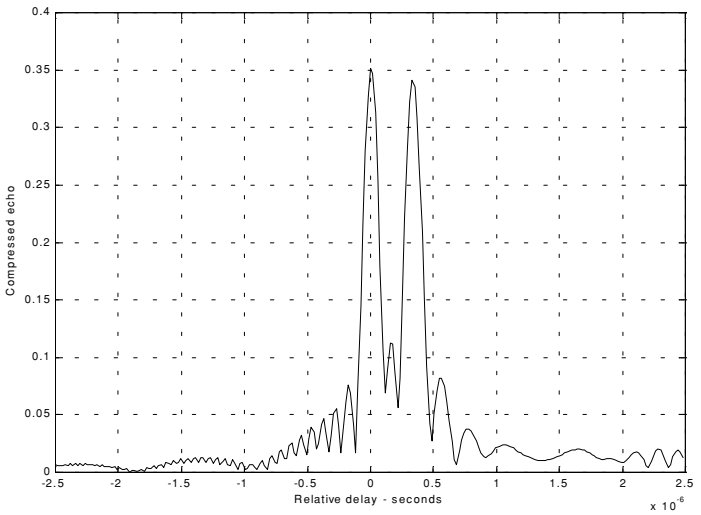

Figure 7.6. Compressed echo signal. Scatterers are resolved.

7.3.2. Stretch Processor

Stretch processing, also known as "active correlation," is normally used to process extremely high bandwidth LFM waveforms. This processing technique consists of the following steps: First, the radar returns are mixed with a replica (reference signal) of the transmitted waveform. This is followed by Low Pass Filtering (LPF) and coherent detection. Next, Analog to Digital (A/D) conversion is performed; and finally, a bank of Narrow Band Filters (NBFs) is used in order to extract the tones that are proportional to target range, since stretch processing effectively converts time delay into frequency. All returns from the same range bin produce the same constant frequency. Fig. 7.7 shows a block diagram for a stretch processing receiver. The reference signal is an LFM waveform that has the same LFM slope as the transmitted LFM signal. It exists over the duration of the radar "receive-window," which is computed from the difference between the radar maximum and minimum range. Denote the start frequency of the reference chirp as f_r.

Consider the case when the radar receives returns from a few close (in time or range) targets, as illustrated in Fig. 7.7. Mixing with the reference signal and performing low pass filtering are effectively equivalent to subtracting the return frequency chirp from the reference signal. Thus, the LPF output consists of constant tones corresponding to the targets' positions. The normalized transmitted signal can be expressed by

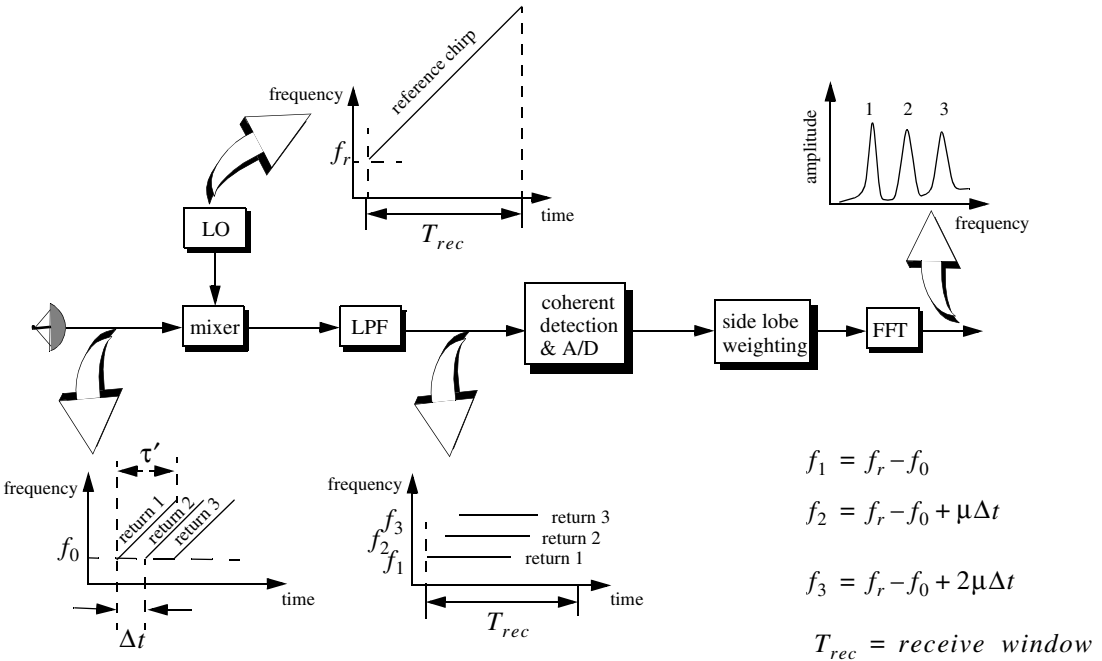

Figure 7.7. Stretch processing block diagram.

$$s_1(t) = \cos\left(2\pi\left(f_0 t + \frac{\mu}{2}t^2\right)\right) \qquad 0 \le t \le \tau' \tag{7.26}$$

where $\mu = B/\tau'$ is the LFM coefficient and f_0 is the chirp start frequency. Assume a point scatterer at range R. The received signal by the radar is

$$s_r(t) = a\cos\left[2\pi\left(f_0(t-\Delta\tau) + \frac{\mu}{2}(t-\Delta\tau)^2\right)\right] \tag{7.27}$$

where a is proportional to target RCS, antenna gain, and range attenuation. The time delay $\Delta\tau$ is

$$\Delta\tau = 2R/c \tag{7.28}$$

The reference signal is

$$s_{ref}(t) = 2\cos\left(2\pi\left(f_r t + \frac{\mu}{2}t^2\right)\right) \qquad 0 \le t \le T_{rec} \tag{7.29}$$

The received window in seconds is

$$T_{rec} = \frac{2(R_{max} - R_{min})}{c} = \frac{2R_{rec}}{c} \tag{7.30}$$

It is customary to let $f_r = f_0$. The output of the mixer is made of the product of the received and reference signals. After low pass filtering the signal is

$$s_0(t) = a\cos(2\pi f_0 t\Delta\tau + 2\pi\mu\Delta\tau t - \pi\mu(\Delta\tau)^2) \tag{7.31}$$

Substituting Eq. (7.28) into (7.31) and collecting terms yield

$$s_0(t) = a\ \cos\left[\left(\frac{4\pi BR}{c\tau'}\right)t + \frac{2R}{c}\left(2\pi f_0 - \frac{2\pi BR}{c\tau'}\right)\right] \tag{7.32}$$

and since $\tau' \gg 2R/c$, Eq. (7.32) is approximated by

$$s_0(t) \approx a\ \cos\left[\left(\frac{4\pi BR}{c\tau'}\right)t + \frac{4\pi R}{c}f_0\right] \tag{7.33}$$

The instantaneous frequency is

$$f_{inst} = \frac{1}{2\pi}\frac{d}{dt}\left(\frac{4\pi BR}{c\tau'}t + \frac{4\pi R}{c}f_0\right) = \frac{2BR}{c\tau'} \tag{7.34}$$

which clearly indicates that target range is proportional to the instantaneous frequency. Therefore, proper sampling of the LPF output and taking the FFT of

the sampled sequence lead to the following conclusion: a peak at some frequency f_1 indicates presence of a target at range

$$R_1 = f_1 c\tau'/2B \tag{7.35}$$

Assume I close targets at ranges R_1, R_2, and so forth ($R_1 < R_2 < \ldots < R_I$). From superposition, the total signal is

$$s_r(t) = \sum_{i=1}^{I} a_i(t)\cos\left[2\pi\left(f_0(t-\tau_i) + \frac{\mu}{2}(t-\tau_i)^2\right)\right] \tag{7.36}$$

where $\{a_i(t);\ i = 1, 2, \ldots, I\}$ are proportional to the targets' cross sections, antenna gain, and range. The times $\{\tau_i = (2R_i/c);\ i = 1, 2, \ldots, I\}$ represent the two-way time delays, where τ_1 coincides with the start of the receive window. Using Eq. (7.32) the overall signal at the output of the LPF can then be described by

$$s_o(t) = \sum_{i=1}^{I} a_i\cos\left[\left(\frac{4\pi BR_i}{c\tau'}\right)t + \frac{2R_i}{c}\left(2\pi f_0 - \frac{2\pi BR_i}{c\tau'}\right)\right] \tag{7.37}$$

And hence, target returns appear at constant frequency tones that can be resolved using the FFT. Consequently, determining the proper sampling rate and FFT size is very critical. The rest of this section presents a methodology for computing the proper FFT parameters required for stretch processing.

Assume a radar system using a stretch processor receiver. The pulse width is τ' and the chirp bandwidth is B. Since stretch processing is normally used in extreme bandwidth cases (i.e., very large B), the receive window over which radar returns will be processed is typically limited to few meters to possibly less than 100 meters. The compressed pulse range resolution is computed from Eq. (7.8). Declare the FFT size by N and its frequency resolution by Δf. The frequency resolution can be computed using the following procedure: consider two adjacent point scatterers at range R_1 and R_2. The minimum frequency separation, Δf, between those scatterers so that they are resolved can be computed from Eq. (7.34). More precisely,

$$\Delta f = f_2 - f_1 = \frac{2B}{c\tau'}(R_2 - R_1) = \frac{2B}{c\tau'}\Delta R \tag{7.38}$$

Substituting Eq. (7.8) into Eq. (7.38) yields

$$\Delta f = \frac{2B}{c\tau'}\ \frac{c}{2B} = \frac{1}{\tau'} \tag{7.39}$$

The maximum resolvable frequency by the FFT is limited to the region $\pm N\Delta f/2$. Thus, the maximum resolvable frequency is

$$\frac{N\Delta f}{2} > \frac{2B(R_{max} - R_{min})}{c\tau'} = \frac{2BR_{rec}}{c\tau'} \tag{7.40}$$

Using Eqs. (7.30) and (7.39) into Eq. (7.40) and collecting terms yield

$$N > 2BT_{rec} \tag{7.41}$$

For better implementation of the FFT, choose an FFT of size

$$N_{FFT} \geq N = 2^m \tag{7.42}$$

m is a nonzero positive integer. The sampling interval is then given by

$$\Delta f = \frac{1}{T_s N_{FFT}} \Rightarrow T_s = \frac{1}{\Delta f N_{FFT}} \tag{7.43}$$

MATLAB Function "stretch.m"

The function *"stretch.m"* presents a digital implementation of stretch processing. It is given in Listing 7.2 in Section 7.5. The syntax is as follows:

[y] = stretch (nscat, taup, f0, b, rmin, rrec, scat_range, scat_rcs, win)

where

Symbol	Description	Units	Status
nscat	*number of point scatterers within the received window*	*none*	*input*
rmin	*minimum range of receive window*	*Km*	*input*
rrec	*range receive window*	*m*	*input*
taup	*uncompressed pulse width*	*seconds*	*input*
f0	*chirp start frequency*	*Hz*	*input*
b	*chirp bandwidth*	*Hz*	*input*
scat_range	*vector of scatterers range*	*Km*	*input*
scat_rsc	*vector of scatterers RCS*	m^2	*input*
win	*0 = no window* *1 = Hamming* *2 = Kaiser with parameter pi* *3 = Chebychev - sidelobes at -60dB*	*none*	*input*
y	*compressed output*	*volts*	*output*

The user can access this function either by a MATLAB function call or by executing the MATLAB program *"stretch_driver.m"* which utilizes MATLAB based GUI. The outputs of this function are the complex array y and plots of the uncompressed and compressed echo signal versus time. As an example, consider the case where

nscat	*3*
rmin	*150 Km*
rrec	*30 m*
taup	*10 ms*
f0	*5.6 GHz*
b	*1 GHz*
scat_range	*rmin in Km+ {1.5, 7.5, 15.5} m*
scat_rsc	*{1, 1, 2} m^2*
win	*2 (Kaiser)*

Note that the compressed pulse range resolution, without using a window, is $\Delta R = 0.15cm$. Figs. 7.8 and 7.9, respectively, show the uncompressed and compressed echo signals corresponding to this example.

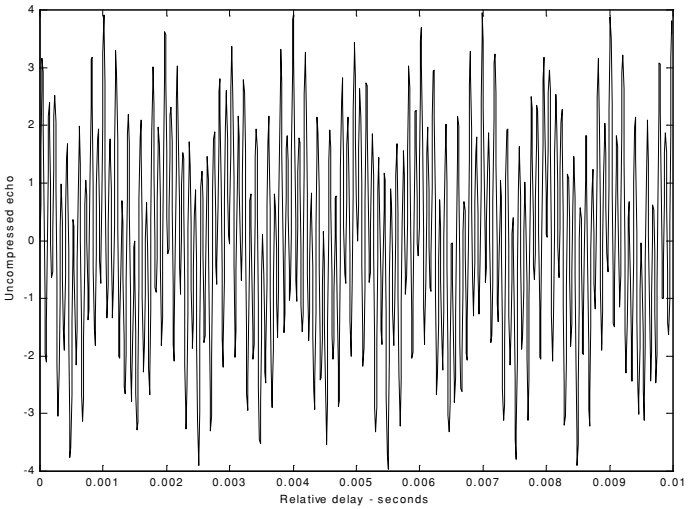

Figure 7.8. Uncompressed echo signal. Three targets are unresolved.

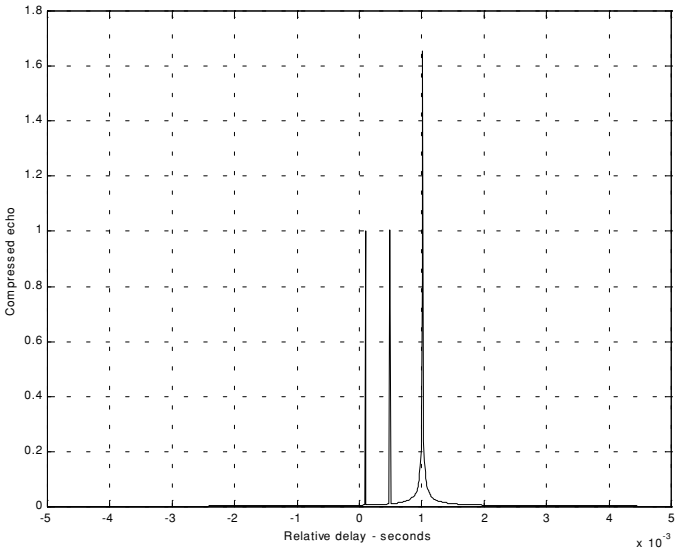

Figure 7.9. Compressed echo signal. Three targets are resolved.

7.3.3. Distortion Due to Target Velocity

Up to this point, we have analyzed pulse compression with no regards to target velocity. In fact, all analyses provided assumed stationary targets. Uncompensated target radial velocity, or equivalently Doppler shift, degrades the quality of the HRR profile generated by pulse compression. In Chapter 5, the effects of radial velocity on SFW were analyzed; similar distortion in the HRR profile is also present with LFM waveforms when target radial velocity is not compensated for.

The two effects of target radial velocity (Doppler frequency) on the radar received pulse were developed in Chapter 1. When the target radial velocity is not zero, the received pulse width is expanded (or compressed) by the time dilation factor. Additionally, the received pulse center frequency is shifted by the amount of Doppler frequency. When these effects are not compensated for, the pulse compression processor output is distorted. This is illustrated in Fig. 7.10. Fig. 7.10a shows a typical output of the pulse compression processor with no distortion. Alternatively, Figs. 7.10b, 7.10c, and 7.10d show the output of the pulse compression processor when 5% shift of the chirp center frequency and 10% time dilation are present.

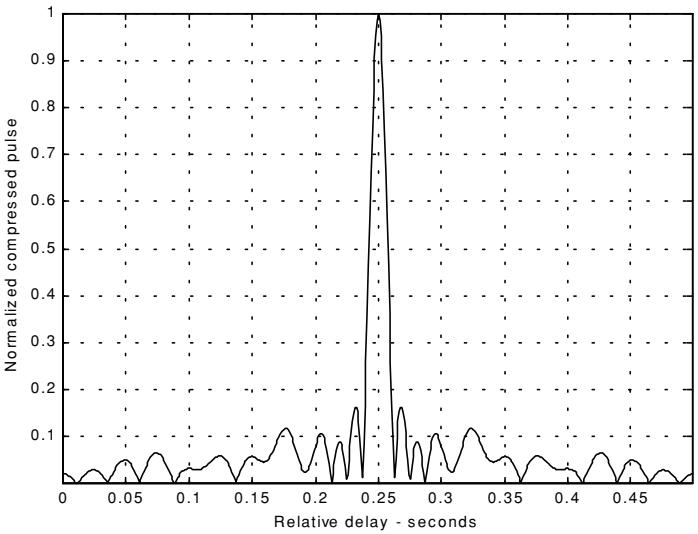

Figure 7.10a. Compressed pulse output of a pulse compression processor. No distortion is present. This figure can be reproduced using MATLAB program *"fig7_10"* given in Listing 7.3 in Section 7.5.

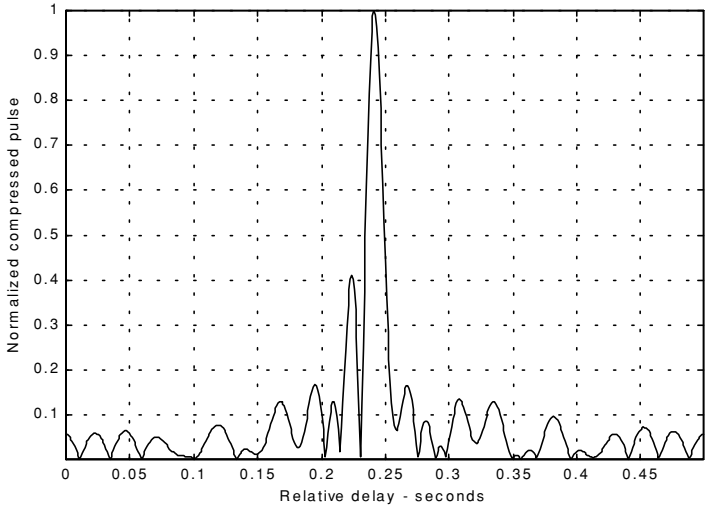

Figure 7.10b. Mismatched compressed pulse; 5% Doppler shift.

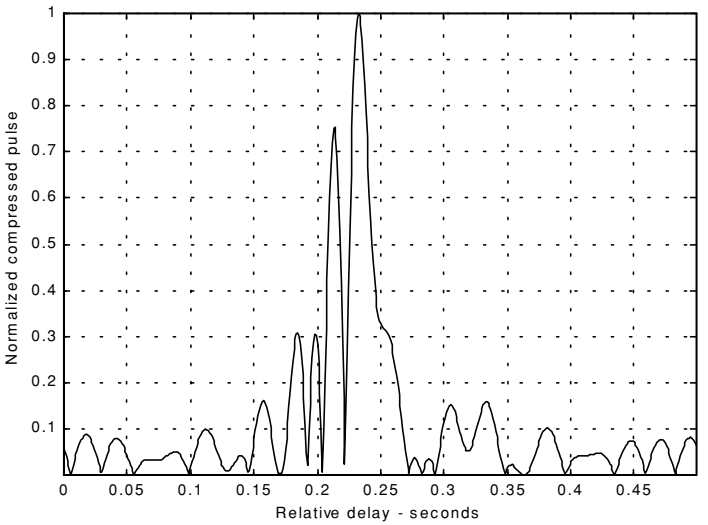

Figure 7.10c. Mismatched compressed pulse; 10% time dilation.

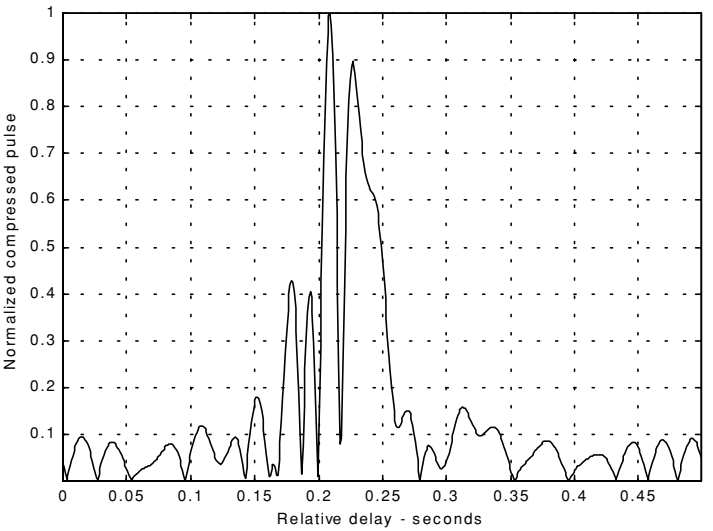

Figure 7.10d. Mismatched compressed pulse; 10% time dilation and 5% Doppler shift.

Correction for the distortion caused by the target radial velocity can be overcome by using the following approach. Over a period of few pulses, the radar data processor estimates the radial velocity of the target under track. Then, the chirp slope and pulse width of the next transmitted pulse are changed to account for the estimated Doppler frequency and time dilation.

7.3.4. Range Doppler Coupling

Plots and characteristics of the ambiguity function for an LFM waveform were presented in Chapter 6. However, the distinctive property of range Doppler coupling associated with LFM was not presented. Range Doppler coupling is a phrase used to describe the shift in the delay/range response of an LFM ambiguity function due to the presence of a Doppler shift. The nature of range Doppler coupling can be better understood by analyzing the LFM ambiguity function. An expression for an LFM ambiguity function was developed in Chapter 6, and is repeated here as Eq. (7.44):

$$\left|\chi(\tau;f_d)\right|^2 = \left|\left(1-\frac{|\tau|}{\tau'}\right)\frac{\sin\left(\pi\tau'(\mu\tau+f_d)\left(1-\frac{|\tau|}{\tau'}\right)\right)}{\pi\tau'(\mu\tau+f_d)\left(1-\frac{|\tau|}{\tau'}\right)}\right|^2 \qquad |\tau| \le \tau' \qquad \textbf{(7.44)}$$

For this purpose, consider the sketch of an LFM ambiguity function shown in Fig. 7.11.

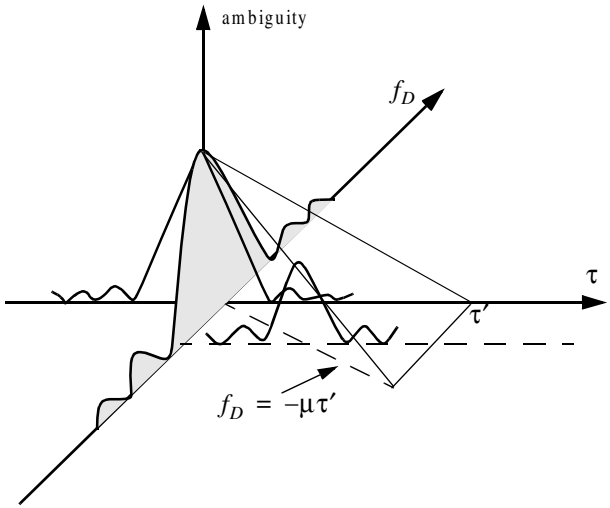

Figure 7.11. Illustration of range Doppler coupling for an LFM pulse.

The ambiguity surface extends from $-\tau'$ to τ' in range and from $-\infty$ to ∞ in Doppler. The response has a maximum at the point $(\tau, f_D) = (0, 0)$. Profiles parallel to the Doppler axis have maxima above the line $f_D = -\mu\tau$ which passes through the origin. The presence of radial velocity forces the peak of the ambiguity surface to a point that has a peak value smaller than the maximum that occurs at the origin. However, as long as the shift is less than the line $f_D = 1/\tau'$, the ambiguity function response exerts acceptable reduction in peak values, as illustrated in Fig. 7.11. This is the reason why some times LFM waveforms are called Doppler invariant.

7.4. Digital Pulse Compression

In this section we will briefly discuss three digital pulse compression techniques. They are frequency codes, binary phase codes, and poly-phase codes. Costas codes, Barker Codes, and Frank codes will be presented to illustrate, respectively, frequency, binary phase, and poly-phase coding. We will determine the pulse compression goodness of a code, based on its autocorrelation function since in the absence of noise, the output of the matched filter is proportional to the code autocorrelation. Given the autocorrelation function of a certain code, the main lobe width (compressed pulse width) and the side lobe levels are the two factors that need to be considered in order to evaluate the code's pulse compression characteristics.

7.4.1. Frequency Coding (Costas Codes)

Construction of Costas codes can be understood from the construction process of Stepped Frequency Waveforms (SFW) described in Chapter 5. In SFW, a relatively long pulse of length τ' is divided into N subpulses, each of width τ_1 ($\tau' = N\tau_1$). Each group of N subpulses is called a burst. Within each burst the frequency is increased by Δf from one subpulse to the next. The overall burst bandwidth is $N\Delta f$. More precisely,

$$\tau_1 = \tau'/N \tag{7.45}$$

and the frequency for the ith subpulse is

$$f_i = f_0 + i\Delta f \qquad ; \; i = 1, N \tag{7.46}$$

where f_0 is a constant frequency and $f_0 \gg \Delta f$. It follows that the time-bandwidth product of this waveform is

$$\Delta f\tau' = N^2 \tag{7.47}$$

Costas signals (or codes) are similar to SFW, except that the frequencies for the subpulses are selected in a random fashion, according to some predetermined rule or logic. For this purpose, consider the $N \times N$ matrix shown in Fig. 7.12. In this case, the rows are indexed from $i = 1, 2, \ldots, N$ and the columns are indexed from $j = 0, 1, 2, \ldots, (N-1)$. The rows are used to denote the subpulses and the columns are used to denote the frequency. A *"dot"* indicates the frequency value assigned to the associated subpulse. In this fashion, Fig. 7.12a shows the frequency assignment associated with a SFW. Alternatively, the frequency assignments in Fig. 7.12b are chosen randomly. For a matrix of size $N \times N$, there are a total of $N!$ possible ways of assigning the *"dots"* (i.e., $N!$ possible codes).

The sequences of *"dots"* assignment for which the corresponding ambiguity function approaches an ideal or a *"thumbtack"* response are called Costas codes. A near thumbtack response was obtained by Costas[1] by using the following logic: only one frequency per time slot (row) and per frequency slot (column). Therefore, for an $N \times N$ matrix the number of possible Costas codes is drastically less than $N!$. For example, there are $N_c = 4$ possible Costas codes for $N = 3$, and $N_c = 40$ possible codes for $N = 5$. It can be shown that the code density, defined as the ratio $N_c/N!$, significantly gets smaller as N becomes larger.

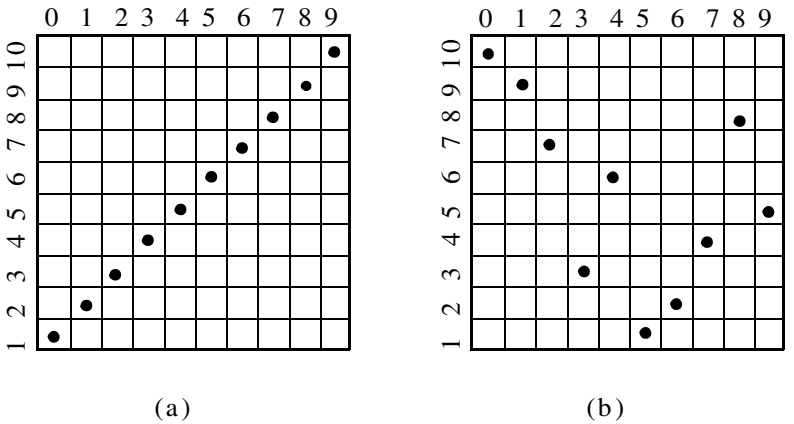

Figure 7.12. Frequency assignment for a burst of N subpulses. (a) SFW (stepped LFM); (b) Costas code of length Nc = 10.

1. Costas, J. P., A study of a Class of Detection Waveforms Having Nearly Ideal Range-Doppler Ambiguity Properties, *Proc. IEEE* ***72***, 1984, pp. 996-1009.

There are numerous analytical ways to generate Costas codes. In this section we will describe two of these methods. First, let q be an odd prime number, and choose the number of subpulses as

$$N = q - 1 \tag{7.48}$$

Define γ as the primitive root of q. A primitive root of q (an odd prime number) is defined as γ such that the powers $\gamma, \gamma^2, \gamma^3, \ldots, \gamma^{q-1}$ modulo q generate every integer from 1 to $q - 1$.

In the first method, for an $N \times N$ matrix, label the rows and columns, respectively, as

$$\begin{aligned} i &= 0, 1, 2, \ldots, (q-2) \\ j &= 1, 2, 3, \ldots, (q-1) \end{aligned} \tag{7.49}$$

Place a dot in the location (i, j) corresponding to the frequency f_i (from Eq. (7.46)) if and only if

$$i = (\gamma)^j \ \ (modulo \ \ q) \tag{7.50}$$

In the next method, Costas code is first obtained from the logic described above; then by deleting the first row and first column from the matrix a new code is generated. This method produces a Costas code of length $N = q - 2$.

Define the normalized complex envelope of the Costas signal as

$$s(t) = \frac{1}{\sqrt{N\tau_1}} \sum_{l=0}^{N-1} s_l(t - l\tau_1) \tag{7.51}$$

$$s_l(t) = \begin{pmatrix} \exp(j2\pi f_l t) & 0 \le t \le \tau_1 \\ 0 & elsewhere \end{pmatrix} \tag{7.52}$$

Costas showed that the output of the matched filter is

$$\chi(\tau, f_D) = \frac{1}{N} \sum_{l=0}^{N-1} \exp(j2\pi l f_D \tau) \left\{ \Phi_{ll}(\tau, f_D) + \sum_{\substack{q=0 \\ q \ne l}}^{N-1} \Phi_{lq}(\tau - (l-q)\tau_1, f_D) \right\} \tag{7.53}$$

$$\Phi_{lq}(\tau, f_D) = \left(\tau_1 - \frac{|\tau|}{\tau_1}\right) \frac{\sin\alpha}{\alpha} \ \exp(-j\beta - j2\pi f_q \tau) \qquad , \ |\tau| \le \tau_1 \tag{7.54}$$

$$\alpha = \pi(f_l - f_q - f_D)(\tau_1 - |\tau|) \tag{7.55}$$

$$\beta = \pi(f_l - f_q - f_D)(\tau_1 + |\tau|) \tag{7.56}$$

Three-dimensional plots for the ambiguity function of Costas signals show the near thumbtack response of the ambiguity function. All sidelobes, except for few around the origin, have amplitude $1/N$. Few sidelobes close to the origin have amplitude $2/N$, which is typical of Costas codes. The compression ratio of a Costas code is approximately N.

7.4.2. Binary Phase Codes

In this case, a relatively long pulse of width τ' is divided into N smaller pulses; each is of width $\Delta\tau = \tau'/N$. Then, the phase of each sub-pulse is randomly chosen as either 0 or π radians relative to some CW reference signal. It is customary to characterize a sub-pulse that has 0 phase (amplitude of +1 Volt) as either "1" or "+." Alternatively, a sub-pulse with phase equal to π (amplitude of -1 Volt) is characterized by either "0" or "-." The compression ratio associated with binary phase codes is equal to $\xi = \tau'/\Delta\tau$, and the peak value is N times larger than that of the long pulse. The goodness of a compressed binary phase code waveform depends heavily on the random sequence of the phase for the individual sub-pulses.

One family of binary phase codes that produce compressed waveforms with constant side lobe levels equal to unity is the Barker code. Fig. 7.13 illustrates this concept for a Barker code of length seven. A Barker code of length n is denoted as B_n. There are only seven known Barker codes that share this unique property; they are listed in Table 7.1. Note that B_2 and B_4 have complementary forms that have the same characteristics. Since there are only seven Barker codes, they are not used when radar security is an issue.

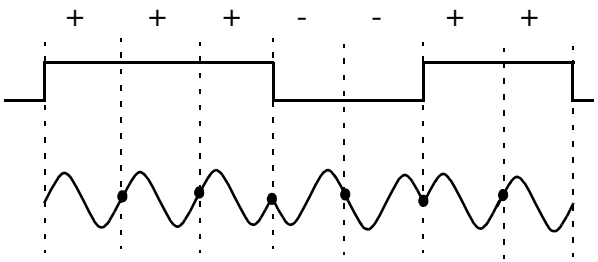

Figure 7.13. Binary phase code of length 7.

TABLE 7.1. Barker codes.

Code symbol	Code length	Code elements	Side lode reduction (dB)
B_2	2	+- ++	6.0
B_3	3	++-	9.5
B_4	4	++-+ +++-	12.0
B_5	5	+++-+	14.0
B_7	7	+++--+-	16.9
B_{11}	11	+++---+--+-	20.8
B_{13}	13	+++++--++-+-+	22.3

In general, the autocorrelation function (which is an approximation for the matched filter output) for a B_N Barker code will be $2N\Delta\tau$ wide. The main lobe is $2\Delta\tau$ wide; the peak value is equal to N. There are $(N-1)/2$ side lobes on either side of the main lobe; this is illustrated in Fig. 7.14 for a B_{13}. Notice that the main lobe is equal to 13, while all side lobes are unity.

The most side lobe reduction offered by a Barker code is $-22.3dB$, which may not be sufficient for the desired radar application. However, Barker codes can be combined to generate much longer codes. In this case, a B_m code can be used within a B_n code (m within n) to generate a code of length mn. The compression ratio for the combined B_{mn} code is equal to mn. As an example, a combined B_{54} is given by

$$B_{54} = \{11101, 11101, 00010, 11101\} \tag{7.57}$$

and is illustrated in Fig. 7.15. Unfortunately, the side lobes of a combined Barker code autocorrelation function are no longer equal to unity.

Some side lobes of a Barker code autocorrelation function can be reduced to zero if the matched filter is followed by a linear transversal filter with impulse response given by

$$h(t) = \sum_{k=-N}^{N} \beta_k \delta(t - 2k\Delta\tau) \tag{7.58}$$

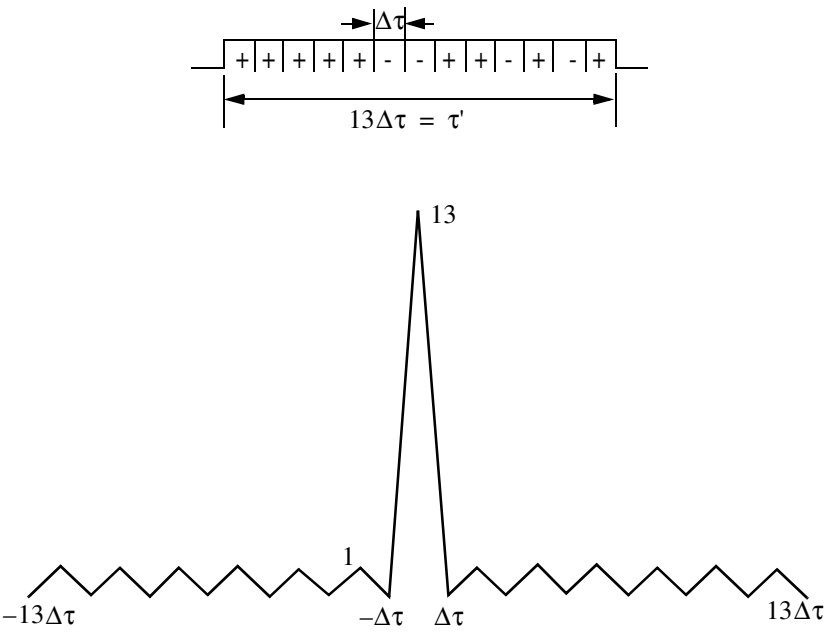

Figure 7.14. Barker code of length 13, and its corresponding autocorrelation function.

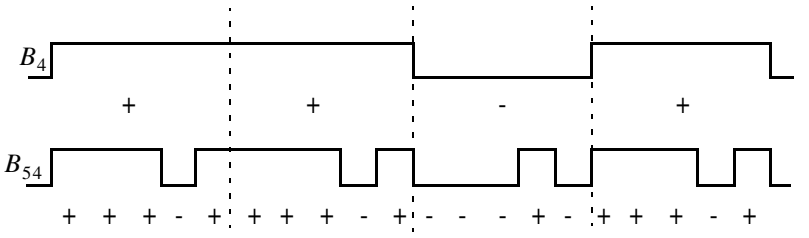

Figure 7.15. A combined B_{54} Barker code.

where N is the filter's order, the coefficients β_k ($\beta_k = \beta_{-k}$) are to be determined, $\delta(\cdot)$ is the delta function, and $\Delta\tau$ is the Barker code sub-pulse width. A filter of order N produces N zero side lobes on either side of the main lobe. The main lobe amplitude and width do not change. This is illustrated in Fig. 7.16.

In order to illustrate this approach further, consider the case where the input to the matched filter is B_{11}, and assume $N = 4$. The autocorrelation for a B_{11} code is

$$\phi_{11} = \{-1, 0, -1, 0, -1, 0, -1, 0, -1, 0, 11, \\ 0, -1, 0, -1, 0, -1, 0, -1, 0, -1\} \tag{7.59}$$

The output of the transversal filter is the discrete convolution between its impulse response and the sequence ϕ_{11}. At this point we need to compute the coefficients β_k that guarantee the desired filter output (i.e., unchanged main lobe and four zero side lobe levels). Performing the discrete convolution as defined in Eq. (7.58), and collecting equal terms ($\beta_k = \beta_{-k}$) yield the following set of five linearly independent equations:

$$\begin{bmatrix} 11 & -2 & -2 & -2 & -2 \\ -1 & 10 & -2 & -2 & -1 \\ -1 & -2 & 10 & -2 & -1 \\ -1 & -2 & -1 & 11 & -1 \\ -1 & -1 & -1 & -1 & 11 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{bmatrix} = \begin{bmatrix} 11 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{7.60}$$

The solution of Eq. (7.60) is left as an exercise. Note that by setting the first equation equal to 11 and all other equations to 0 and then solving for β_k guarantees that the main peak remains unchanged, and that the next four side lobes are zeros. So far we have assumed that coded pulses have rectangular shapes. Using other pulses of other shapes, such as Gaussian, may produce better side lobe reduction and a larger compression ratio.

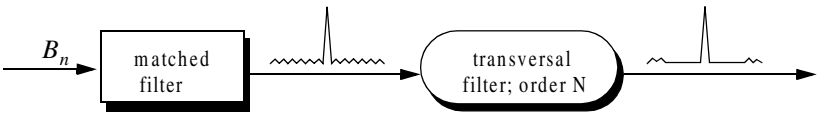

Figure 7.16. A linear transversal filter of order N can be used to produce N zero side lobes in the autocorrelation function. In this figure, N = 4.

7.4.3. *Frank Codes*

Codes that use any harmonically related phases based on a certain fundamental phase increment are called poly-phase codes. We will demonstrate this coding technique using Frank codes. In this case, a single pulse of width τ' is divided into N equal groups; each group is subsequently divided into other N sub-pulses each of width $\Delta\tau$. Therefore, the total number of sub-pulses within each pulse is N^2, and the compression ratio is $\xi = N^2$. As before, the phase within each sub-pulse is held constant with respect to some CW reference signal.

A Frank code of N^2 sub-pulses is referred to as an N-phase Frank code. The first step in computing a Frank code is to divide $360°$ by N, and define the result as the fundamental phase increment $\Delta\varphi$. More precisely,

$$\Delta\varphi = \frac{360°}{N} \tag{7.61}$$

Note that the size of the fundamental phase increment decreases as the number of groups is increased, and because of phase stability, this may degrade the performance of very long Frank codes. For N-phase Frank code the phase of each sub-pulse is computed from

$$\begin{pmatrix} 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 2 & 3 & \dots & N-1 \\ 0 & 2 & 4 & 6 & \dots & 2(N-1) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & (N-1) & 2(N-1) & 3(N-1) & \dots & (N-1)^2 \end{pmatrix} \Delta\varphi \tag{7.62}$$

where each row represents a group, and a column represents the sub-pulses for that group. For example, a 4-phase Frank code has $N = 4$, and the fundamental phase increment is $\Delta\varphi = (360°/4) = 90°$. It follows that

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 90° & 180° & 270° \\ 0 & 180° & 0 & 180° \\ 0 & 270° & 180° & 90° \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{pmatrix} \tag{7.63}$$

Therefore, a Frank code of 16 elements is given by

$$F_{16} = \{1 \;\; 1 \;\; 1 \;\; 1 \;\; 1 \;\; j \;\; -1 \;\; -j \;\; 1 \;\; -1 \;\; 1 \;\; -1 \;\; 1 \;\; -j \;\; -1 \;\; j\} \tag{7.64}$$

The phase increments within each row represent a stepwise approximation of an up-chirp LFM waveform. The phase increments for subsequent rows increase linearly versus time. Thus, the corresponding LFM chirp slopes also increase linearly for subsequent rows. This is illustrated in Fig. 7.17, for F_{16}.

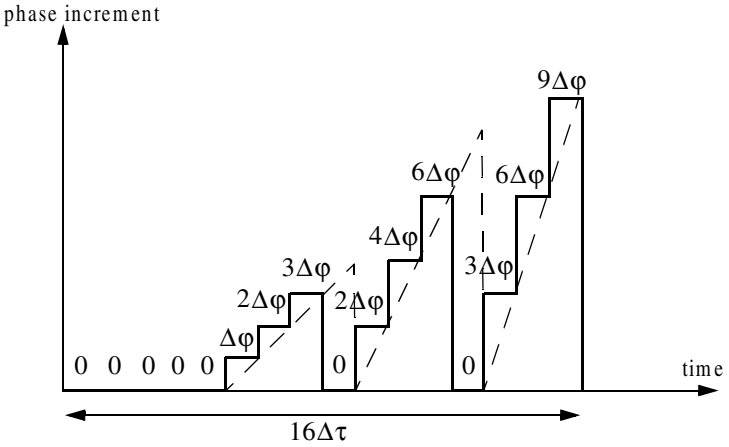

Figure 7.17. Stepwise approximation of an up-chirp waveform, using a Frank code of 16 elements.

7.4.4. Pseudo-Random (PRN) Codes

Pseudo-random (PRN) codes are also known as Maximal Length Sequences (MLS) codes. These codes are called pseudo-random because the statistics associated with their occurrence is similar to that associated with the coin-toss sequences. Maximum length sequences are periodic with period L and the code values take on two binary values (+1 and -1). The MLS correlation function is

$$\phi(n) = \begin{cases} L & n = 0, \pm L, \pm 2L, \ldots \\ -1 & elsewhere \end{cases} \qquad \textbf{(7.65)}$$

Fig. 7.18 shows a typical sketch for an MLS autocorrelation function. Clearly these codes have the advantage that the compression ratio becomes very large as the period is increased. Additionally, adjacent peaks (grating lobes) become farther apart.

Maximum length sequences exist for all integer values m, with a period equal to $2^m - 1$. They can be generated using shift register circuits with the proper feedback connections, where the sum is a modulo-2 operation. This is

illustrated in Fig. 7.19 for $m = 4$ (i.e., $L = 15$). Note that the circuit shown in Fig. 7.19 is not the only one that can produce this code.

In radar applications, long codes are very desirable. However, having very long codes presents many possibilities for the feedback connections through the modulo-2 adder. For example, for $m = 80$, the period is $L = 2^{80} - 1$, which is very huge and may take years to produce the corresponding code. Therefore, there is a need for a more systematic method for producing MLS codes.

In practice, typical MLS codes are produced by using the primitive polynomials with the proper degree that corresponds to the code, and the feedback connections are made according to the chosen polynomial, as illustrated in Fig. 7.19 for $m = 4$. In this example the primitive polynomial is $x^4 + x + 1$. Of course the initial loading for the registers must be different from all zeros. More details on primitive polynomials can be found in many sited references.

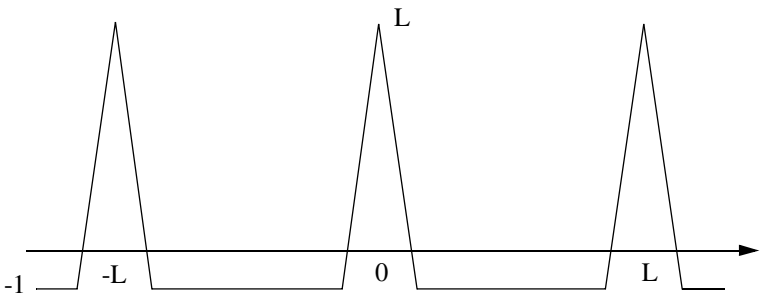

Figure 7.18. Typical autocorrelation of an MLS code of length L.

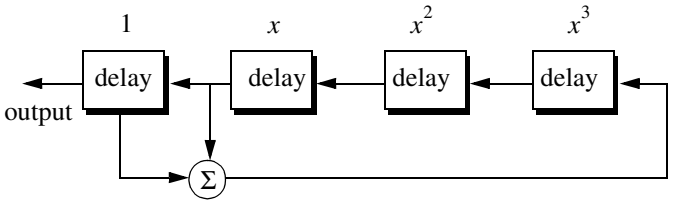

Figure 7.19. Circuit for generating an MLS sequence of length $L = 15$. The primitive polynomial is $x^4 + x + 1$.

7.5. MATLAB Listings

This section presents listings for all MATLAB programs/functions used in this chapter. The user is advised to rerun these programs with different input parameters.

Listing 7.1. MATLAB Function "matched_filter.m"

```
function [y] = matched_filter(nscat, taup, f0, b, rmin, rrec, scat_range,
scat_rcs, winid)
%
eps = 1.0e-16;
htau = taup / 2.;
c = 3.e8;
n = fix(2. * taup * b);
m = power_integer_2(n);
nfft = 2.^m;
x(nscat,1:nfft) = 0.;
y(1:nfft) = 0.;
replica(1:nfft) = 0.;
if( winid == 0.)
  win(1:nfft) = 1.;
  win =win';
else
  if(winid == 1.)
    win = hamming(nfft);
  else
    if( winid == 2.)
      win = kaiser(nfft,pi);
    else
      if(winid == 3.)
        win = chebwin(nfft,60);
      end
    end
  end
end
deltar = c / 2. / b;
max_rrec = deltar * nfft / 2.;
maxr = max(scat_range) - rmin;
if(rrec > max_rrec | maxr >= rrec )
  'Error. Receive window is too large; or scatterers fall outside window'
  break
end
trec = 2. * rrec / c;
```

```
deltat = taup / nfft;
t = 0: deltat:taup-eps;
uplimit = max(size(t));
replica(1:uplimit) = exp(i * 2.* pi * (.5 * (b/taup) .* t.^2));
figure(3)
subplot(2,1,1)
plot(real(replica))
title('Matched filter time domain response')
subplot(2,1,2)
plot(fftshift(abs(fft(replica))));
title('Matched filter frequency domain response')
for j = 1:1:nscat
   t_tgt = 2. * (scat_range(j) - rmin) / c +htau;
   x(j,1:uplimit) = scat_rcs(j) .* exp(i *  2.* pi * ...
      (.5 * (b/taup) .* (t+t_tgt).^2));
   y = y + x(j,:);
end
figure(1)
plot(t,real(y),'k')
xlabel ('Relative delay - seconds')
ylabel ('Uncompressed echo')
title ('Zero delay coincide with minimum range')
rfft = fft(replica,nfft);
yfft = fft(y,nfft);
out= abs(ifft((rfft .* conj(yfft)) .* win' )) ./ (nfft);
figure(2)
time = -htau:deltat:htau-eps;
plot(time,out,'k')
xlabel ('Relative delay - seconds')
ylabel ('Compressed echo')
title ('Zero delay coincide with minimum range')
grid
```

Listing 7.2. MATLAB Function "stretch.m"

```
function [y] = stretch(nscat,taup,f0,b,rmin,rrec,scat_range,scat_rcs,winid)
eps = 1.0e-16;
htau = taup / 2.;
c = 3.e8;
trec = 2. * rrec / c;
n = fix(2. * trec * b);
m = power_integer_2(n);
nfft = 2.^m;
x(nscat,1:nfft) = 0.;
```

```
y(1:nfft) = 0.;
if( winid == 0.)
  win(1:nfft) = 1.;
  win =win';
else
  if(winid == 1.)
    win = hamming(nfft);
  else
    if( winid == 2.)
      win = kaiser(nfft,pi);
    else
      if(winid == 3.)
        win = chebwin(nfft,60);
      end
    end
  end
end
deltar = c / 2. / b;
max_rrec = deltar * nfft / 2.;
maxr = max(scat_range) - rmin;
if(rrec > max_rrec | maxr >= rrec )
  'Error. Receive window is too large; or scatterers fall outside window'
  break
end
deltat = taup / nfft;
t = 0: deltat:taup-eps;
uplimit = max(size(t));
for j = 1:1:nscat
  psi1 = 4. * pi * scat_range(j) * f0 / c - ...
    4. * pi * b * scat_range(j) * scat_range(j) / c / c/ taup;
  psi2 = (4. * pi * b * scat_range(j) / c / taup) .* t;
  x(j,1:uplimit) = scat_rcs(j) .* exp(i * psi1 + i .* psi2);
  y = y + x(j,:);
end
figure(1)
plot(t,real(y),'k')
xlabel ('Relative delay - seconds')
ylabel ('Uncompressed echo')
title ('Zero delay coincide with minimum range')
ywin = y .* win';
yfft = fft(y,nfft) ./ nfft;
out= fftshift(abs(yfft));
figure(2)
time = -htau:deltat:htau-eps;
```

```
plot(time,out,'k')
xlabel ('Relative delay - seconds')
ylabel ('Compressed echo')
title ('Zero delay coincide with minimum range')
grid
```

Listing 7.3. MATLAB Program "fig7_10.m'

```
clear all
eps = 1.5e-5;
t = 0:0.001:.5;
y = chirp(t,0,.25,20);
figure(1)
plot(t,y);
yfft = fft(y,512) ;
ycomp = fftshift(abs(ifft(yfft .* conj(yfft))));
maxval = max (ycomp);
ycomp = eps + ycomp ./ maxval;
figure(1)
del = .5 /512.;
tt = 0:del:.5-eps;
plot (tt,ycomp,'k')
xlabel ('Relative delay - seconds');
ylabel('Normalized compressed pulse')
grid
%change center frequency
y1 = chirp (t,0,.25,21);
y1fft = fft(y1,512);
y1comp = fftshift(abs(ifft(y1fft .* conj(yfft))));
maxval = max (y1comp);
y1comp = eps + y1comp ./ maxval;
figure(2)
plot (tt,y1comp,'k')
xlabel ('Relative delay - seconds');
ylabel('Normalized compressed pulse')
grid
%change pulse width
t = 0:0.001:.45;
y2 = chirp (t,0,.225,20);
y2fft = fft(y2,512);
y2comp = fftshift(abs(ifft(y2fft .* conj(yfft))));
maxval = max (y2comp);
y2comp = eps + y2comp ./ maxval;
figure(3)
```

```
plot (tt,y2comp,'k')
xlabel ('Relative delay - seconds');
ylabel('Normalized compressed pulse')
grid
```

Problems

7.1. Starting with Eq. (7.17), prove Eq. (7.21).

7.2. The smallest positive primitive root of $q = 11$ is $\gamma = 2$; for $N = 10$ generate the corresponding Costas matrix.

7.3. Develop a MATLAB program to plot the ambiguity function associated with Costas codes. Use Eqs. (7.53) through (7.56). Your program should generate 3-D plots, contour plots, and zero delay/Doppler cuts. Verify the side lobe behaviour and the compression ratio of Costas codes.

7.4. Consider the 7-bit Barker code, designated by the sequence $x(n)$. (a) Compute and plot the autocorrelation of this code. (b) A radar uses binary phase coded pulses of the form $s(t) = r(t)\cos(2\pi f_0 t)$, where $r(t) = x(0),\ for\ \ 0 < t < \Delta t$, $r(t) = x(n),\ for\ \ n\Delta t < t < (n+1)\Delta t$, and $r(t) = 0,\ \ for\ \ t > 7\Delta t$. Assume $\Delta t = 0.5\mu s$. (a) Give an expression for the autocorrelation of the signal $s(t)$, and for the output of the matched filter when the input is $s(t - 10\Delta t)$; (b) compute the time bandwidth product, the increase in the peak SNR, and the compression ratio.

7.5. (a) Perform the discrete convolution between the sequence ϕ_{11} defined in Eq. (7.59), and the transversal filter impulse response (i.e., derive Eq. (7.60). (b) Solve Eq. (7.60), and sketch the corresponding transversal filter output.

7.6. Repeat the previous problem for $N = 13$ and $k = 6$. Use Barker code of length 13.

7.7. Develop a Barker code of length 35. Consider both B_{75} and B_{57}.

7.8. Write a computer program to calculate the discrete correlation between any two finite length sequences. Verify your code by comparing your results to the output of the MATLAB function *"xcorr"*.

7.9. Compute the discrete autocorrelation for an F_{16} Frank code.

7.10. Generate a Frank code of length 8, F_8.