Chapter 8 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, hence reducing the SNR. Since the average transmitted power is directly linked to the receiver SNR, it is often desirable to increase the pulse width (i.e., the average transmitted power) while simultaneously maintaining adequate range resolution. This can be made possible by using pulse compression techniques and the matched filter receiver. 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, two pulse compression techniques are discussed. The first technique is known as correlation processing which is predominantly 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.

8.1. 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

$$N_i = 2 \frac{\eta_0}{2} B \tag{8.1}$$

where the factor of two is used to account for both negative and positive frequency bands, as illustrated in Fig. 8.1. The average input signal power over a pulse duration τ_0 is

$$S_i = \frac{E_x}{\tau_0} \tag{8.2}$$

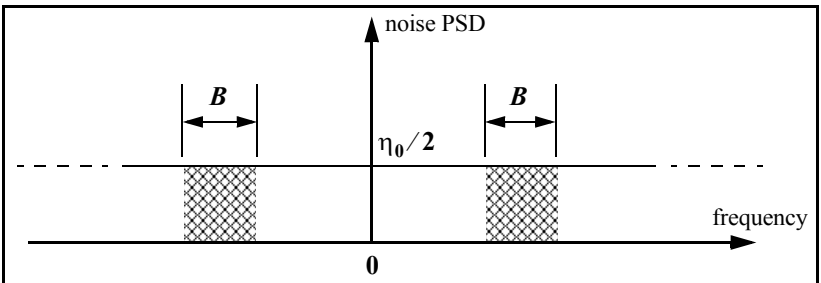

Figure 8.1. Input noise power.

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

$$(SNR)_i = \frac{S_i}{N_i} = \frac{E}{\eta_0 B\tau_0} \tag{8.3}$$

The output peak instantaneous SNR to the input SNR ratio is

$$\frac{SNR(t_0)}{(SNR)_i} = 2B\tau_0 \tag{8.4}$$

The quantity $B\tau_0$ is referred to as the time-bandwidth product for a given waveform or its corresponding matched filter. The factor $B\tau_0$ 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 $B\tau_0$. Clearly, the compression gain becomes smaller than $B\tau_0$ as the spectrum of the matched filter deviates from that of the input signal.

8.2. Radar Equation with Pulse Compression

The radar equation for a pulsed radar can be written as

$$SNR = \frac{P_t \tau_0 G^2 \lambda^2 \sigma}{(4\pi)^3 R^4 k T_0 F L} \tag{8.5}$$

where P_t is peak power, τ_0 is pulse width, G is antenna gain, σ is target RCS, R is range, k is Boltzmann's constant, T_0 is 290 degrees Kelvin, 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 as being 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. (8.5) can be written as

$$(SNR)_{\tau_c} = \frac{P_t \tau_c G^2 \lambda^2 \sigma}{(4\pi)^3 R^4 k T_0 F L} \tag{8.6}$$

The SNR for the uncompressed pulse is then derived from Eq. (8.6) as

$$SNR = \frac{P_t(\tau_0 = n_P \tau_c) G^2 \lambda^2 \sigma}{(4\pi)^3 R^4 k T_0 F L} \tag{8.7}$$

where n_P is the number of subpulses. Equation (8.7) is denoted as the radar equation with pulse compression.

Observation of Eq. (8.5) and Eq.(8.7) 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, 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:

$$\Delta R = \frac{c}{2B} \tag{8.8}$$

8.3. Basic Principal of Pulse Compression

For this purpose, consider a long pulse with LFM modulation and assume a matched filter receiver. The output of the matched filter (along the delay axis, i.e., range) is an order of magnitude narrower than that at its input. More precisely, the matched filter output is compressed by a factor $\xi = B\tau_0$, where τ_0 is the pulse width and B is the bandwidth. Thus, by using long pulses and wideband LFM modulation, large compression ratios can be achieved.

Figure 8.2 shows an ideal LFM pulse compression process. Part (a) shows the envelope of a pulse, part (b) shows the frequency modulation (in this case it is an upchirp LFM) with bandwidth $B = f_2 - f_1$. Part (c) shows the matched filter time-delay characteristic while part (d) shows the compressed pulse envelope. Finally part (e) shows the matched filter input/output waveforms.

Figure 8.3 illustrates the advantage of pulse compression using a more realistic LFM waveform. In this example, 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. Figure 8.3a 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 distinct targets. In fact, when using LFM, returns from neighboring targets are resolved as long as they are separated in time by τ_c, the compressed pulse width.

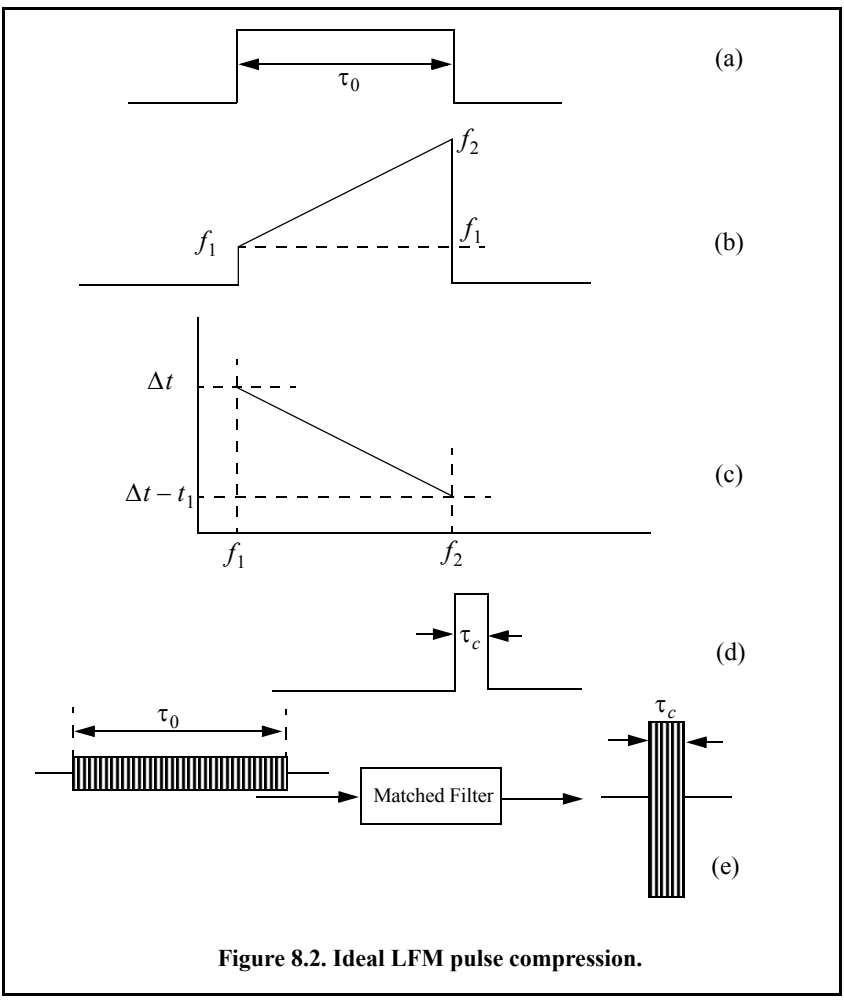

Figure 8.2. Ideal LFM pulse compression.

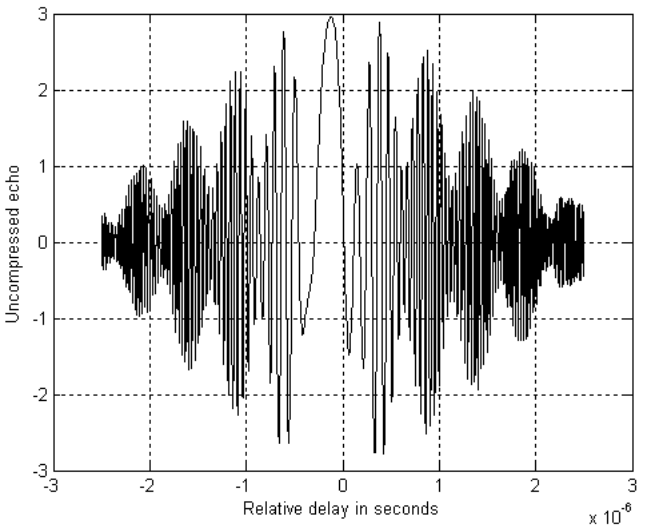

Figure 8.3a. Composite echo signal for two unresolved targets.

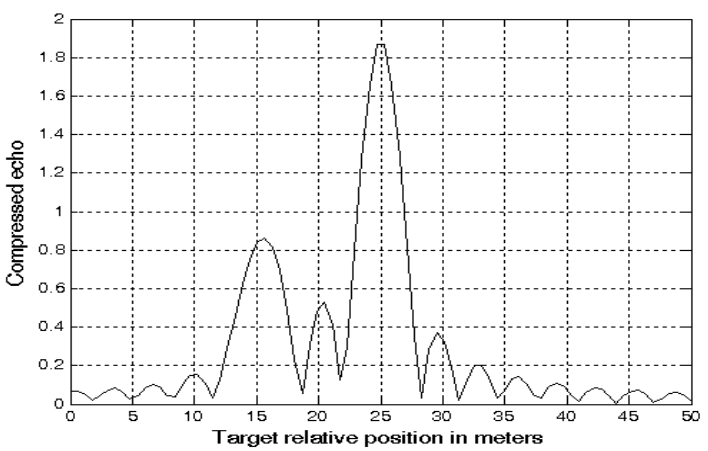

Figure 8.3b. Composite echo signal corresponding to Fig. 8.3a after pulse compression.

8.4. Correlation Processor

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 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. 8.4.

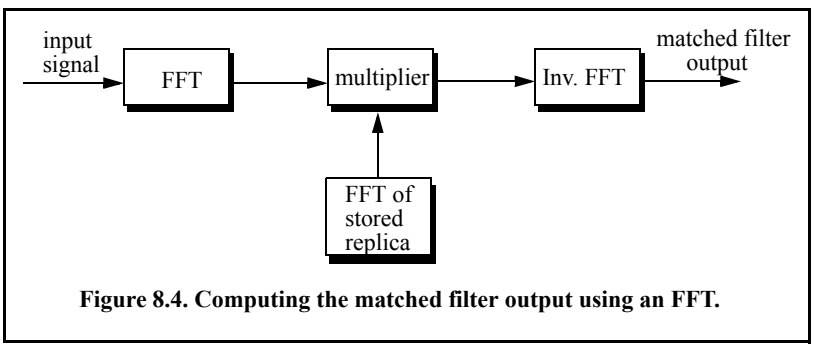

Figure 8.4. Computing the matched filter output using an FFT.

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,

$$y(t) = s((t) \otimes h(t)) \tag{8.9}$$

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

$$FFT\{s((t) \otimes h(t))\} = S(f) \cdot H(f) \tag{8.10}$$

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

$$y = FFT^{-1}\{S \cdot H\} \tag{8.11}$$

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 sidelobe levels of the compressed waveform. For the LFM case the first sidelobe is approximately

13.4dB below the main peak, and for most radar applications this may not be sufficient. In practice, high sidelobe levels are not preferable because noise and/or jammers located at the sidelobes 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 sidelobe 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). 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

$$R_{rec} = R_{max} - R_{min} \tag{8.12}$$

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

$$s(t) = \exp\left(j2\pi\left(f_0 t + \frac{\mu}{2}t^2\right)\right) \qquad 0 \le t \le \tau_0 \tag{8.13}$$

τ_0 is the pulse width, $\mu = B/\tau_0$, and B is the bandwidth.

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

$$s_r(t) = a_1 \exp\left(j2\pi\left(f_0(t-t_1) + \frac{\mu}{2}(t-t_1)^2\right)\right) \tag{8.14}$$

where a_1 is proportional to target RCS, antenna gain, and range attenuation. The time delay t_1 is given by

$$t_1 = 2R_1/c \tag{8.15}$$

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 lowpass filtering, is then given by

$$\phi(t) = 2\pi\left(-f_0 t_1 + \frac{\mu}{2}(t-t_1)^2\right) \tag{8.16}$$

and the instantaneous frequency is

$$f_i(t) = \frac{1}{2\pi}\frac{d}{dt}\phi(t) = \mu(t - t_1) = \frac{B}{\tau_0}\left(t - \frac{2R_1}{c}\right) \tag{8.17}$$

The quadrature components are

$$\begin{pmatrix} x_I(t) \\ x_Q(t) \end{pmatrix} = \begin{pmatrix} \cos\phi(t) \\ \sin\phi(t) \end{pmatrix} \tag{8.18}$$

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

$$f_s \geq 2B \tag{8.19}$$

and the sampling interval is

$$\Delta t \leq 1/2B \tag{8.20}$$

Using Eq. (8.17) it can be shown that (the proof is left as an exercise) the frequency resolution of the FFT is

$$\Delta f = 1/\tau_0 \tag{8.21}$$

The minimum required number of samples is

$$N = \frac{1}{\Delta f \Delta t} = \frac{\tau_0}{\Delta t} \tag{8.22}$$

Equating Eqs. (8.20) and (8.22) yields

$$N \geq 2B\tau_0 \tag{8.23}$$

Consequently, a total of $2B\tau_0$ real samples, or $B\tau_0$ complex samples, is sufficient to completely describe an LFM waveform of duration τ_0 and bandwidth B. For example, an LFM signal of duration $\tau_0 = 20\ \mu s$ 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 to the next power of two, by zero padding. Thus, the total number of samples, for some positive integer m, is

$$N_{FFT} = 2^m \geq N \tag{8.24}$$

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. 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

$$\phi(t) = \sum_{i=1}^{I} 2\pi\left(-f_0 t_i + \frac{\mu}{2}(t - t_i)^2\right) \tag{8.25}$$

The times $\{t_i = (2R_i/c);\ i = 1, 2, \ldots, I\}$ represent the two-way time delays, where t_1 coincides with the start of the receive window. As an example, consider the case where

# targets	R_{rec}	*pulse width*	*Band-width*	*targets range*	*Target RCS*	*Window type*
3	*200m*	*0.005ms*	*100e6 Hz*	*[10 75 120] m*	*[1 2 1]m^2*	*Ham-ming*

Note that the compressed pulsed range resolution is $\Delta R = 1.5m$. Figure 8.5 and Fig. 8.6 shows the real part and the amplitude spectrum of the replica used for this example. Figure 8.7 shows the uncompressed echo, while Fig. 8.8 shows the compressed MF output. Note that the scatterer amplitude attenuation is a function of the inverse of the scatterer's range within the receive window. Figure 8.9 is similar to Fig. 8.8, except in this case the first and second scatterers are less than 1.5 meter apart (they are at 70 and 71 meters).

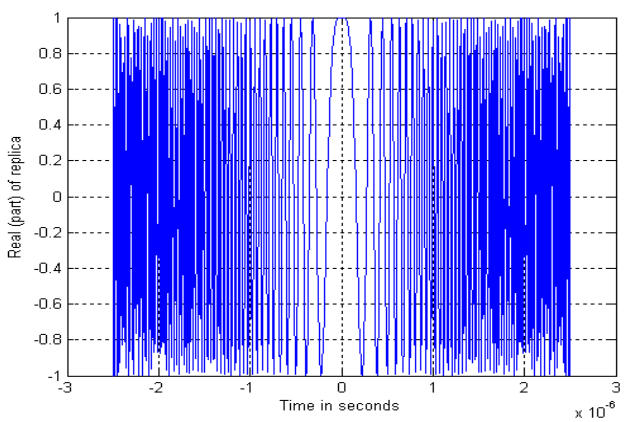

Figure 8.5. Real part of replica.

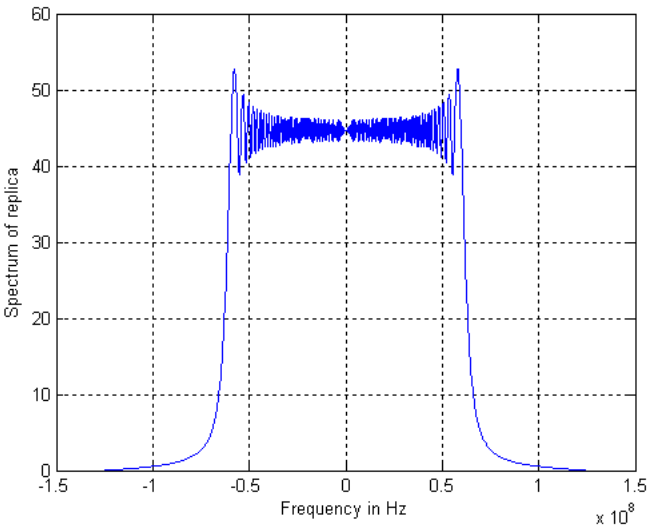

Figure 8.6. Replica spectrum.

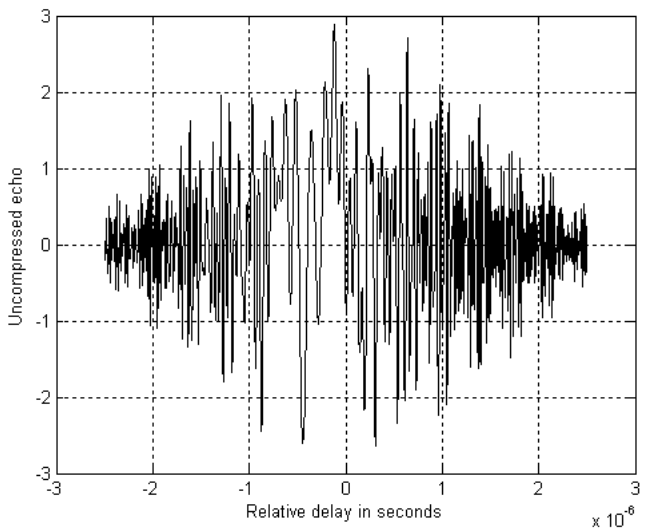

Figure 8.7. Uncompressed echo signal. Scatterers are not resolved.

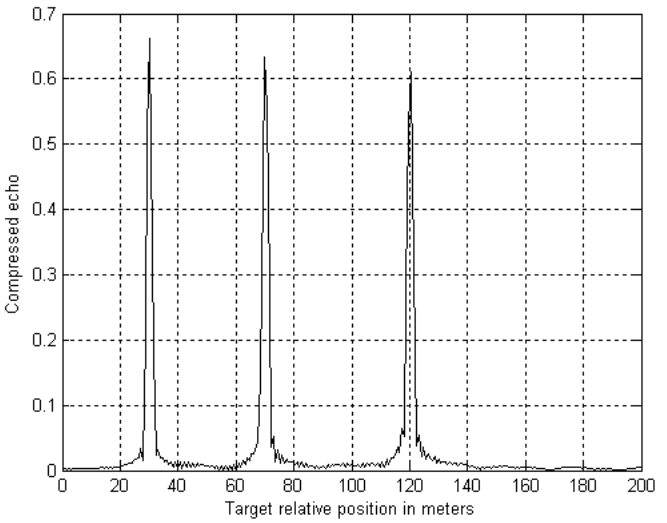

Figure 8.8. Compressed echo signal corresponding to Fig. 8.7. Scatterers are completely resolved.

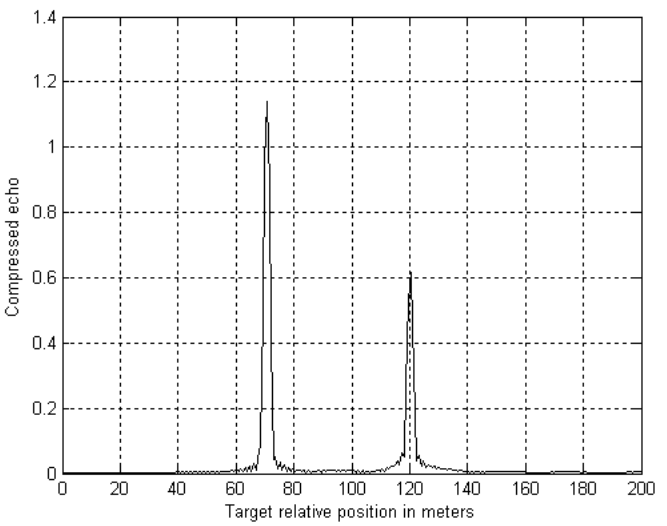

Figure 8.8. Compressed echo signal corresponding to Fig. 5.7. Scatterers are completely resolved.

8.5. 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.

8.5.1. Single LFM Pulse

Figure 8.10 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. 8.10. Mixing with the reference signal and performing lowpass 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

$$s_1(t) = \cos\left(2\pi\left(f_0 t + \frac{\mu}{2}t^2\right)\right) \qquad 0 \le t \le \tau_0 \tag{8.26}$$

where $\mu = B/\tau_0$ is the LFM coefficient and f_0 is the chirp start frequency. Assume a point scatterer at range R. The signal received by the radar is

$$s_r(t) = a\cos\left[2\pi\left(f_0(t-t_0) + \frac{\mu}{2}(t-t_0)^2\right)\right] \tag{8.27}$$

where a is proportional to target RCS, antenna gain, and range attenuation. The time delay t_0 is

$$t_0 = 2R/c \tag{8.28}$$

The reference signal is

$$s_{ref}(t) = 2\cos\left(2\pi\left(f_r t + \frac{\mu}{2}t^2\right)\right) \qquad 0 \le t \le T_{rec} \tag{8.29}$$

The receive window in seconds is

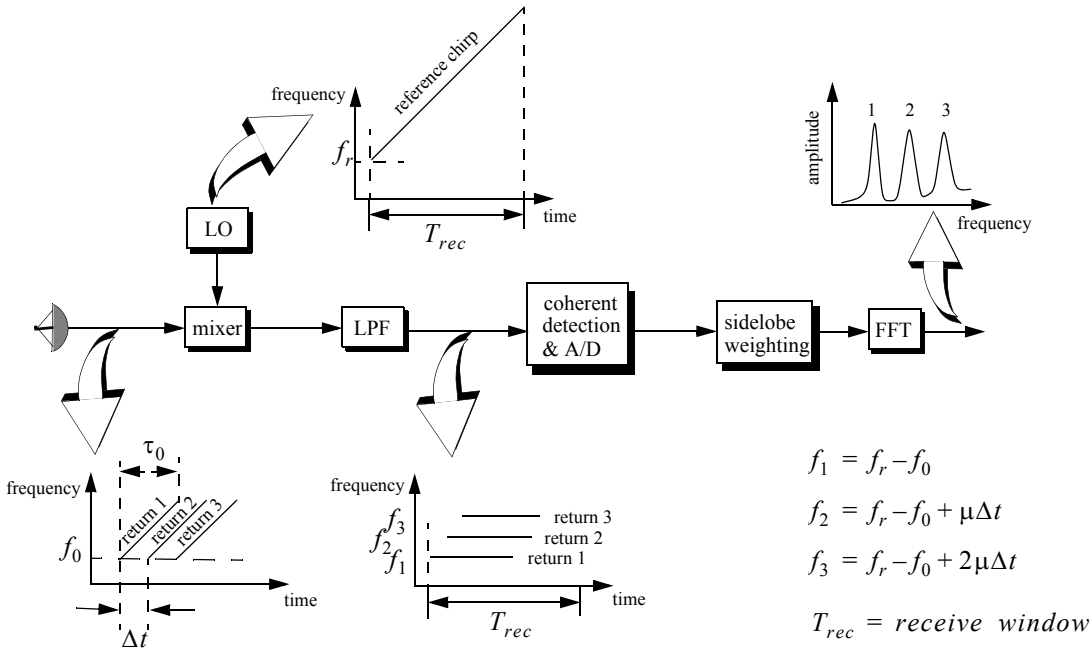

Figure 8.10. Stretch processing block diagram.

$$T_{rec} = \frac{2(R_{max} - R_{min})}{c} = \frac{2R_{rec}}{c} \tag{8.30}$$

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

$$s_0(t) = a\cos(2\pi f_0 t_0 + 2\pi\mu t_0 t - \pi\mu(t_0)^2) \tag{8.31}$$

Substituting Eq. (8.28) into Eq. (8.31) and collecting terms yield

$$s_0(t) = a \cos\left[\left(\frac{4\pi BR}{c\tau_0}\right)t + \frac{2R}{c}\left(2\pi f_0 - \frac{2\pi BR}{c\tau_0}\right)\right] \tag{8.32}$$

and since $\tau_0 \gg 2R/c$, Eq. (8.32) is approximated by

$$s_0(t) \approx a \cos\left[\left(\frac{4\pi BR}{c\tau_0}\right)t + \frac{4\pi R}{c} f_0\right] \tag{8.33}$$

The instantaneous frequency is

$$f_{inst} = \frac{1}{2\pi}\frac{d}{dt}\left(\left(\frac{4\pi BR}{c\tau_0}t + \frac{4\pi R}{c}f_0\right)\right) = \frac{2BR}{c\tau_0} \tag{8.34}$$

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

$$R_1 = f_1 c\tau_0 / 2B \tag{8.35}$$

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

$$s_r(t) = \sum_{i=1}^{I} a_i(t)\cos\left[2\pi\left(f_0(t - t_i) + \frac{\mu}{2}(t - t_i)^2\right)\right] \tag{8.36}$$

where $\{a_i(t);\ i = 1, 2, \ldots, I\}$ are proportional to the targets' cross sections, antenna gain, and range. The times $\{t_i = (2R_i/c);\ i = 1, 2, \ldots, I\}$ represent the two-way time delays, where t_1 coincides with the start of the receive window. Using Eq. (8.32) the overall signal at the output of the LPF can then be described by

$$s_o(t) = \sum_{i=1}^{I} a_i \cos\left[\left(\frac{4\pi BR_i}{c\tau_0}\right)t + \frac{2R_i}{c}\left(2\pi f_0 - \frac{2\pi BR_i}{c\tau_0}\right)\right] \tag{8.37}$$

Hence, target returns appear as 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_0 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 from a few meters to possibly less than 100 meters. The compressed pulse range resolution is computed from Eq. (8.8). Declare the FFT size to be N and its frequency resolution to be Δf. The frequency resolution can be computed using the following procedure: Consider two adjacent point scatterers at ranges R_1 and R_2. The minimum frequency separation, Δf, between those scatterers so that they are resolved can be computed from Eq. (8.34). More precisely,

$$\Delta f = f_2 - f_1 = \frac{2B}{c\tau_0}(R_2 - R_1) = \frac{2B}{c\tau_0}\Delta R \tag{8.38}$$

Substituting Eq. (8.8) into Eq. (8.38) yields

$$\Delta f = \frac{2B}{c\tau_0}\ \frac{c}{2B} = \frac{1}{\tau_0} \tag{8.39}$$

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

$$\frac{N\Delta f}{2} > \frac{2B(R_{max} - R_{min})}{c\tau_0} = \frac{2BR_{rec}}{c\tau_0} \tag{8.40}$$

Using Eqs. (8.30) and (8.39) into Eq. (8.40) and collecting terms yield

$$N > 2BT_{rec} \tag{8.41}$$

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

$$N_{FFT} \geq N = 2^m \tag{8.42}$$

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

$$\Delta f = \frac{1}{T_s N_{FFT}} \Rightarrow T_s = \frac{1}{\Delta f N_{FFT}} \tag{8.43}$$

As an example, consider the case where

# targets	*3*
pulsewidth	*10 ms*
center frequency	*5.6 GHz*
bandwidth	*1 GHz*
receive window	*30 m*
relative target's range	*[2 5 10] m*
target's RCS	*[1, 1, 2] m^2*
window	*2 (Kaiser)*

Note that the compressed pulse range resolution, without using a window, is $\Delta R = 0.15m$. Figure 8.11 and Fig. 8.12, respectively, show the uncompressed and compressed echo signals corresponding to this example. Figures 8.13 a and b are similar to Fig. 8.11 and Fig. 8.12 except in this case two of the scatterers are less than 15 cm apart (i.e., unresolved targets at $R_{relative} = [3, 3.1]m$).

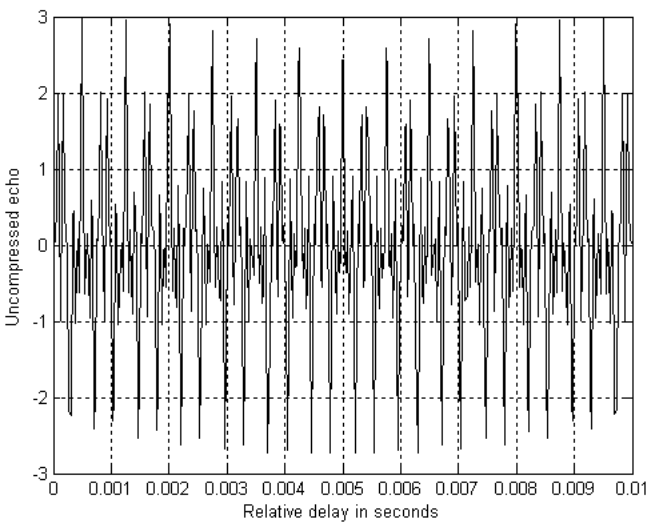

Figure 8.11. Uncompressed echo signal. Three targets are unresolved.

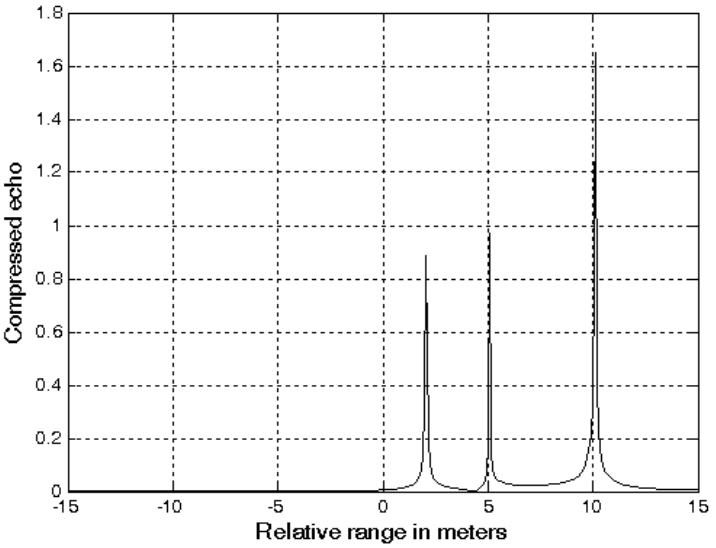

Figure 8.12. Compressed echo signal. Three targets are resolved.

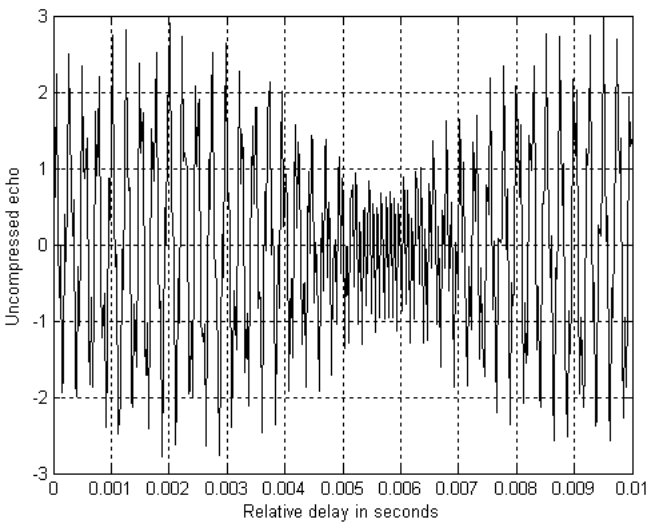

Figure 8.13a. Uncompressed echo signal. Three targets.

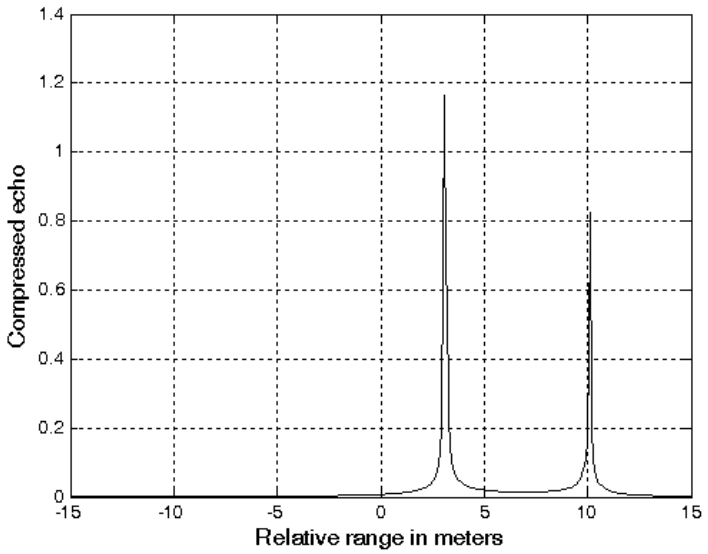

Figure 8.13b. Compressed echo signal. Three targets, two are not resolved.

8.5.2. Stepped Frequency Waveforms

Stepped Frequency Waveforms (SFW) are used in extremely wide band radar applications where very large time bandwidth product is required. Generation of SFW was discussed in Chapter 5. For this purpose, consider an LFM signal whose bandwidth is B_i and whose pulsewidth is T_i and refer to it as the primary LFM. Divide this long pulse into N subpulses each of width τ_0 to generate a sequence of pulses whose PRI is denoted by T. It follows that $T_i = (n-1)T$. Define the beginning frequency for each subpulse as that value measured from the primary LFM at the leading edge of each subpulse, as illustrated in Fig. 8.14. That is

$$f_i = f_0 + i\Delta f; \;\; i = 0, N-1 \tag{8.44}$$

where Δf is the frequency step from one subpulse to another. The set of n subpulses is often referred to as a burst. Each subpulse can have its own LFM modulation. To this end, assume that each subpluse is of width τ_0 and bandwidth B, then the LFM slope of each pulse is

$$\mu = \frac{B}{\tau_0} \tag{8.45}$$

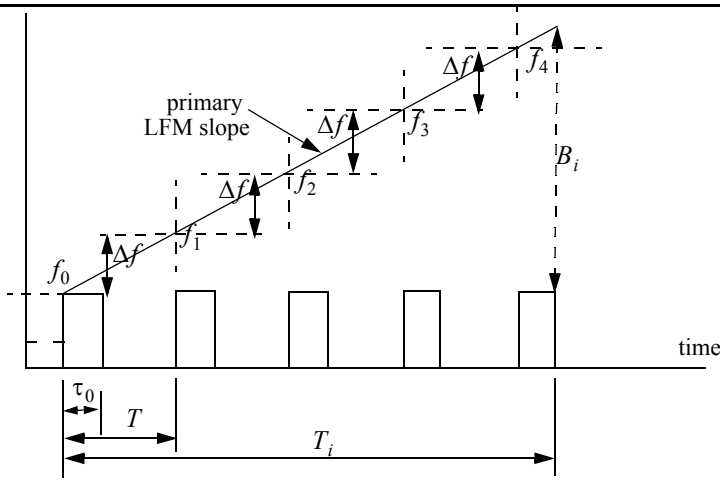

Figure 8.14. Example of stepped frequency waveform burst; $N = 5$.

The SFW operation and processing involve the following steps:

1. A series of N narrow-band LFM pulses is transmitted. The chirp beginning frequency from pulse to pulse is stepped by a fixed frequency step Δf, as defined in Eq. (8.44). Each group of N pulses is referred to as a burst.
2. The LFM slope (quadratic phase term) is first removed from the received signal, as described in Fig. 8.10. The reference slope must be equal to the combined primary LFM and single subpulse slopes. Thus, the received signal is reduced to a series of subpulses.
3. These subpulses are then sampled at a rate that coincides with the center of each pulse, sampling rate equivalent to $(1/T)$.
4. The quadrature components for each burst are collected and stored.
5. Spectral weighting (to reduce the range sidelobe levels) is applied to the quadrature components. Corrections for target velocity, phase, and amplitude variations are applied.
6. The IDFT of the weighted quadrature components of each burst is calculated to synthesize a range profile for that burst. The process is repeated for M bursts to obtain consecutive high resolution range profiles.

Within a burst, the transmitted waveform for the i^{th} step can be described as

$$x_i(t) = \begin{cases} C_i \dfrac{1}{\sqrt{\tau_0}} Rect\left(\dfrac{t}{\tau_0}\right) e^{j2\pi\left(f_i t + \frac{\mu}{2}t^2\right)} & ; \; iT \le t \le iT + \tau_0 \\ 0 & elsewhere \end{cases} \qquad \textbf{(3.46)}$$

where C_i are constants. The received signal from a target located at range R_0 is then given by

$$x_{ri}(t) = C_i' e^{j2\pi\left[f_i(t-\Delta(t))-\frac{\mu}{2}(t-\Delta(t))^2\right]}, \quad iT+\Delta(t) \le t \le iT+\tau_0+\Delta(t) \tag{8.47}$$

where C_i' are constant and the round trip delay $\Delta(t)$ is given by

$$\Delta(t) = \frac{R_0 - vt}{c/2} \tag{8.48}$$

where c is the speed of light and v is the target radial velocity.

In order to remove the quadratic phase term, mixing is first performed with the reference signal given by

$$y_i(t) = e^{j2\pi\left(f_i t+\frac{\mu}{2}t^2\right)} \quad ; \; iT \le t \le iT+\tau_0 \tag{8.49}$$

Next lowpass filtering is performed to extract the quadrature components. More precisely, the quadrature components are given by

$$\begin{pmatrix} x_I(t) \\ x_Q(t) \end{pmatrix} = \begin{pmatrix} A_i \cos\phi_i(t) \\ A_i \sin\phi_i(t) \end{pmatrix} \tag{8.50}$$

where A_i are constants, and

$$\phi_i(t) = -2\pi f_i\left(\frac{2R_0}{c} - \frac{2vt}{c}\right) \tag{8.51}$$

where now $f_i = \Delta f$. For each pulse, the quadrature components are then sampled at

$$t_i = iT + \frac{\tau_r}{2} + \frac{2R_0}{c} \tag{8.52}$$

τ_r is the time delay associated with the range that corresponds to the start of the range profile.

The quadrature components can then be expressed in complex form as

$$X_i = A_i e^{j\phi_i} \tag{8.53}$$

Equation (8.53) represents samples of the target reflectivity, due to a single burst, in the frequency domain. This information can then be transformed into a series of range delay reflectivity (i.e., range profile) values by using the IDFT. It follows that

$$H_l = \frac{1}{N}\sum_{i=0}^{N-1} X_i \ \exp\left(j\frac{2\pi li}{N}\right) \qquad ;\ 0 \le l \le N-1 \tag{8.54}$$

Substituting Eq. (8.51) and Eq. (8.53) into (8.54) and collecting terms yield

$$H_l = \frac{1}{N}\sum_{i=0}^{N-1} A_i \ \exp\left\{j\left(\frac{2\pi li}{N} - 2\pi f_i\left(\frac{2R_0}{c} - \frac{2vt_i}{c}\right)\right)\right\} \tag{8.55}$$

By normalizing with respect to N and by assuming that $A_i = 1$ and that the target is stationary (i.e., $v = 0$), then Eq. (8.55) can be written as

$$H_l = \sum_{i=0}^{N-1} \exp\left\{j\left(\frac{2\pi li}{N} - 2\pi f_i\frac{2R_0}{c}\right)\right\} \tag{8.56}$$

Using $f_i = i\Delta f$ inside Eq. (8.56) yields

$$H_l = \sum_{i=0}^{N-1} \exp\left\{j\frac{2\pi i}{N}\left(-\frac{2NR_0\Delta f}{c} + l\right)\right\} \tag{8.57}$$

which can be simplified to

$$H_l = \frac{\sin\pi\zeta}{\sin\frac{\pi\zeta}{N}} \ \exp\left(j\frac{N-1}{2} \ \frac{2\pi\zeta}{N}\right) \tag{8.58}$$

where

$$\zeta = \frac{-2NR_0\Delta f}{c} + l \tag{8.59}$$

Finally, the synthesized range profile is

$$|H_l| = \left|\frac{\sin\pi\zeta}{\sin\frac{\pi\zeta}{N}}\right| \tag{8.60}$$

Range Resolution and Range Ambiguity in SFW

As usual, range resolution is determined from the overall system bandwidth. Assuming an SFW with N steps and step size Δf, then the corresponding range resolution is equal to

$$\Delta R = \frac{c}{2N\Delta f} \tag{8.61}$$

Range ambiguity associated with an SFW can be determined by examining the phase term that corresponds to a point scatterer located at range R_0. More precisely,

$$\phi_i(t) = 2\pi f_i \frac{2R_0}{c} \tag{8.62}$$

It follows that

$$\frac{\Delta\phi}{\Delta f} = \frac{4\pi(f_{i+1} - f_i)}{(f_{i+1} - f_i)}\frac{R_0}{c} = \frac{4\pi R_0}{c} \tag{8.63}$$

or equivalently,

$$R_0 = \frac{\Delta\phi}{\Delta f}\frac{c}{4\pi} \tag{8.64}$$

It is clear from Eq. (8.64) that range ambiguity exists for $\Delta\phi = \Delta\phi + 2N\pi$. Therefore,

$$R_0 = \frac{\Delta\phi + 2N\pi}{\Delta f}\frac{c}{4\pi} = R_0 + N\left(\frac{c}{2\Delta f}\right) \tag{8.65}$$

and the unambiguous range window is

$$R_u = \frac{c}{2\Delta f} \tag{8.66}$$

A range profile synthesized using a particular SFW represents the relative range reflectivity for all scatterers within the unambiguous range window, with respect to the absolute range that corresponds to the burst time delay. Additionally, if a specific target extent is larger than R_u, then all scatterers falling outside the unambiguous range window will fold over and appear in the synthesized profile. This fold-over problem is identical to the spectral fold-over that occurs when using a Fast Fourier Transform (FFT) to resolve certain signal frequency contents. For example, consider an FFT with frequency resolution $\Delta f = 50Hz$ and size $NFFT = 64$. In this case, this FFT can resolve frequency tones between $-1600Hz$ and $1600Hz$. When this FFT is used to resolve the frequency content of a sine-wave tone equal to $1800Hz$, fold-over occurs and a spectral line at the fourth FFT bin (i.e., $200Hz$) appears. Therefore, in order to avoid fold-over in the synthesized range profile, the frequency step Δf must be

$$\Delta f \leq c/2E \tag{8.67}$$

where E is the target extent in meters.

Additionally, the pulsewidth must also be large enough to contain the whole target extent. Thus,

$$\Delta f \leq 1/\tau_0 \tag{8.68}$$

and in practice,

$$\Delta f \leq 1/2\tau_0 \tag{8.69}$$

This is necessary in order to reduce the amount of contamination of the synthesized range profile caused by the clutter surrounding the target under consideration.

For example, assume that the range profile starts at $R_0 = 900m$ and that

# targets	*pulsewidth*	*N*	Δf	*1/T*	*v*
3	100μsec	*64*	$10MHz$	$100KHz$	*0.0*

In this case,

$$\Delta R = \frac{3\times 10^8}{2\times 64\times 10\times 10^6} = 0.235m, \text{ and } R_u = \frac{3\times 10^8}{2\times 10\times 10^6} = 15m$$

Thus, scatterers that are more than 0.235 meters apart will appear as distinct peaks in the synthesized range profile. Assume two cases; in the first case, [scat_range] = [908, 910, 912] meters, and in the second case, [scat_range] = [908, 910, 910.2] meters. In both cases, let [scat_rcs] = [100, 10, 1] meters squared. Figure 8.15 shows the synthesized range profiles generated using the function "SWF.m" and the first case when the Hamming window is not used. Figure 8.16 is similar to Fig. 8.15, except in this case the Hamming window is used. Figure 8.17 shows the synthesized range profile that corresponds to the second case (Hamming window is used). Note that all three scatterers were resolved in Fig. 8.15 and Fig. 8.16; however, the last two scatterers are not resolved in Fig. 8.17, because they are separated by less than ΔR.

Next, consider another case where *[scat_range] = [908, 912, 916] meters.* Figure 8.18 shows the corresponding range profile. In this case, foldover occurs, and the last scatterer appears at the lower portion of the synthesized range profile. Also, consider the case where [scat_range] = [908, 910, 923] meters. Figure 8.19 shows the corresponding range profile. In this case, ambiguity is associated with the first and third scatterers since they are separated by $15m$. Both appear at the same range bin.

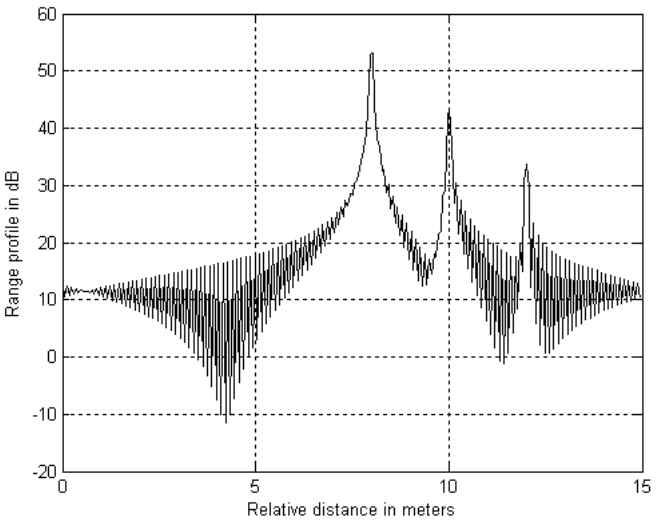

Figure 8.15. Synthetic range profile for three resolved scatterers. No window.

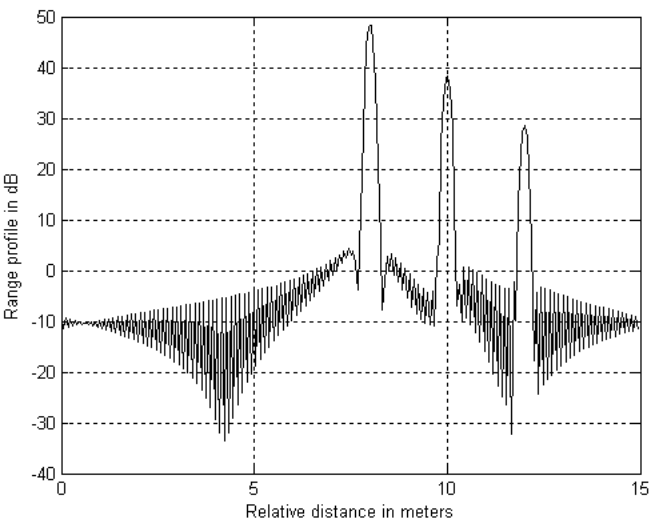

Figure 8.16. Synthetic range profile for three scatterers. Hamming window.

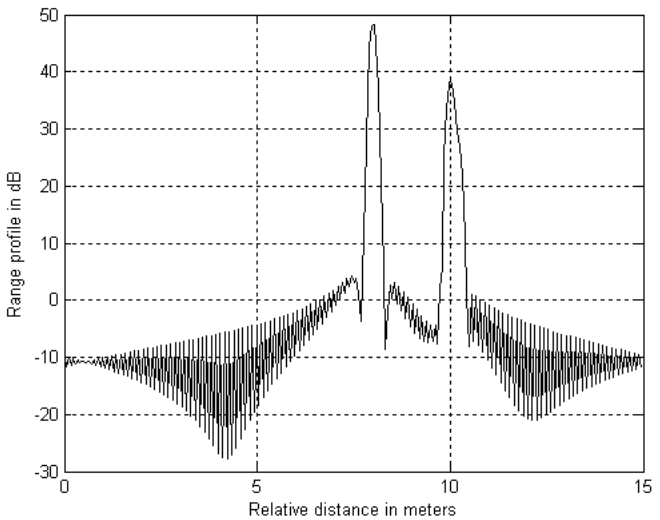

Figure 8.17. Synthetic range profile for three scatterers. Two are unresolved.

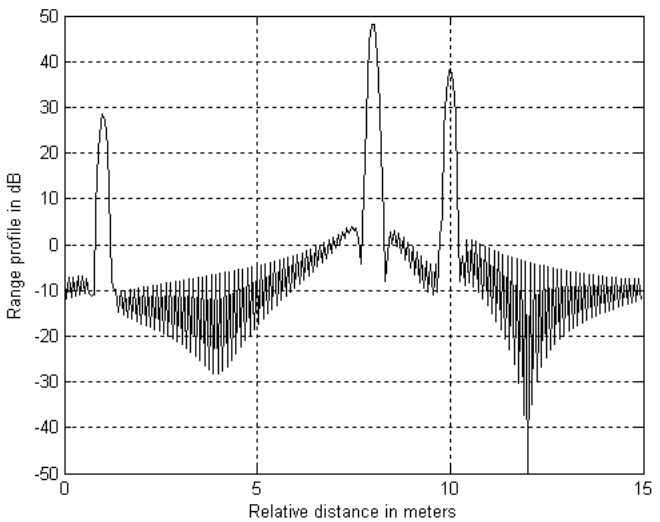

Figure 8.18. Synthetic range profile for three scatterers. Third scatterer folds over.

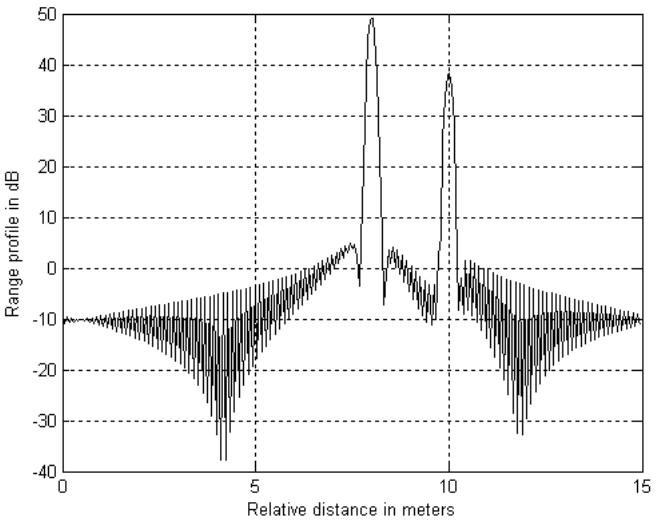

Figure 8.19. Synthetic range profile for three scatterers. The first and third scatterers appear in the same FFT bin.

8.5.2.1. Effect of Target Velocity

The range profile defined in Eq. (8.60) is obtained by assuming that the target under examination is stationary. The effect of target velocity on the synthesized range profile can be determined by starting with Eq. (8.55) and assuming that $v \neq 0$. Performing similar analysis as that of the stationary target case yields a range profile given by

$$H_l = \sum_{i=0}^{N-1} A_i \exp\left\{ j\frac{2\pi li}{N} - j2\pi f_i\left[\frac{2R}{c} - \frac{2v}{c}\left(iT + \frac{\tau_r}{2} + \frac{2R}{c}\right)\right]\right\} \qquad \textbf{(8.70)}$$

The additional phase term present in Eq. (8.70) distorts the synthesized range profile. In order to illustrate this distortion, consider the SFW described in the previous section, and assume the three scatterers of the first case. Also, assume that $v = 200 m/s$. Figure 8.20 shows the synthesized range profile for this case. Comparisons of Figs. 8.16 and 8.20 clearly show the distortion effects caused by the uncompensated target velocity. Figure 8.21 is similar to Fig. 8.20 except in this case, $v = -200 m/s$. Note in either case, the targets have moved from their expected positions (to the left or right) by $Disp = 2 \times n \times v/PRF$ *(1.28 m).*

This distortion can be eliminated by multiplying the complex received data at each pulse by the phase term

$$\Phi = \exp\left(-j2\pi f_i\left[\frac{2\hat{v}}{c}\left(iT+\frac{\tau_r}{2}+\frac{2\hat{R}}{c}\right)\right]\right) \quad \textbf{(3.71)}$$

$\hat{v}$ and R are, respectively, estimates of the target velocity and range. This process of modifying the phase of the quadrature components is often referred to as "phase rotation." In practice, when good estimates of $\hat{v}$ and $\hat{R}$ are not available, then the effects of target velocity are reduced by using frequency hopping between the consecutive pulses within the SFW. In this case, the frequency of each individual pulse is chosen according to a predetermined code. Waveforms of this type are often called Frequency Coded Waveforms (FCW). Costas waveforms or signals are a good example of this type of waveform.

Figure 8.22 shows a synthesized range profile for a moving target whose RCS is $\sigma = 10m^2$ and $v = 10m/s$. The initial target range is at $R = 912m$. All other parameters are as before. This figure can be reproduced using the following MATLAB code.

```
clear all;
close all;
nscat = 1;
scat_range = 912;
scat_rcs = 10;
n =64;
deltaf = 10e6;
prf = 10e3;
v = 10;
rnote = 900,
winid = 1;
count = 0;
for time = 0:.05:3
   count = count +1;
   hl = SFW (nscat, scat_range, scat_rcs, n, deltaf, prf, v, rnote, winid);
   array(count,:) = transpose(hl);
   hl(1:end) = 0;
   scat_range =  scat_range - 2 * n * v / prf;
end
figure (1)
 numb = 2*256;% this number matches that used in hrr_profile.
 delx_meter = 15 / numb;
 xmeter = 0:delx_meter:15-delx_meter;
 imagesc(xmeter, 0:0.05:4,array)
 colormap(gray)
ylabel ('Time in seconds')
xlabel('Relative distance in meters')
```

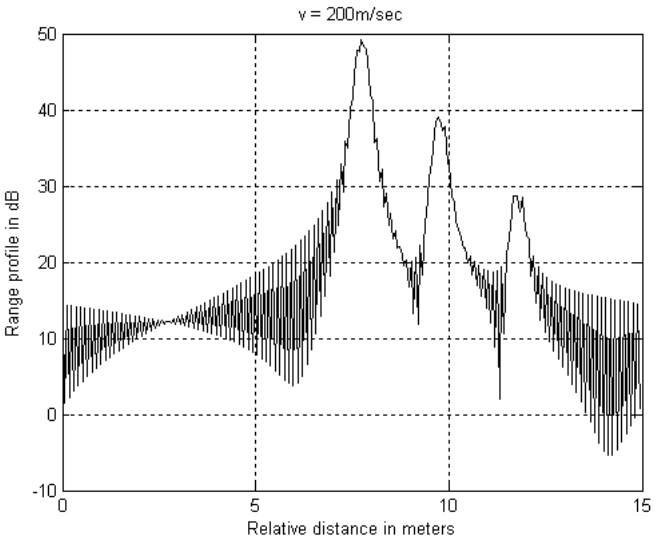

Figure 8.20. Illustration of range profile distortion due to target velocity.

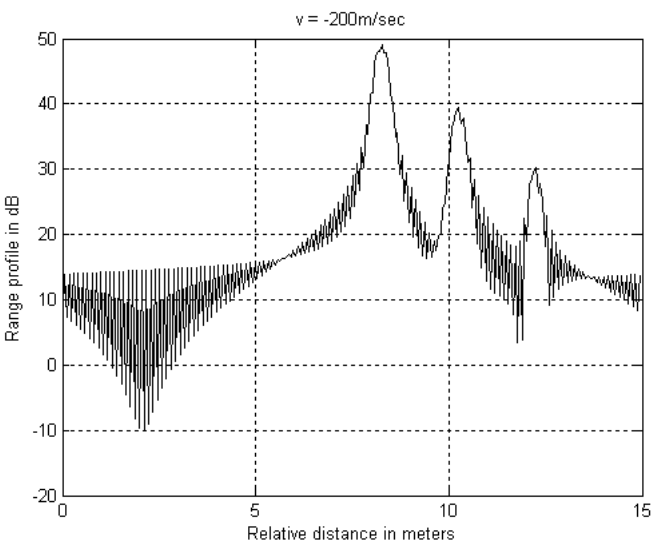

Figure 8.21. Illustration of range profile distortion due to target velocity.

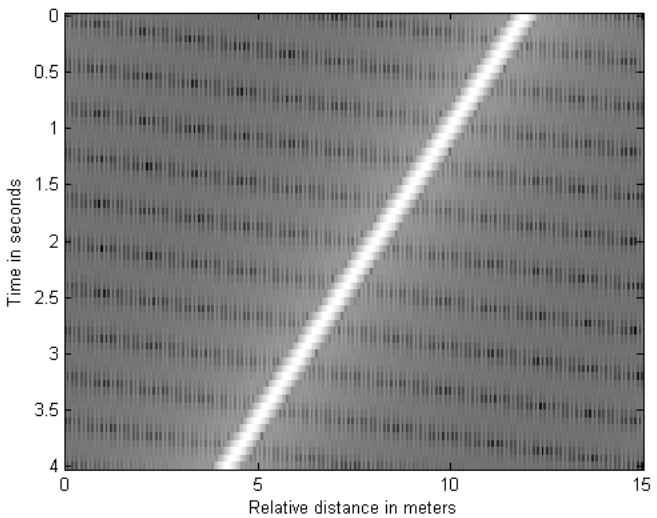

Figure 8.22. Synthesized range profile for a moving target (4 seconds long).

8.6. MATLAB Program Listings

This section presents listings for all the MATLAB programs used to produce all of the MATLAB-generated figures in this chapter.

8.6.1. MATLAB Function "matched_filter.m"

The function *"matched_filter.m"* performs fast convolution processing. The user can access this function either by a MATLAB function call or by executing the MATLAB program *"matched_filter_gui.m,"* which utilizes a MATLAB-based GUI. The work space associated with this program is shown in Fig. 8.23. The outputs for this function include plots of the compressed and uncompressed signals as well as the replica used in the pulse compression process. This function utilizes the function *"power_integer_2.m."*

The function *"matched_filter.m"* syntax is as follows:

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

where

Symbol	Description	Units	Status
nscat	*number of point scatterers within the received window*	*none*	*input*
rrec	*receive window size*	*m*	*input*
taup	*uncompressed pulse width*	*seconds*	*input*
b	*chirp bandwidth*	*Hz*	*input*
scat_range	*vector of scatterers' relative range (within the receive window)*	*m*	*input*
scat_rcs	*vector of scatterers' RCS*	m^2	*input*
win	*0 = no window* *1 = Hamming* *2 = Kaiser with parameter pi* *3 = Chebychev side-lobes at -60dB*	*none*	*input*
y	*normalized compressed output*	*volts*	*output*

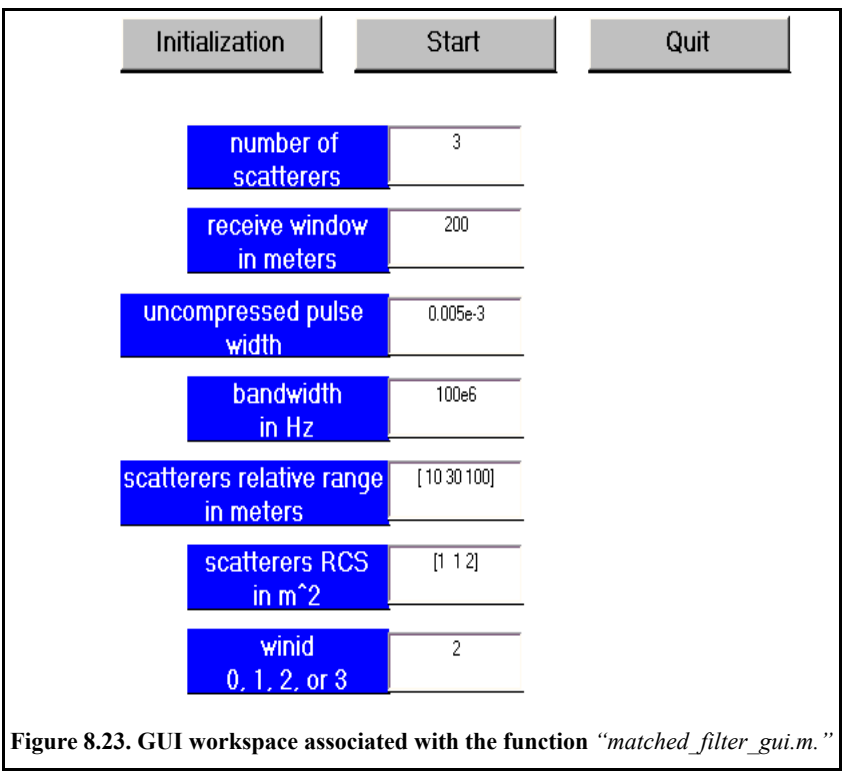

Figure 8.23. GUI workspace associated with the function *"matched_filter_gui.m."*

MATLAB Function "matched_filter.m" Listing

```
function [y] = matched_filter(nscat,taup,b,rrec,scat_range,scat_rcs,winid)
eps = 1.0e-16;
% time bandwidth product
time_B_product = b * taup;
if(time_B_product < 5 )
   fprintf('************ Time Bandwidth product is TOO SMALL **************')
   fprintf('\n Change b and or taup')
  return
end
% speed of light
c = 3.e8;
% number of samples
n = fix(5 * taup * b);
% initialize input, output, and replica vectors
x(nscat,1:n) = 0.;
y(1:n) = 0.;
replica(1:n) = 0.;
% determine proper window
if( winid == 0.)
  win(1:n) = 1.;
end
if(winid == 1.);
   win = hamming(n)';
end
if( winid == 2.)
   win = kaiser(n,pi)';
end
if(winid == 3.)
   win = chebwin(n,60)';
end
% check to ensure that scatterers are within recieve window
index = find(scat_range > rrec);
if (index ~= 0)
   'Error. Receive window is too large; or scatterers fall outside window'
  return
end
% calculate sampling interval
t = linspace(-taup/2,taup/2,n);
replica = exp(i * pi * (b/taup) .* t.^2);
figure(1)
subplot(2,1,1)
plot(t,real(replica))
ylabel('Real (part) of replica')
xlabel('Time in seconds')
grid
subplot(2,1,2)
```

```
sampling_interval = taup / n;
freqlimit = 0.5/ sampling_interval;
freq = linspace(-freqlimit,freqlimit,n);
plot(freq,fftshift(abs(fft(replica))));
ylabel('Spectrum of replica')
xlabel('Frequency in Hz')
grid
 for j = 1:1:nscat
    range = scat_range(j) ;
    x(j,:) = scat_rcs(j) .* exp(i * pi * (b/taup) .* (t +(2*range/c)).^2) ;
    y = x(j,:)  + y;
end
figure(2)
 y = y .* win;
plot(t,real(y),'k')
xlabel ('Relative delay in seconds')
ylabel ('Uncompressed echo')
grid
out =xcorr(replica, y);
out = out ./ n;
s = taup * c /2;
Npoints = ceil(rrec * n /s);
dist =linspace(0, rrec, Npoints);
delr = c/2/b;
figure(3)
plot(dist,abs(out(n:n+Npoints-1)),'k')
xlabel ('Target relative position in meters')
ylabel ('Compressed echo')
grid
return
```

MATLAB Function *"power_integer_2.m"* ***Listing***

```
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
return
```

8.6.2. MATLAB Function "stretch.m"

The function *"stretch.m"* presents a digital implementation of stretch processing. The syntax is as follows:

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

where

Symbol	Description	Units	Status
nscat	*number of point scatterers within the receive window*	*none*	*input*
taup	*uncompressed pulse width*	*seconds*	*input*
f0	*chirp start frequency*	*Hz*	*input*
b	*chirp bandwidth*	*Hz*	*input*
scat_range	*vector of scatterers' range*	*m*	*input*
rrec	*range receive window*	*m*	*input*
scat_rcs	*vector of scatterers' RCS*	m^2	*input*
win	*0 = no window* *1 = Hamming* *2 = Kaiser with parameter pi* *3 = Chebychev side-lobes 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_gui.m,"* which utilizes MATLAB-based GUI and is shown in Fig. 8.24. The outputs of this function are the complex array *y* and plots of the uncompressed and compressed echo signal versus time.

MATLAB Function "stretch.m" Listing

```
function [y] = stretch(nscat, taup, f0, b, scat_range, rrec, 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:n) = 0.;
y(1:n) = 0.;
if( winid == 0.)
  win(1:n) = 1.;
```

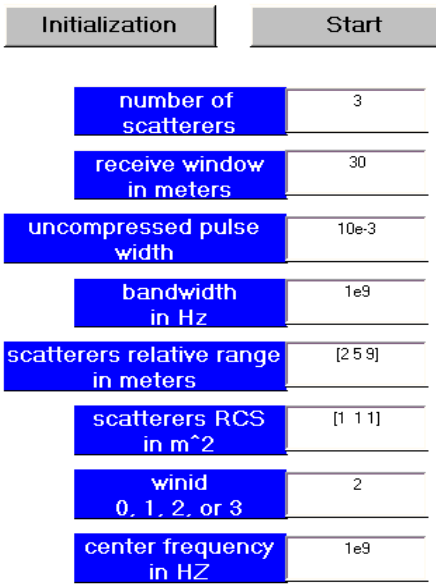

Figure 8.24. GUI workspace associated with the function *"stretch_gui.m."*

```
 win =win';
else
  if(winid == 1.)
    win = hamming(n);
  else
    if( winid == 2.)
      win = kaiser(n,pi);
    else
      if(winid == 3.)
        win = chebwin(n,60);
      end
    end
  end
end
deltar = c / 2. / b;
max_rrec = deltar * nfft / 2.;
maxr = max(scat_range);
if(rrec > max_rrec | maxr >= rrec )
  'Error. Receive window is too large; or scatterers fall outside window'
  return
end
```

```
t = linspace(0,taup,n);
for j = 1:1:nscat
    range = scat_range(j);% + rmin;
   psi1 = 4. * pi * range * f0 / c - ...
      4. * pi * b * range * range / c / c/ taup;
   psi2 = (2*4. * pi * b * range / c / taup) .* t;
   x(j,:) = scat_rcs(j) .* exp(i * psi1 + i .* psi2);
   y = y + x(j,:);
end
figure(1)
plot(t,real(y),'k')
xlabel ('Relative delay in seconds')
ylabel ('Uncompressed echo')
grid
ywin = y .* win';
yfft = fft(y,n) ./ n;
out= fftshift(abs(yfft));
figure(2)
delinc = rrec/ n;
%dist = linspace(-delinc-rrec/2,rrec/2,n);
dist = linspace((-rrec/2), rrec/2,n);
plot(dist,out,'k')
xlabel ('Relative range in meters')
ylabel ('Compressed echo')
axis auto
grid
```

8.6.3. MATLAB Function "SFW.m"

The function *"SFW.m"* computes and plots the range profile for a specific SFW. This function utilizes an Inverse Fast Fourier Transform (IFFT) of a size equal to twice the number of steps. Hamming window of the same size is also assumed. The syntax is as follows:

[hl] = SFW (nscat, scat_range, scat_rcs, n, deltaf, prf, v, r0, winid)

where

Symbol	Description	Units	Status
nscat	*number of scatterers that make up the target*	*none*	*input*
scat_range	*vector containing range to individual scatterers*	*meters*	*input*
scat_rcs	*vector containing RCS of individual scatterers*	*meter square*	*input*
n	*number of steps*	*none*	*input*

Symbol	Description	Units	Status
deltaf	*frequency step*	*Hz*	*input*
prf	*PRF of SFW*	*Hz*	*input*
v	*target velocity*	*meter/second*	*input*
r0	*profile starting range*	*meters*	*input*
winid	*number>0 for Hamming window* *number < 0 for no window*	*none*	*input*
hl	*range profile*	*dB*	*output*

MATLAB Function "SFW.m" Listing

```
function [hl] = SFW (nscat, scat_range, scat_rcs, n, deltaf, prf, v, rnote,winid)
% Range or Time domain Profile
% Range_Profile returns the Range or Time domain plot of a simulated
% HRR SFWF returning from a predetermined number of targets with a predetermined
% RCS for each target.
c=3.0e8;  % speed of light (m/s)
num_pulses  = n;
SNR_dB = 40;
nfft = 256;
% carrier_freq = 9.5e9; %Hz (10GHz)
freq_step    = deltaf; %Hz (10MHz)
V = v;  % radial velocity (m/s)  -- (+)=towards radar (-)=away
PRI = 1. / prf; % (s)
if (nfft > 2*num_pulses)
   num_pulses = nfft/2;
else
end
Inphase = zeros((2*num_pulses),1);
Quadrature = zeros((2*num_pulses),1);
Inphase_tgt    = zeros(num_pulses,1);
Quadrature_tgt = zeros(num_pulses,1);
IQ_freq_domain = zeros((2*num_pulses),1);
Weighted_I_freq_domain = zeros((num_pulses),1);
Weighted_Q_freq_domain = zeros((num_pulses),1);
Weighted_IQ_time_domain = zeros((2*num_pulses),1);
Weighted_IQ_freq_domain = zeros((2*num_pulses),1);
abs_Weighted_IQ_time_domain = zeros((2*num_pulses),1);
dB_abs_Weighted_IQ_time_domain = zeros((2*num_pulses),1);
taur = 2. * rnote / c;
for jscat = 1:nscat
  ii = 0;
  for i = 1:num_pulses
     ii = ii+1;
```

```
    rec_freq = ((i-1)*freq_step);
    Inphase_tgt(ii) = Inphase_tgt(ii) + sqrt(scat_rcs(jscat)) * cos(-2*pi*rec_freq*...
      (2.*scat_range(jscat)/c - 2*(V/c)*((i-1)*PRI + taur/2 + 2*scat_range(jscat)/c)));
            Quadrature_tgt(ii) = Quadrature_tgt(ii) + sqrt(scat_rcs(jscat))*sin(-
2*pi*rec_freq*...
      (2*scat_range(jscat)/c - 2*(V/c)*((i-1)*PRI + taur/2 + 2*scat_range(jscat)/c)));
  end
end
if(winid >= 0)
   window(1:num_pulses) = hamming(num_pulses);
else
   window(1:num_pulses) = 1;
end
Inphase = Inphase_tgt;
Quadrature = Quadrature_tgt;
Weighted_I_freq_domain(1:num_pulses) = Inphase(1:num_pulses).* window';
Weighted_Q_freq_domain(1:num_pulses) = Quadrature(1:num_pulses).* window';
Weighted_IQ_freq_domain(1:num_pulses)= Weighted_I_freq_domain + ...
  Weighted_Q_freq_domain*j;
Weighted_IQ_freq_domain(num_pulses:2*num_pulses)=0.+0.i;
Weighted_IQ_time_domain = (ifft(Weighted_IQ_freq_domain));
abs_Weighted_IQ_time_domain = (abs(Weighted_IQ_time_domain));
dB_abs_Weighted_IQ_time_domain =
20.0*log10(abs_Weighted_IQ_time_domain)+SNR_dB;
% calculate the unambiguous range window size
Ru = c /2/deltaf;
hl = dB_abs_Weighted_IQ_time_domain;
 numb = 2*num_pulses;
delx_meter = Ru / numb;
xmeter = 0:delx_meter:Ru-delx_meter;
plot(xmeter, dB_abs_Weighted_IQ_time_domain,'k')
xlabel ('Relative distance in meters')
ylabel ('Range profile in dB')
grid
```