Chapter 7 Target Detection and Pulse Integration

7.1. Target Detection in the Presence of Noise

A simplified block diagram of a radar receiver that employs an envelope detector followed by a threshold decision is shown in Fig. 7.1. The input signal to the receiver is composed of the radar echo signal $s(t)$ and additive zero mean white Gaussian noise random process $n(t)$, with variance σ^2. The input noise is assumed to be spatially incoherent and uncorrelated with the signal.

The output of the bandpass intermediate frequency (IF) filter is the signal $v(t)$, which can be written as a bandpass random process. That is,

$$\begin{aligned} v(t) &= v_I(t)\cos\omega_0 t + v_Q(t)\sin\omega_0 t = r(t)\cos(\omega_0 t - \Phi(t)) \\ v_I(t) &= r(t)\cos\Phi(t) \\ v_Q(t) &= r(t)\sin\Phi(t) \end{aligned} \tag{7.1a}$$

$$\begin{aligned} r(t) &= \sqrt{[v_I(t)]^2 + [v_Q(t)]^2} \\ \Phi(t) &= \left[\tan\left(\frac{v_Q(t)}{v_I(t)}\right)\right]^{-1} \end{aligned} \tag{7.1b}$$

where $\omega_0 = 2\pi f_0$ is the radar operating frequency, $r(t)$ is the envelope of $v(t)$, the phase is $\Phi(t) = \text{atan}(v_Q/v_I)$, and the subscripts I, and Q, respectively, refer to the in-phase and quadrature components.

A target is detected when $r(t)$ exceeds the threshold value v_T, where the decision hypotheses are

$$\begin{aligned} s(t) + n(t) > v_T &\Rightarrow Detection \\ n(t) > v_T &\Rightarrow False\ \ alarm \end{aligned}$$

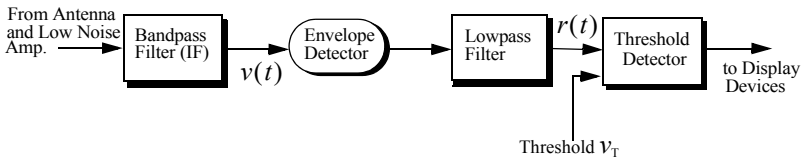

Figure 7.1. Simplified block diagram of an envelope detector and threshold receiver.

The case when the noise subtracts from the signal (while a target is present) to make $r(t)$ smaller than the threshold is called a miss. Radar designers seek to maximize the probability of detection for a given probability of false alarm.

The IF filter output is a complex random variable that is composed of either noise alone or noise plus target return signal (sine wave of amplitude A). The quadrature components corresponding to the case of noise alone are

$$\begin{aligned} v_I(t) &= n_I(t) \\ v_Q(t) &= n_Q(t) \end{aligned} \tag{7.2}$$

and for the second case,

$$\begin{aligned} v_I(t) = A + n_I(t) = r(t)\cos\Phi(t) \Rightarrow n_I(t) &= r(t)\cos\Phi(t) - A \\ v_Q(t) = n_Q(t) &= r(t)\sin\Phi(t) \end{aligned} \tag{7.3}$$

where the noise quadrature components $n_I(t)$ and $n_Q(t)$ are uncorrelated zero mean lowpass Gaussian noise with equal variances, σ^2. The joint Probability Density Function (*pdf*) of the two random variables $n_I;n_Q$ is

$$\begin{aligned} f_{n_I n_Q}(n_I, n_Q) &= \frac{1}{2\pi\sigma^2}\exp\left(-\frac{n_I^2 + n_Q^2}{2\sigma^2}\right) \\ &= \frac{1}{2\pi\sigma^2}\exp\left(-\frac{(r\cos\varphi - A)^2 + (r\sin\varphi)^2}{2\sigma^2}\right) \end{aligned} \tag{7.4}$$

The *pdfs* of the random variables $r(t)$ and $\Phi(t)$, respectively, represent the modulus and phase of $v(t)$. The joint *pdf* for the two random variables $r(t);\Phi(t)$ are derived using a similar approach to that developed in Chapter 3. More precisely,

$$f_{R\Phi}(r, \varphi) = f_{n_I n_Q}(n_I, n_Q)|\boldsymbol{J}| \tag{7.5}$$

where $\boldsymbol{J}$ is a matrix of derivatives defined by

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial n_I}{\partial r} & \dfrac{\partial n_I}{\partial \varphi} \\ \dfrac{\partial n_Q}{\partial r} & \dfrac{\partial n_Q}{\partial \varphi} \end{bmatrix} = \begin{bmatrix} \cos\varphi & -r\sin\varphi \\ \sin\varphi & r\cos\varphi \end{bmatrix} \tag{7.6}$$

The determinant of the matrix of derivatives is called the Jacobian, and in this case it is equal to

$$|\mathbf{J}| = r(t) \tag{7.7}$$

Substituting Eq. (7.4) and Eq. (7.7) into Eq. (7.5) and collecting terms yield

$$f_{R\Phi}(r, \varphi) = \frac{r}{2\pi\sigma^2}\exp\left(-\frac{r^2 + A^2}{2\sigma^2}\right)\exp\left(\frac{rA\cos\varphi}{\sigma^2}\right) \tag{7.8}$$

The *pdf* for $r(t)$ alone is obtained by integrating Eq. (7.8) over φ

$$f_R(r) = \int_0^{2\pi} f_{R\Phi}(r, \varphi)d\varphi = \frac{r}{\sigma^2}\exp\left(-\frac{r^2 + A^2}{2\sigma^2}\right)\frac{1}{2\pi}\int_0^{2\pi}\exp\left(\frac{rA\cos\varphi}{\sigma^2}\right)d\varphi \tag{7.9}$$

where the integral inside Eq. (7.9) is known as the modified Bessel function of zero order,

$$I_0(\beta) = \frac{1}{2\pi}\int_0^{2\pi} e^{\beta\cos\theta}\, d\theta \tag{7.10}$$

Thus,

$$f_R(r) = \frac{r}{\sigma^2}I_0\left(\frac{rA}{\sigma^2}\right)\exp\left(-\frac{r^2 + A^2}{2\sigma^2}\right) \tag{7.11}$$

which is the Rician probability density function. The case when $A/\sigma^2 = 0$ (noise alone) was analyzed in Chapter 3 and the resulting *pdf* is a Rayleigh probability density function

$$f_R(r) = \frac{r}{\sigma^2}\exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{7.12}$$

When (A/σ^2) is very large, Eq. (7.11) becomes a Gaussian probability density function of mean A and variance σ^2:

$$f_R(r) \approx \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(r-A)^2}{2\sigma^2}\right) \tag{7.13}$$

Figure 7.2 shows plots for the Rayleigh and Gaussian densities.

The density function for the random variable Φ is obtained from

$$f_\Phi(\varphi) = \int_0^r f_{R\Phi}(r, \varphi)\ dr \tag{7.14}$$

While the detailed derivation is left as an exercise, the result of Eq. (7.14) is

$$f_\Phi(\varphi) = \frac{1}{2\pi} \exp\left(\frac{-A^2}{2\sigma^2}\right) + \frac{A\cos\varphi}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(A\sin\varphi)^2}{2\sigma^2}\right) F\left(\frac{A\cos\varphi}{\sigma}\right) \tag{7.15}$$

where

$$F(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}}\ e^{-\zeta^2/2}\ d\xi \tag{7.16}$$

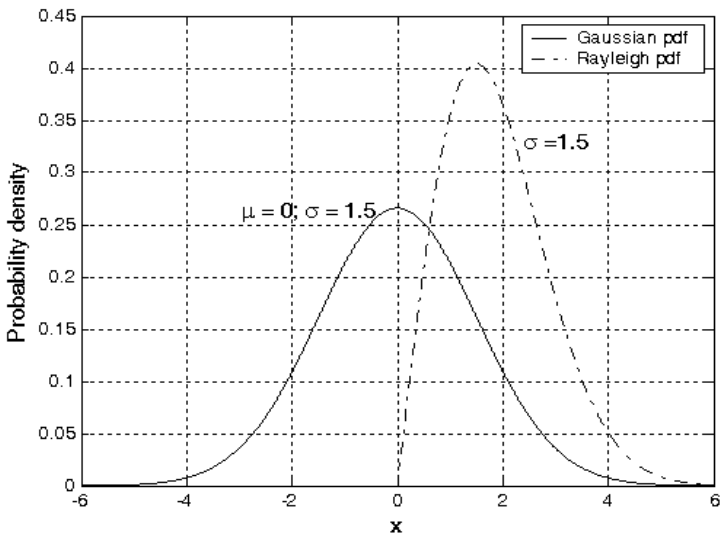

Figure 7.2. Gaussian and Rayleigh probability densities.

The function $F(x)$ can be found tabulated in most mathematical formula reference books. Note that for the case of noise alone ($A = 0$), Eq. (7.15) collapses to a uniform *pdf* over the interval $\{0, 2\pi\}$. One excellent approximation for the function $F(x)$ is

$$F(x) = 1 - \left(\frac{1}{0.661x + 0.339\sqrt{x^2 + 5.51}}\right)\frac{1}{\sqrt{2\pi}}e^{-x^2/2} \qquad x \geq 0 \tag{7.17}$$

and for negative values of x

$$F(-x) = 1 - F(x) \tag{7.18}$$

7.2. Probability of False Alarm

The probability of false alarm P_{fa} is defined as the probability that a sample r of the signal $r(t)$ will exceed the threshold voltage v_T when noise alone is present in the radar:

$$P_{fa} = \int_{v_T}^{\infty} \frac{r}{\sigma^2}\exp\left(-\frac{r^2}{2\sigma^2}\right) dr = \exp\left(\frac{-v_T^2}{2\sigma^2}\right) \tag{7.19}$$

$$v_T = \sqrt{2\sigma^2 \ln\left(\frac{1}{P_{fa}}\right)} \tag{7.20}$$

Figure 7.3 shows a plot of the normalized threshold versus the probability of false alarm. It is evident from this figure that P_{fa} is very sensitive to small changes in the threshold value. The false alarm time T_{fa} is related to the probability of false alarm by

$$T_{fa} = t_{int}/P_{fa} \tag{7.21}$$

where t_{int} represents the radar integration time, or the average time that the output of the envelope detector will pass the threshold voltage. Since the radar operating bandwidth B is the inverse of t_{int}, by substituting Eq. (7.19) into Eq. (7.20), we can write T_{fa} as

$$T_{fa} = \frac{1}{B}\exp\left(\frac{v_T^2}{2\sigma^2}\right) \tag{7.22}$$

Minimizing T_{fa} means increasing the threshold value, and as a result the radar maximum detection range is decreased. The choice of an acceptable value for T_{fa} becomes a compromise depending on the radar mode of operation.

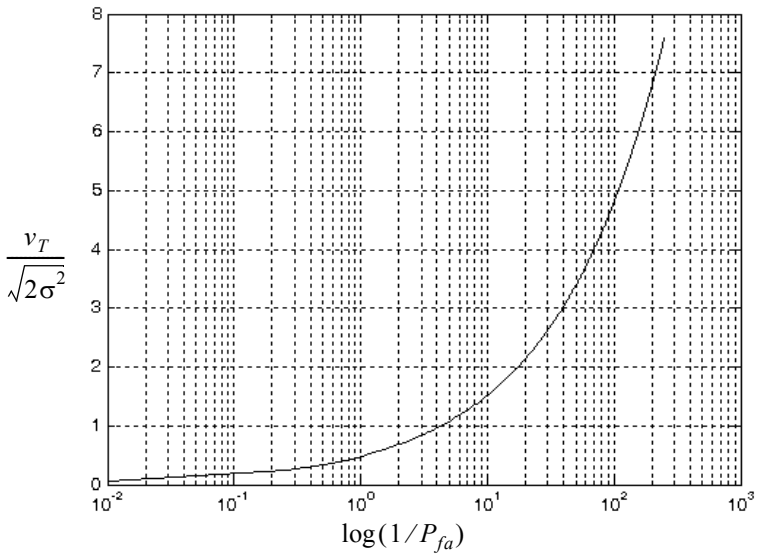

Figure 7.3. Normalized detection threshold versus probability of false alarm.

The false alarm number is defined as

$$n_{fa} = \frac{-\ln(2)}{\ln(1-P_{fa})} \approx \frac{\ln(2)}{P_{fa}} \tag{7.23}$$

Other slightly different definitions for the false alarm number exist in the literature, causing a source of confusion for many non-expert readers. Other than the definition in Eq. (7.23), the most commonly used definition for the false alarm number is the one introduced by Marcum (1960). Marcum defines the false alarm number as the reciprocal of P_{fa}. In this text, the definition given in Eq. (7.23) is always assumed. Hence, a clear distinction is made between Marcum's definition of the false alarm number and the definition in Eq. (7.23).

7.3. Probability of Detection

The probability of detection P_D is the probability that a sample r of $r(t)$ will exceed the threshold voltage in the case of noise plus signal,

$$P_D = \int_{v_T}^{\infty} \frac{r}{\sigma^2} I_0\left(\frac{rA}{\sigma^2}\right) \exp\left(-\frac{r^2+A^2}{2\sigma^2}\right) dr \tag{7.24}$$

If we assume that the radar signal is a sine waveform with amplitude A, then its power is $A^2/2$. Now, by using $SNR = A^2/2\sigma^2$ (single-pulse SNR) and $(v_T^2/2\sigma^2) = \ln(1/P_{fa})$, then Eq. (7.24) can be rewritten as

$$P_D = \int_{\sqrt{2\sigma^2 \ln(1/p_{fa})}}^{\infty} \frac{r}{\sigma^2} I_0\left(\frac{rA}{\sigma^2}\right) \exp\left(-\frac{r^2 + A^2}{2\sigma^2}\right) dr = Q\left[\sqrt{\frac{A^2}{\sigma^2}}, \sqrt{2\ln\left(\frac{1}{P_{fa}}\right)}\right] \quad \textbf{(7.25)}$$

$$Q[\alpha, \beta] = \int_{\beta}^{\infty} \zeta I_0(\alpha\zeta) e^{-(\zeta^2 + \alpha^2)/2} \, d\zeta \quad \textbf{(7.26)}$$

Q is called Marcum's Q-function. When P_{fa} is small and P_D is relatively large so that the threshold is also large, Eq. (7.25) can be approximated by

$$P_D \approx F\left(\frac{A}{\psi} - \sqrt{2\ln\left(\frac{1}{P_{fa}}\right)}\right) \quad \textbf{(7.27)}$$

where $F(x)$ is given by Eq. (7.16). Many approximations for computing Eq. (7.25) can be found throughout the literature. One very accurate approximation presented by North (1963) is given by

$$P_D \approx 0.5 \times erfc(\sqrt{-\ln P_{fa}} - \sqrt{SNR + 0.5}) \quad \textbf{(7.28)}$$

where the complementary error function is

$$erfc(z) = 1 - \frac{2}{\sqrt{\pi}} \int_0^z e^{-v^2} dv \quad \textbf{(7.29)}$$

The integral given in Eq. (7.25) is complicated and can be computed using numerical integration techniques. Parl[1] developed an excellent algorithm to numerically compute this integral. It is summarized as follows:

$$Q[a, b] = \begin{cases} \frac{\alpha_n}{2\beta_n} \exp\left(\frac{(a-b)^2}{2}\right) & a < b \\ 1 - \left(\frac{\alpha_n}{2\beta_n} \exp\left(\frac{(a-b)^2}{2}\right)\right) & a \geq b \end{cases} \quad \textbf{(7.30)}$$

1. Parl, S., A New Method of Calculating the Generalized Q Function, *IEEE Trans. Information Theory,* Vol. IT-26, January 1980, pp. 121-124.

$$\alpha_n = d_n + \frac{2n}{ab}\alpha_{n-1} + \alpha_{n-2} \tag{7.31}$$

$$\beta_n = 1 + \frac{2n}{ab}\beta_{n-1} + \beta_{n-2} \tag{7.32}$$

$$d_{n+1} = d_n d_1 \tag{7.33}$$

$$\alpha_0 = \begin{Bmatrix} 1 & a < b \\ 0 & a \geq b \end{Bmatrix} \tag{7.34}$$

$$d_1 = \begin{Bmatrix} a/b & a < b \\ b/a & a \geq b \end{Bmatrix} \tag{7.35}$$

$\alpha_{-1} = 0.0$, $\beta_0 = 0.5$, and $\beta_{-1} = 0$. The recursive Eq. (7.30) through Eq. (7.33) are computed continuously until $\beta_n > 10^p$ for values of $p \geq 3$. The accuracy of the algorithm is enhanced as the value of p is increased. The MATLAB function *"marcumsq.m"* implements Parl's algorithm to calculate the probability of detection defined in Eq. (7.24). The syntax is as follows:

Pd = marcumsq(alpha, beta)

where *alpha* and *beta* are from Eq. (7.26). Figure 7.4 shows plots of the probability of detection, P_D, versus the single pulse SNR, with the P_{fa} as a parameter using this function. The following MATLAB program can be used to reproduce Fig. 7.4. It uses the function *"marcumsq.m."*

```
% This program is used to produce Fig. 7.4
close all; clear all;
for nfa = 2:2:12
  b = sqrt(-2.0 * log(10^(-nfa)));
  index = 0;
  hold on
  for snr = 0:.1:18
    index = index +1;
    a = sqrt(2.0 * 10^(.1*snr));
    pro(index) = marcumsq(a,b);
  end
  x = 0:.1:18;
  set(gca,'ytick',[.1 .2 .3 .4 .5 .6  .7 .75 .8 .85 .9 .95 .9999])
  set(gca,'xtick',[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18])
   loglog(x, pro,'k');
end
hold off
xlabel ('Single pulse SNR in dB'); ylabel ('Probability of detection')
grid
```

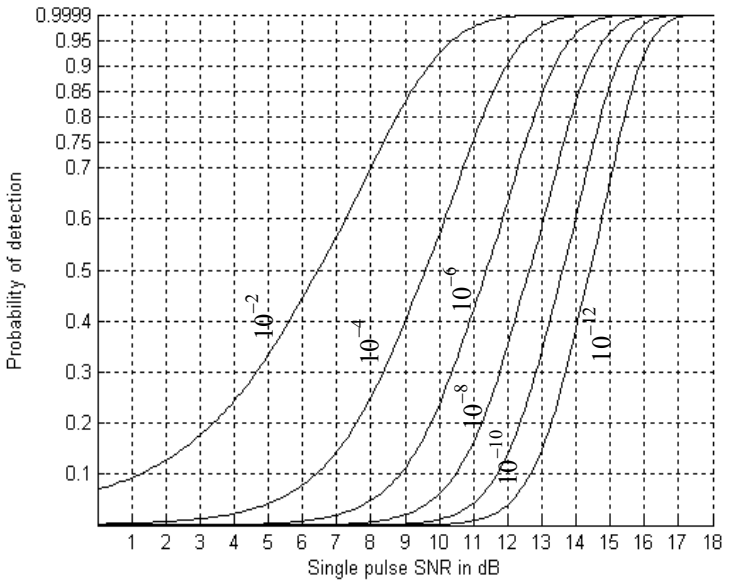

Figure 7.4. Probability of detection versus single pulse SNR, for several values of P_{fa}.

7.4. Pulse Integration

When a target is located within the radar beam during a single scan, it may reflect several pulses. By adding the returns from all pulses returned by a given target during a single scan, the radar sensitivity (SNR) can be increased. The number of returned pulses depends on the antenna scan rate and the radar PRF. More precisely, the number of pulses returned from a given target is given by

$$n_P = \frac{\theta_a T_{sc} f_r}{2\pi} \tag{7.36}$$

where θ_a is the azimuth antenna beamwidth, T_{sc} is the scan time, and f_r is the radar PRF. The number of reflected pulses may also be expressed as

$$n_P = \frac{\theta_a f_r}{\dot{\theta}_{scan}} \tag{7.37}$$

where $\dot{\theta}_{scan}$ is the antenna scan rate in degrees per second. Note that when using Eq. (7.36), θ_a is expressed in radians, while when using Eq. (7.37), it is expressed in degrees. As an example, consider a radar with an azimuth antenna

beamwidth $\theta_a = 3°$, antenna scan rate $\dot{\theta}_{scan} = 45°/\text{sec}$ (antenna scan time, $T_{sc} = 8\,\text{sec}$), and a PRF $f_r = 300Hz$. Using either Eq. (7.36) or Eq. (7.37) yields $n_P = 20$ pulses.

The process of adding radar returns from many pulses is called radar pulse integration. Pulse integration can be performed on the quadrature components prior to the envelope detector. This is called coherent integration or predetection integration. Coherent integration preserves the phase relationship between the received pulses. Thus a buildup in the signal amplitude is achieved. Alternatively, pulse integration performed after the envelope detector (where the phase relation is destroyed) is called noncoherent or postdetection integration.

Radar designers should exercise caution when utilizing pulse integration for the following reasons. First, during a scan a given target will not always be located at the center of the radar beam (i.e., have maximum gain). In fact, during a scan a given target will first enter the antenna beam at the 3-dB point, reach maximum gain, and finally leave the beam at the 3-dB point again. Thus, the returns do not have the same amplitude even though the target RCS may be constant and all other factors that may introduce signal loss remain the same.

Other factors that may introduce further variation to the amplitude of the returned pulses include target RCS and propagation path fluctuations. Additionally, when the radar employs a very fast scan rate, an additional loss term is introduced due to the motion of the beam between transmission and reception. This is referred to as scan loss. A distinction should be made between scan loss due to a rotating antenna (which is described here) and the term scan loss that is normally associated with phased array antennas (which takes on a different meaning in that context).

Finally, since coherent integration utilizes the phase information from all integrated pulses, it is critical that any phase variation between all integrated pulses be known with a great level of confidence. Consequently, target dynamics (such as target range, range rate, tumble rate, RCS fluctuation) must be estimated or computed accurately so that coherent integration can be meaningful. In fact, if a radar coherently integrates pulses from targets without proper knowledge of the target dynamics, it suffers a loss in SNR rather than the expected SNR buildup. Knowledge of target dynamics is not as critical when employing noncoherent integration; nonetheless, target range rate must be estimated so that only the returns from a given target within a specific range bin are integrated. In other words, one must avoid range walk (i.e., having a target cross between adjacent range bins during a single scan).

A comprehensive analysis of pulse integration should take into account issues such as the probability of detection P_D, probability of false alarm P_{fa}, the target statistical fluctuation model, and the noise or interference of statistical models. This is the subject of the rest of this chapter.

7.4.1. Coherent Integration

In coherent integration, when a perfect integrator is used (100% efficiency), to integrate n_P pulses, the SNR is improved by the same factor. Otherwise, integration loss occurs, which is always the case for noncoherent integration. Coherent integration loss occurs when the integration process is not optimum. This could be due to target fluctuation, instability in the radar local oscillator, or propagation path changes.

Denote the single pulse SNR required to produce a given probability of detection as $(SNR)_1$. The SNR resulting from coherently integrating n_P pulses is then given by

$$(SNR)_{CI} = n_P(SNR)_1 \tag{7.38}$$

Coherent integration cannot be applied over a large number of pulses, particularly if the target RCS is varying rapidly. If the target radial velocity is known and no acceleration is assumed, the maximum coherent integration time is limited to

$$t_{CI} = \sqrt{\lambda/2a_r} \tag{7.39}$$

where λ is the radar wavelength and a_r is the target radial acceleration. Coherent integration time can be extended if the target radial acceleration can be compensated for by the radar.

In order to demonstrate the improvement in the SNR using coherent integration, consider the case where the radar return signal contains both signal plus additive noise. The mth pulse is

$$y_m(t) = s(t) + n_m(t) \tag{7.40}$$

where $s(t)$ is the radar signal return of interest and $n_m(t)$ is white uncorrelated additive noise signal with variance σ^2. Coherent integration of n_P pulses yields

$$z(t) = \frac{1}{n_P}\sum_{m=1}^{n_P} y_m(t) = \sum_{m=1}^{n_P}\frac{1}{n_P}[s(t) + n_m(t)] = s(t) + \sum_{m=1}^{n_P}\frac{1}{n_P}n_m(t) \tag{7.41}$$

The total noise power in $z(t)$ is equal to the variance. More precisely,

$$\sigma^2_{n_P} = E\left[\left(\sum_{m=1}^{n_P}\frac{1}{n_P}n_m(t)\right)\left(\sum_{l=1}^{n_P}\frac{1}{n_P}n_l(t)\right)^*\right] \tag{7.42}$$

where E is the expected value operator. It follows that

$$\sigma_{n_P}^2 = \frac{1}{n_P^2} \sum_{m,\, l\,=\,1}^{n_P} E[n_m(t) n_l{}^*(t)] = \frac{1}{n_P^2} \sum_{m,\, l\,=\,1}^{n_P} \sigma_{ny}^2 \delta_{ml} = \frac{1}{n_P} \sigma_{ny}^2 \qquad \textbf{(7.43)}$$

where σ_{ny}^2 is the single pulse noise power and δ_{ml} is equal to zero for $m \neq l$ and unity for $m = l$. Observation of Eqs. (7.41) and (7.42) shows that the desired signal power after coherent integration is unchanged, while the noise power is reduced by the factor $1/n_P$. Thus, the SNR after coherent integration is improved by n_P.

7.4.2. Noncoherent Integration

When the phase of the integrated pulses is not known so that coherent integration is no longer possible, another form of pulse integration is done. In this case, pulse integration is performed by adding (integrating) the individual pulses' envelopes or the square of their envelopes. Thus, the term noncoherent integration is adopted. A block diagram of radar receiver utilizing noncoherent integration is illustrated in Fig. 7.5.

The performance difference (measured in SNR) between the linear envelope detector and the quadratic (square law) detector is practically negligible. Robertson (1967) showed that this difference is typically less than $0.2dB$; he showed that the performance difference is higher than $0.2dB$ only for cases where $n_P > 100$ and $P_D < 0.01$. Both of these conditions are of no practical significance in radar applications. It is much easier to analyze and implement the square law detector in real hardware than is the case for the envelope detector. Therefore, most authors make no distinction between the type of detector used when referring to noncoherent integration, and the square law detector is almost always assumed. The analysis presented in this book will always assume, unless indicated otherwise, noncoherent integration using the square law detector.

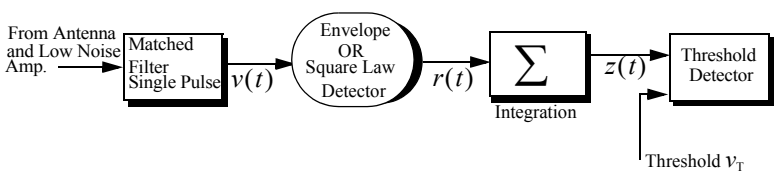

Figure 7.5. Simplified block diagram of a radar detector when noncoherent integration is used.

7.4.3. Improvement Factor and Integration Loss

Noncoherent integration is less efficient than coherent integration. Actually, the noncoherent integration gain is always smaller than the number of noncoherently integrated pulses. This loss in integration is referred to as postdetection or square-law detector loss.

Define $(SNR)_{NCI}$ as the SNR required to achieve a specific P_D given a particular P_{fa} when n_P pulses are integrated noncoherently. Also denote the single pulse SNR as $(SNR)_1$. It follows that

$$(SNR)_{NCI} = (SNR)_1 \times I(n_P) \tag{7.44}$$

where $I(n_P)$ is called the integration improvement factor. An empirically derived expression for the improvement factor that is accurate within $0.8dB$ is reported in Peebles (1998) as

$$[I(n_P)]_{dB} = 6.79(1 + 0.253P_D)\left(1 + \frac{\log(1/P_{fa})}{46.6}\right)\log(n_P) \tag{7.45}$$
$$(1 - 0.140\log(n_P) + 0.018310(\log n_P)^2)$$

The top part of Fig. 7.6 shows plots of the integration improvement factor as a function of the number of integrated pulses with P_D and P_{fa} as parameters using Eq. (7.45). The integration loss in dB is defined as

$$[L_{NCI}]_{dB} = 10\log n_P - [I(n_P)]_{dB} \tag{7.46}$$

The lower part of Fig. 7.6 shows plots of the corresponding integration loss versus n_P with P_D and P_{fa} as parameters. This figure can be reproduced using the following MATLAB code which uses MATLAB function *"improv_fac.m."*

```
% This program is used to produce Fig. 7.6
% It uses the function "improv_fac.m".
clear all;
close all;
Pfa = [1e-2, 1e-6, 1e-8, 1e-10];
Pd = [.5 .8 .95 .99];
np = linspace(1,1000,10000);
I(1,:) = improv_fac (np, Pfa(1), Pd(1));
I(2,:) = improv_fac (np, Pfa(2), Pd(2));
I(3,:) = improv_fac (np, Pfa(3), Pd(3));
I(4,:) = improv_fac (np, Pfa(4), Pd(4));
index = [1 2 3 4];
L(1,:) = 10.*log10(np) - I(1,:);
L(2,:) = 10.*log10(np) - I(2,:);
L(3,:) = 10.*log10(np) - I(3,:);
```

```
L(4,:) = 10.*log10(np) - I(4,:);
subplot(2,1,2);
semilogx (np, L(1,:), 'k:', np, L(2,:), 'k-.', ...
np, L(3,:), 'k-.', np, L(4,:), 'k')
xlabel ('Number of pulses');
ylabel ('Integration loss in dB')
axis tight; grid
subplot(2,1,1);
semilogx (np, I(1,:), 'k:', np, I(2,:), 'k-.', np, ...
 I(3,:), 'k--', np, I(4,:), 'k')
xlabel ('Number of pulses');
ylabel ('Improvement factor in dB')
legend ('pd=.5, Pfa=1e-2','pd=.8, Pfa=1e-6','pd=.95, ...
Pfa=1e-8','pd=.99, Pfa=1e-10');
grid;
axis tight
```

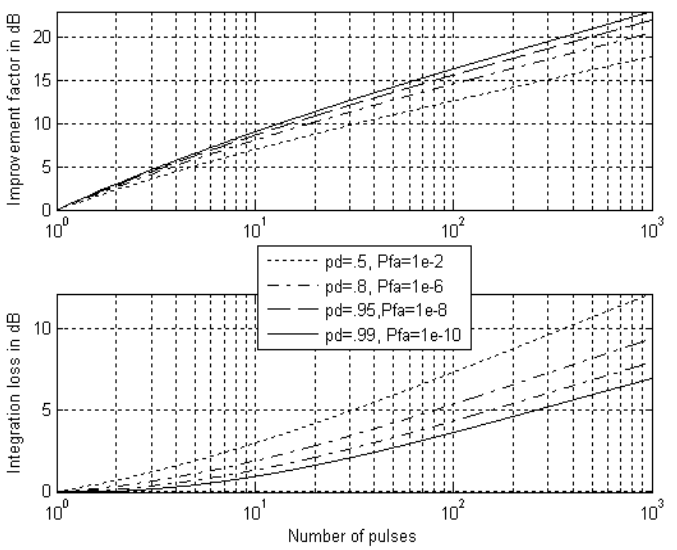

Figure 7.6. Typical plots for the improvement factor and integration loss versus number of noncoherently integrated pulses.

7.5. Target Fluctuation

Target detection utilizing the square law detector was first analyzed by Marcum[1], where he assumed a constant RCS (nonfluctuating target). This work was extended by Swerling[2] to four distinct cases of target RCS fluctuation. These cases have come to be known as Swerling models. They are Swerling I, Swerling II, Swerling III, and Swerling IV. The constant RCS case analyzed by Marcum is widely known as Swerling 0 or equivalently Swerling V. Target fluctuation introduces an additional loss factor in the SNR as compared to the case where fluctuation is not present given the same P_D and P_{fa}.

Swerling I targets have constant amplitude over one antenna scan or observation interval; however, a Swerling I target amplitude varies independently from scan to scan according to a chi-square probability density function with two degrees of freedom. The amplitude of Swerling II targets fluctuates independently from pulse to pulse according to a chi-square probability density function with two degrees of freedom. Target fluctuation associated with a Swerling III model is from scan to scan according to a chi-square probability density function with four degrees of freedom. Finally, the fluctuation of Swerling IV targets is from pulse to pulse according to a chi-square probability density function with four degrees of freedom.

Swerling showed that the statistics associated with Swerling I and II models apply to targets consisting of many small scatterers of comparable RCS values, while the statistics associated with Swerling III and IV models apply to targets consisting of one large RCS scatterer and many small equal RCS scatterers. Noncoherent integration can be applied to all four Swerling models; however, coherent integration cannot be used when the target fluctuation is either Swerling II or Swerling IV. This is because the target amplitude decorrelates from pulse to pulse (fast fluctuation) for Swerling II and IV models, and thus phase coherency cannot be maintained.

The chi-square *pdf* with $2N$ degrees of freedom can be written as

$$f_X(x) = \frac{N}{(N-1)!\ \sqrt{\sigma_x^2}} \left(\frac{Nx}{\sigma_x}\right)^{N-1} \exp\left(-\frac{Nx}{\sigma_x}\right) \qquad \textbf{(7.47)}$$

where σ_x is the standard deviation for the RCS value. Using this equation, the *pdf* associated with Swerling I and II targets can be obtained by letting $N = 1$, which yields a Rayleigh *pdf*. More precisely,

1. Marcum, J. I., A Statistical Theory of Target Detection by Pulsed Radar, *IRE Transactions on Information Theory,* Vol IT-6, pp. 59-267, April 1960.
2. Swerling, P., Probability of Detection for Fluctuating Targets, *IRE Transactions on Information Theory*, Vol IT-6, pp. 269-308, April 1960.

$$f_X(x) = \frac{1}{\sigma_x}\exp\left(-\frac{x}{\sigma_x}\right) \qquad x \geq 0 \tag{7.48}$$

Letting $N = 2$ yields the *pdf* for Swerling III and IV type targets,

$$f_X(x) = \frac{4x}{\sigma_x^2}\exp\left(-\frac{2x}{\sigma_x}\right) \qquad x \geq 0 \tag{7.49}$$

7.6. Probability of False Alarm Formulation for a Square Law Detector

Computation of the general formula for the probability of false alarm P_{fa} and subsequently the rest of square law detection theory requires knowledge and good understating of the incomplete Gamma function. Hence, those readers who are not familiar with this function are advised to read Appendix 7.A before proceeding with the rest of this chapter.

DiFranco and Rubin[1] derived a general form relating the threshold and P_{fa} for any number of pulses when noncoherent integration is used. The square law detector under consideration is shown in Fig. 7.7. There are $n_P \geq 2$ pulses integrated noncoherently and the noise power (variance) is σ^2.

The complex envelope in terms of the quadrature components is given by

$$\tilde{r}(t) = r_I(t) + jr_Q(t) \tag{7.50}$$

thus, the square of the complex envelope is

$$|\tilde{r}(t)|^2 = r_I^2(t) + r_Q^2(t) \tag{7.51}$$

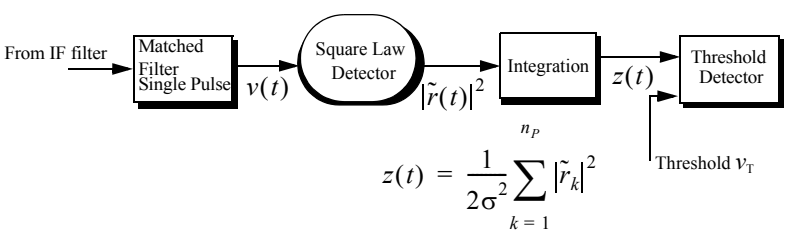

Figure 7.7. Square law detector.

1. DiFranco, J. V. and Rubin, W. L., *Radar Detection*, Artech House, Norwood, MA 1980.

The samples $|\tilde{r}_k|^2$ are computed from the samples of $\tilde{r}(t)$ evaluated at $t = t_k;\ k = 1, 2, \ldots, n_P$. It follows that

$$Z = \frac{1}{2\sigma^2}\sum_{k=1}^{n_P}[r_I^2(t_k) + r_Q^2(t_k)] \tag{7.52}$$

The random variable Z is the sum of $2n_P$ squares of random variables, each of which is a Gaussian random variable with variance σ^2. Thus, using the analysis developed in Chapter 3, the *pdf* for the random variable Z is given by

$$f_Z(z) = \left\{\begin{array}{ll} \dfrac{z^{n_P-1}e^{-z}}{\Gamma(n_P)} & z \geq 0 \\ 0 & z < 0 \end{array}\right\} \tag{7.53}$$

Consequently, the probability of false alarm given a threshold value v_Y is

$$P_{fa} = Prob\{Z \geq v_T\} = \int_{v_T}^{\infty} \frac{z^{n_P-1}e^{-z}}{\Gamma(n_P)}\, dz \tag{7.54}$$

and using analysis provided in Appendix 7.A yields

$$P_{fa} = 1 - \Gamma_I\left(\frac{v_T}{\sqrt{n_P}}, n_P - 1\right) \tag{7.55}$$

Using the algebraic expression for the incomplete Gamma function, Eq. (7.55) can be written as

$$P_{fa} = e^{-v_T}\sum_{k=0}^{n_P-1}\frac{v_T^k}{k!} = 1 - e^{-v_T}\sum_{k=n_P}^{\infty}\frac{v_T^k}{k!} \tag{7.56}$$

The threshold value v_T can then be approximated by the recursive formula used in the Newton-Raphson method. More precisely,

$$v_{T,m} = v_{T,m-1} - \frac{G(v_{T,m-1})}{G'(v_{T,m-1})} \quad ;\ m = 1, 2, 3, \ldots \tag{7.57}$$

The iteration is terminated when $|v_{T,m} - v_{T,m-1}| < v_{T,m-1}/10000.0$. The functions G and G' are

$$G(v_{T,m}) = (0.5)^{n_P/n_{fa}} - \Gamma_I(v_T, n_P) \tag{7.58}$$

$$G'(v_{T,m}) = -\frac{e^{-v_T} v_T^{n_P-1}}{(n_P-1)!} \tag{7.59}$$

The initial value for the recursion is

$$v_{T,0} = n_P - \sqrt{n_P} + 2.3 \sqrt{-\log P_{fa}} \left(\sqrt{-\log P_{fa}} + \sqrt{n_P} - 1\right) \tag{7.60}$$

Figure 7.8 shows plots of the threshold value versus the number of integrated pulses for several values of n_{fa}; remember that $P_{fa} \approx \ln(2)/n_{fa}$. This figure can be reproduced using the following MATLAB code which utilizes the MATLAB function *"threshold.m"*

```
% Use this program to reproduce Fig. 7.8 of text
clear all; close all;
for n= 1: 1:10000
  [pfa1 y1(n)] = threshold(1e4,n);
  [pfa2 y3(n)] = threshold(1e8,n);
  [pfa3 y4(n)] = threshold(1e12,n);
end
n =1:1:10000;
loglog(n,y1,'k',n,y3,'k--',n,y4,'k-.');
xlabel ('Number of pulses');
ylabel ('Threshold');
legend('nfa=1e4','nfa=1e8','nfa=1e12'); grid
```

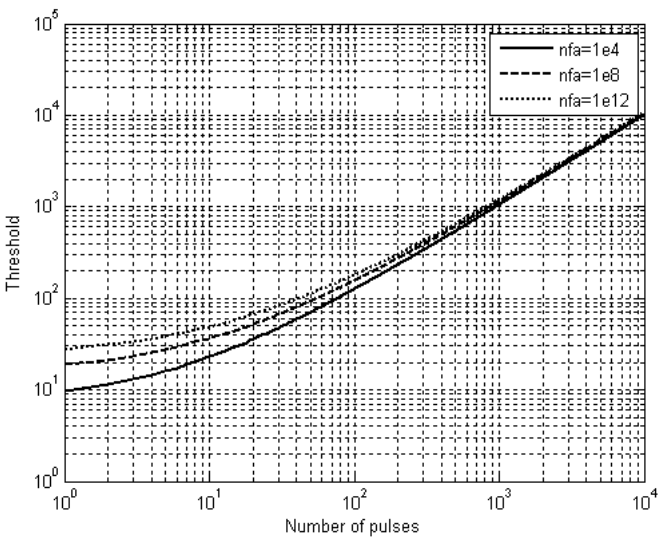

Figure 7.8. Threshold v_T versus n_p for several values of n_{fa}.

7.6.1. Square Law Detection

The *pdf* for the linear envelope $r(t)$ was derived earlier and it is given in Eq. (7.11). Define a new dimensionless variable y as

$$y_n = r_n/\sigma \tag{7.61}$$

and also define

$$\Re_p = A^2/\sigma^2 = 2SNR \tag{7.62}$$

σ^2 is the noise variance. It follows that the *pdf* for the new variable is

$$f_{Y_n}(y_n) = f_{R_n}(r_n)\left|\frac{dr_n}{dy_n}\right| = y_n \; I_0(y_n\sqrt{\Re_p}) \; \exp\left(\frac{-(y_n^2+\Re_p)}{2}\right) \tag{7.63}$$

The output of a square law detector for the nth pulse is proportional to the square of its input. Thus, it is convenient to define a new change variable,

$$z_n = \frac{1}{2}y_n^2 \tag{7.64}$$

The *pdf* for the variable at the output of the square law detector is given by

$$f_{Z_n}(x_n) = f(y_n)\left|\frac{dy_n}{dz_n}\right| = \exp\left(-\left(z_n+\frac{\Re_p}{2}\right)\right) I_0(\sqrt{2z_n\Re_p}) \tag{7.65}$$

Noncoherent integration of n_p pulses is implemented as

$$z = \sum_{n=1}^{n_P} \frac{1}{2}y_n^2 \tag{7.66}$$

Again, $n_P \geq 2$. Since the random variables y_n are independent, the *pdf* for the variable z is

$$f(z) = f((y_1) \otimes f(y_2) \otimes \ldots \otimes f(y_{n_p})) \tag{7.67}$$

The operator $\otimes$ symbolically indicates convolution. The characteristic functions for the individual *pdf*s can then be used to compute the joint *pdf* for Eq. (7.69). The result is

$$f_Z(z) = \left(\frac{2z}{n_P\Re_p}\right)^{(n_P-1)/2} \exp\left(-z-\frac{1}{2}n_P\Re_p\right) I_{n_P-1}(\sqrt{2n_Pz\Re_p}) \tag{7.68}$$

I_{n_P-1} is the modified Bessel function of order n_P-1. Substituting Eq. (7.62) into (7.68) yields

$$f_Z(z) = \left(\frac{z}{n_P SNR}\right)^{(n_P-1)/2} e^{(-z-n_P SNR)} I_{n_P-1}(2\sqrt{n_P z SNR}) \tag{7.69}$$

When target fluctuation is not present (i.e., Swerling 0), the probability of detection is obtained by integrating $f_Z(z)$ from the threshold value to infinity. The probability of false alarm is obtained by letting $\Re_p$ be zero and integrating the *pdf* from the threshold value to infinity. More specifically,

$$P_D\big|_{SNR} = \int_{v_T}^{\infty} \left(\frac{z}{n_P SNR}\right)^{(n_P-1)/2} e^{(-z-n_P SNR)} I_{n_P-1}(2\sqrt{n_P z SNR})dz \tag{7.70}$$

Which can be rewritten as

$$P_D\big|_{SNR} = e^{-n_P SNR}\left(\sum_{k=0}^{\infty}\frac{(n_P SNR)^k}{k!}\right)\left(\sum_{j=0}^{n_P-1+k}\frac{e^{-v_T}\,v_T^j}{j!}\right) \tag{7.71}$$

Alternatively, when target fluctuation is present, then the *pdf* is calculated using the conditional probability density function of Eq. (7.70) with respect to the SNR value of the target fluctuation type. In general, given a fluctuating target with SNR^F, where the superscript indicates fluctuation, the expression for the probability of detection is

$$P_D\big|_{SNR^F} = \int_0^{\infty} P_D\big|_{SNR}\, f_Z(z^F/SNR^F)dz = \tag{7.72}$$
$$\int_0^{\infty} P_D\big|_{SNR}\left(\frac{z^F}{n_P SNR^F}\right)^{(n_P-1)/2} e^{(-z^F-n_P SNR^F)} I_{n_P-1}(2\sqrt{n_P z^F SNR^F})dz$$

Remember that target fluctuation introduces an additional loss term in the SNR. It follows that for the same P_D given the same P_{fa} and the same n_P, $SNR^F > SNR$. One way to calculate this additional SNR is to first compute the required SNR given no fluctuation then add to it the amount of target fluctuation loss to get the required value for SNR^F. How to calculate this fluctuation loss will be addressed later on in this chapter. Meanwhile, hereon after, the superscript $\{\ ^F\}$ will be dropped and it will always be assumed.

7.7. Probability of Detection Calculation

Marcum defined the probability of false alarm for the case when $n_P > 1$ as

$$P_{fa} \approx \ln(2)(n_P/n_{fa}) \tag{7.73}$$

The single pulse probability of detection for nonfluctuating targets is given in Eq. (7.25). When $n_P > 1$, the probability of detection is computed using the Gram-Charlier series. In this case, the probability of detection is

$$P_D \cong \frac{erfc(V/\sqrt{2})}{2} - \frac{e^{-V^2/2}}{\sqrt{2\pi}}[C_3(V^2-1) + C_4 V(3-V^2) - C_6 V(V^4 - 10V^2 + 15)] \quad (7.74)$$

where the constants C_3, C_4, and C_6 are the Gram-Charlier series coefficients, and the variable V is

$$V = \frac{v_T - n_P(1+SNR)}{\varpi} \quad (7.75)$$

In general, values for C_3, C_4, C_6, and ϖ vary depending on the target fluctuation type.

7.7.1. Swerling 0 Target Detection

For Swerling 0 (Swerling V) target fluctuations, the probability of detection is calculated using Eq. (7.74). In this case, the Gram-Charlier series coefficients are

$$C_3 = -\frac{SNR + 1/3}{\sqrt{n_p}(2SNR+1)^{1.5}} \quad (7.76)$$

$$C_4 = \frac{SNR + 1/4}{n_p(2SNR+1)^2} \quad (7.77)$$

$$C_6 = C_3^2/2 \quad (7.78)$$

$$\varpi = \sqrt{n_p(2SNR+1)} \quad (7.79)$$

Figure 7.9 shows a plot for the probability of detection versus SNR for cases $n_p = 1, 10$. Note that it requires less SNR, with ten pulses integrated noncoherently, to achieve the same probability of detection as in the case of a single pulse. Hence, for any given P_D the SNR improvement can be read from the plot. Equivalently, using the function *"improv_fac.m"* leads to about the same result. For example, when $P_D = 0.8$, the function *"improv_fac.m"* gives an SNR improvement factor of $I(10) \approx 8.55dB$. Figure 7.9 shows that the ten pulse SNR is about $6.03dB$. Therefore, the single pulse SNR is about $14.5dB$, which can be read from the figure.

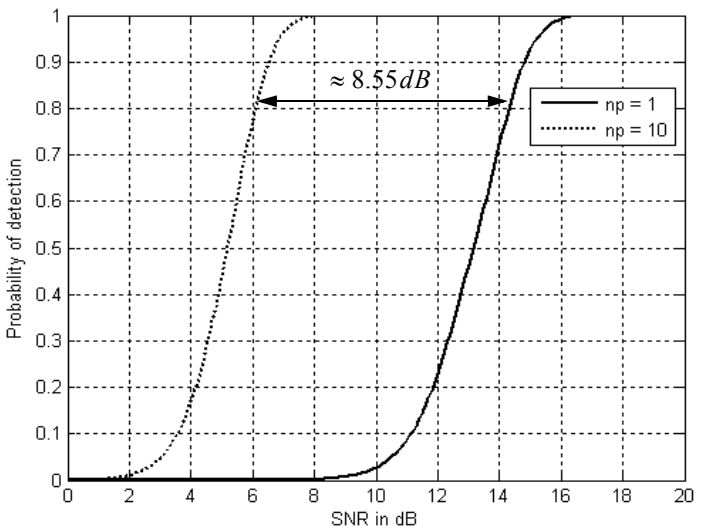

Figure 7.9. Probability of detection versus SNR, $P_{fa} = 10^{-9}$, and noncoherent integration; Swerling 0.

7.7.2. Detection of Swerling I Targets

The exact formula for the probability of detection for Swerling I type targets was derived by Swerling. It is

$$P_D = e^{-(v_T)/(1+SNR)} \quad ; \; n_P = 1 \tag{7.80}$$

$$P_D = 1 - \Gamma_I(v_T, n_P - 1) + \left(1 + \frac{1}{n_P SNR}\right)^{n_p - 1} \Gamma_I\left(\frac{v_T}{1 + \frac{1}{n_P SNR}}, n_P - 1\right) \times e^{-v_T/(1+n_P SNR)} \quad ; \; n_P > 1 \tag{7.81}$$

Figure 7.10 shows a plot of the probability of detection as a function of SNR for $n_p = 1$ and $P_{fa} = 10^{-9}$ for both Swerling I and V (Swerling 0) type fluctuations. Note that it requires more SNR, with fluctuation, to achieve the same P_D as in the case with no fluctuation. This figure can be reproduced using the following MATLAB code.

```
% Generate Figure 7.10
close all;
clear all;
pfa = 1e-9;
nfa = log(2) / pfa;
b = sqrt(-2.0 * log(pfa));
index = 0;
for snr = 0:.01:22
   index = index +1;
   a = sqrt(2.0 * 10^(.1*snr));
  swer0(index) = marcumsq(a,b);
   swer1(index) =  pd_swerling1 (nfa, 1, snr);
end
x = 0:.01:22;
%figure(10)
plot(x, swer0,'k',x,swer1,'k:');
axis([2 22 0 1])
xlabel ('SNR in dB')
ylabel ('Probability of detection')
legend('Swerling 0','Swerling I')
grid
```

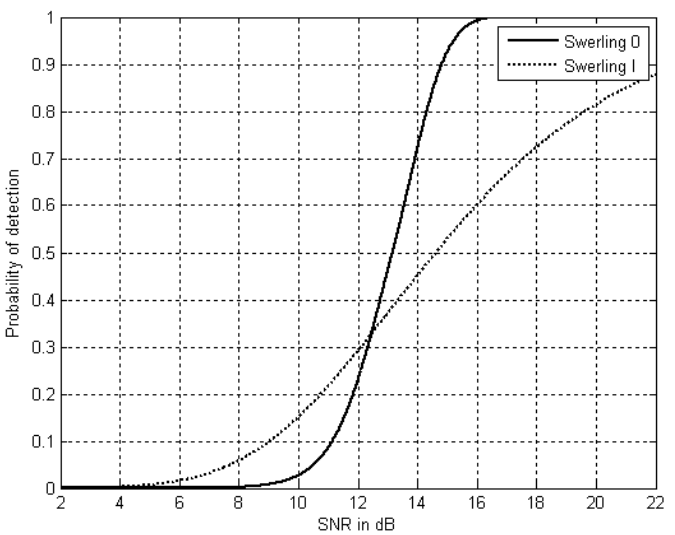

Figure 7.10. Probability of detection versus SNR, single pulse. $P_{fa} = 10^{-9}$.

Figure 7.11 is similar to Fig. 7.10 except in this case $P_{fa} = 10^{-6}$ and $n_P = 5$. This figure can be reproduced using the following MATLAB code

```
% Generate Figure 7.11
clear all
close all
pfa = 1e-6;
nfa = log(2) / pfa;
index = 0;
for snr = -10:.5:30
   index = index +1;
   prob1(index) =  pd_swerling1 (nfa, 5, snr);
   prob0(index) =  pd_swerling5 (nfa, 2, 5, snr);
  end
x = -10:.5:30;
plot(x, prob1,'k',x,prob0,'k:');
axis([-10 30 0 1])
xlabel ('SNR in dB')
ylabel ('Probability of detection')
legend('Swerling I','Swerling 0')
title('Pfa =1e-6;  n=5')
grid
```

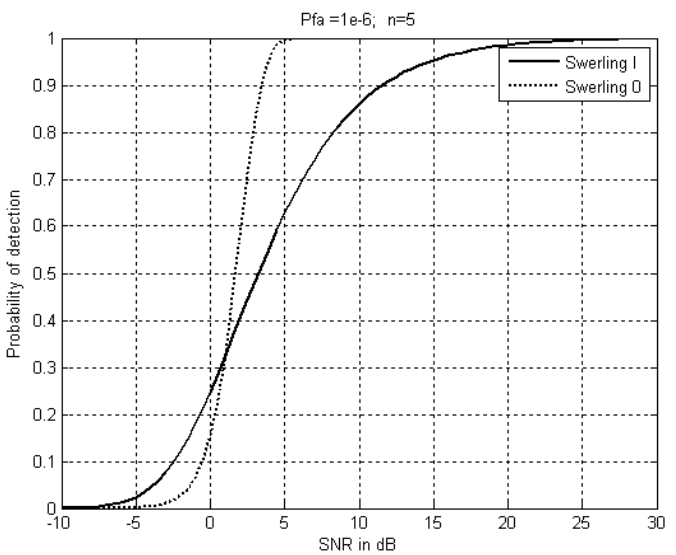

Figure 7.11. Probability of detection versus SNR. Swerling I and Swerling 0.

7.7.3. Detection of Swerling II Targets

In the case of Swerling II targets, the probability of detection is given by

$$P_D = 1 - \Gamma_I\left(\frac{v_T}{(1+SNR)}\ , n_p\right) \qquad ;\ n_P \le 50 \tag{7.82}$$

For the case when $n_P > 50$ the probability of detection is computed using the Gram-Charlier series. In this case,

$$C_3 = -\frac{1}{3\sqrt{n_p}} \qquad ,\ C_6 = \frac{C_3^2}{2} \tag{7.83}$$

$$C_4 = \frac{1}{4n_P} \tag{7.84}$$

$$\varpi = \sqrt{n_P}\ (1+SNR) \tag{7.85}$$

Figure 7.12a shows a plot of the probability of detection for Swerling 0, Swerling I, and Swerling II with $n_P = 5$, where $P_{fa} = 10^{-7}$. This figure can be reproduced using the following MATLAB code. Figure 7.12b is similar to Fig. 7.12a except in this case $n_P = 2$.

```
% Generate Figure 7.12
clc
clear all;
close all;
pfa = 1e-7;
nfa = log(2) / pfa;
index = 0;
for snr = -10:.5:30
  index = index +1;
  prob1(index) = pd_swerling1 (nfa, 5, snr); % Fig. 7.12a
  prob0(index) = pd_swerling5 (nfa, 2, 5, snr); % Fig. 7.12a
  prob2(index) = pd_swerling2 (nfa, 5, snr); % Fig. 7.12a
  % prob1(index) = pd_swerling1 (nfa, 2, snr); % Fig. 7.12b
  % prob0(index) = pd_swerling5 (nfa, 2, 2, snr); % Fig. 7.12b
  % prob2(index) = pd_swerling2 (nfa, 2, snr); % Fig. 7.12b
end
x = -10:.5:30;
plot(x, prob0,'k',x,prob1,'k:',x,prob2,'k--');
axis([-10 30 0 1])
xlabel ('SNR in dB')
ylabel ('Probability of detection')
legend('Swerling 0','Swerling I','Swerling II')
title('Pfa =1e-7; n=5')
grid
```

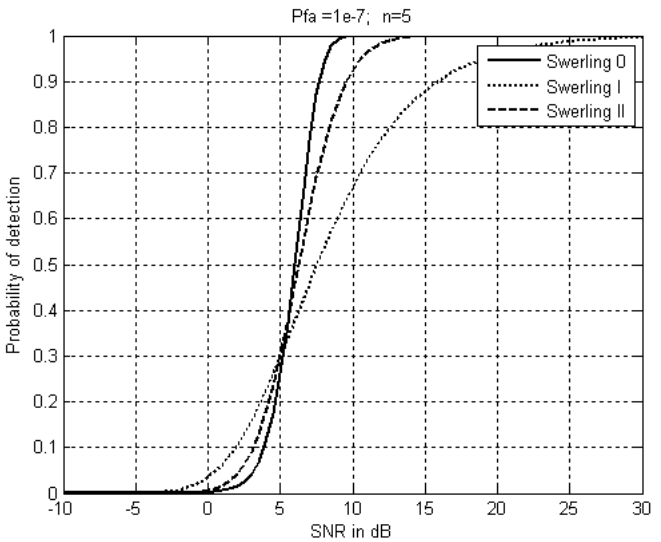

Figure 7.12a. Probability of detection versus SNR. Swerling II, Swerling I and Swerling 0.

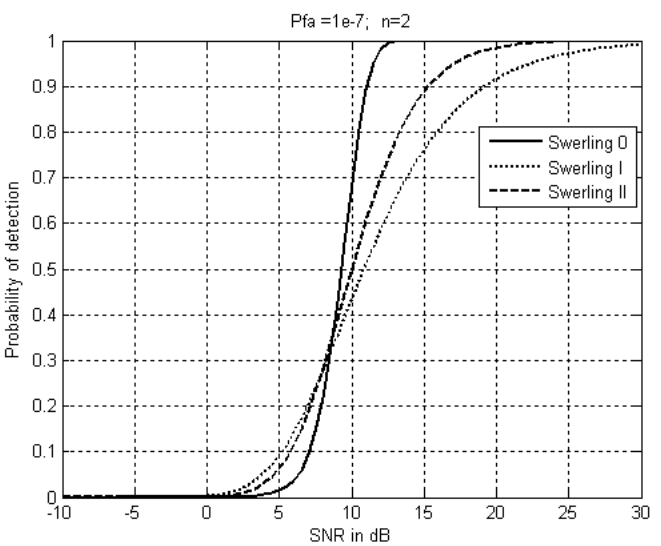

Figure 7.12b. Probability of detection versus SNR. Swerling II, Swerling I, and Swerling 0.

7.7.4. Detection of Swerling III Targets

The exact formulas, developed by Marcum, for the probability of detection for Swerling III type targets when $n_P = 1, 2$

$$P_D = \exp\left(\frac{-v_T}{1 + n_P SNR/2}\right)\left(1 + \frac{2}{n_P SNR}\right)^{n_P - 2} \times K_0$$
$$K_0 = 1 + \frac{v_T}{1 + n_P SNR/2} - \frac{2}{n_P SNR}(n_P - 2) \tag{7.86}$$

For $n_P > 2$ the expression is

$$P_D = \frac{v_T^{n_P - 1} e^{-V_T}}{(1 + n_P SNR/2)(n_P - 2)!} + 1 - \Gamma_I(v_T, n_P - 1) + K_0 \tag{7.87}$$
$$\times \Gamma_I\left(\frac{v_T}{1 + 2/n_p SNR}, n_p - 1\right)$$

Figure 7.13a shows a plot of the probability of detection as a function of SNR for $n_P = 1, 10, 50, 100$, where $P_{fa} = 10^{-9}$. Figure 7.13b shows a plot of the probability of detection for Swerling 0, Swerling I, Swerling II, and Swerling III with $n_P = 5$ and $P_{fa} = 10^{-7}$. Figure 7.13a can be reproduced using the following MATLAB code.

```
% Generate Figure 7.13a
close all;
clear all;
pfa = 1e-9;
nfa = log(2) / pfa;
index = 0;
for snr = -10:.5:30
  index = index +1;
  prob1(index) = pd_swerling3 (nfa, 1, snr);
  prob10(index) = pd_swerling3 (nfa, 10, snr);
  prob50(index) = pd_swerling3(nfa, 50, snr);
  prob100(index) = pd_swerling3 (nfa, 100, snr);
end
x = -10:.5:30;
plot(x, prob1,'k',x,prob10,'k:',x,prob50,'k--', x, prob100,'k-.');
axis([-10 30 0 1])
xlabel ('SNR in dB')
ylabel ('Probability of detection')
legend('np = 1','np = 10','np = 50','np = 100')
grid
```

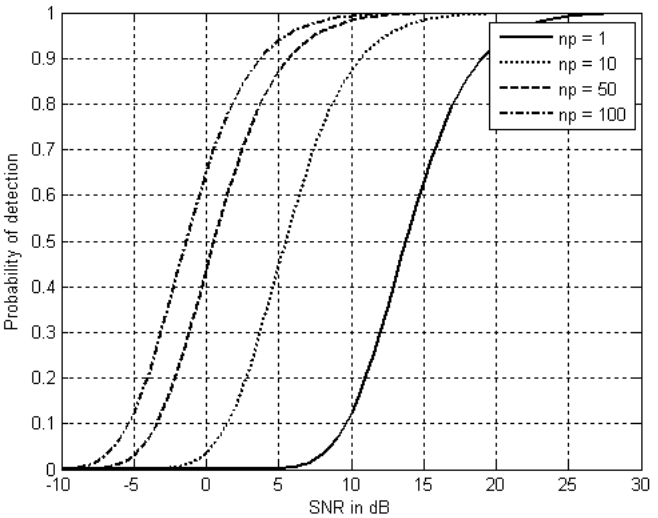

Figure 7.13a. Probability of detection versus SNR. Swerling III. $P_{fa} = 10^{-9}$.

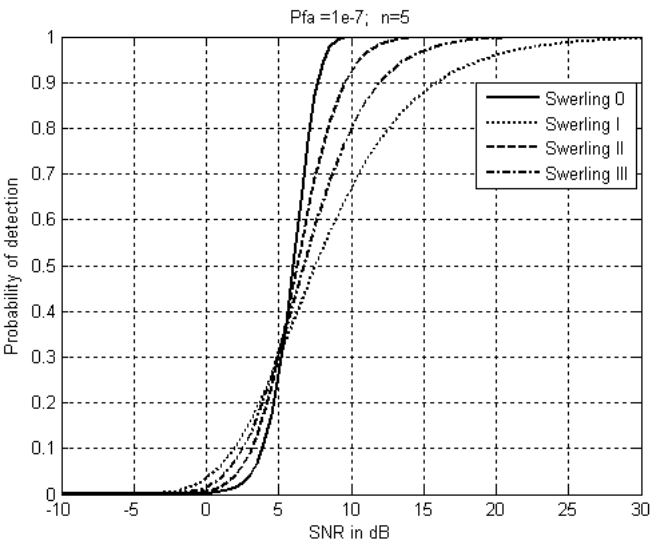

Figure 7.13b. Probability of detection versus SNR. Swerling III, Swerling II, Swerling I, and Swerling 0.

7.7.5. Detection of Swerling IV Targets

The expression for the probability of detection for Swerling IV targets for $n_P < 50$ is

$$P_D = 1 - \left[\gamma_0 + \left(\frac{SNR}{2}\right) n_P \gamma_1 + \left(\frac{SNR}{2}\right)^2 \frac{n_P(n_P-1)}{2!}\gamma_2 + \ldots + \left(\frac{SNR}{2}\right)^{n_P} \gamma_{n_P}\right]\left(1+\frac{SNR}{2}\right)^{-n_P} \tag{7.88}$$

$$\gamma_i = \Gamma_I\left(\frac{v_T}{1+(SNR)/2}\ , n_P + i\right) \tag{7.89}$$

By using the recursive formula

$$\Gamma_I(x, i+1) = \Gamma_I(x, i) - \frac{x^i}{i!\exp(x)} \tag{7.90}$$

then only γ_0 needs to be calculated using Eq. (7.89) and the rest of γ_i are calculated from the following recursion:

$$\gamma_i = \gamma_{i-1} - A_i \qquad ;\ i > 0 \tag{7.91}$$

$$A_i = \frac{v_T/(1+(SNR)/2)}{n_P + i - 1}\ A_{i-1} \qquad ;\ i > 1 \tag{7.92}$$

$$A_1 = \frac{(v_T/(1+(SNR)/2))^{n_P}}{n_P!\exp(v_T/(1+(SNR)/2))} \tag{7.93}$$

$$\gamma_0 = \Gamma_I\left(\frac{v_T}{(1+(SNR)/2)}\ , n_P\right) \tag{7.94}$$

For the case when $n_P \geq 50$, the Gram-Charlier series can be used to calculate the probability of detection. In this case,

$$C_3 = \frac{1}{3\sqrt{n_P}}\frac{2\beta^3 - 1}{(2\beta^2-1)^{1.5}} \qquad ;\ C_6 = \frac{C_3^2}{2} \tag{7.95}$$

$$C_4 = \frac{1}{4n_P}\frac{2\beta^4 - 1}{(2\beta^2-1)^2} \tag{7.96}$$

$$\varpi = \sqrt{n_P(2\beta^2 - 1)} \tag{7.97}$$

$$\beta = 1 + (SNR)/2 \tag{7.98}$$

Figure 7.14 shows plots of the probability of detection as a function of SNR for $n_P = 1, 10, 25, 75$, where $P_{fa} = 10^{-6}$. This figure can be reproduced using the following MATLAB code.

```
clear all; close all;
pfa = 1e-6;
nfa = log(2) / pfa;
index = 0;
for snr = -7:.15:10
  index = index +1;
  prob1(index) = pd_swerling4 (nfa, 5, snr);
  prob10(index) = pd_swerling4 (nfa, 10, snr);
  prob25(index) = pd_swerling4(nfa, 25, snr);
  prob75(index) = pd_swerling4 (nfa, 75, snr);
end
x = -7:.15:10;
plot(x, prob1,'k',x,prob10,'k.',x,prob25,'k:',x, prob75,'k-.','linewidth',1);
xlabel ('SNR - dB')
ylabel ('Probability of detection')
legend('np = 5','np = 10','np = 25','np = 75')
grid; axis tight
```

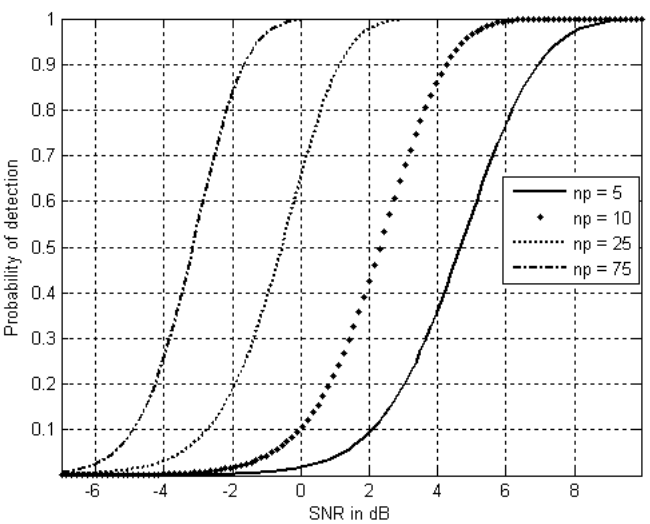

Figure 7.14. Probability of detection versus SNR. Swerling IV. $P_{fa} = 10^{-6}$.

7.8. Computation of the Fluctuation Loss

The fluctuation loss, L_f, can be viewed as the amount of additional SNR required to compensate for the SNR loss due to target fluctuation, given a specific P_D value. Kanter[1] developed an exact analysis for calculating the fluctuation loss. In this text the author will take advantage of the computational power of MATLAB and the MATLAB functions developed for this text to numerically calculate the amount of fluctuation loss. For this purpose consider the MATALB function "*fluct.m"*, where its syntax is as follows:

[SNR] = fluct(pd, pfa, np, sw_case)

where

Symbol	Description	Units	Status
pd	*desired probability of detection*	*none*	*input*
nfa	*desired number of false alarms*	*none*	*input*
np	*number of pulses*	*none*	*input*
sw_case	*0, 1, 2, 3, or 4 depending on the desired Swerling case*	*none*	*input*
SNR	*Resulting SNR*	*dB*	*output*

For example, using the syntax

[SNR0] = fluct(0.8, 1e6, 5, 0)

will calculate the *SNR0* corresponding to a Swerling 0. If one would use this *SNR* in the function *"pd_swerling5.m"* with following syntax

[pd] = pd_swerling5 (1e6, 1, 5, SNR0)

the resulting P_D will be equal to 0.8 . Similarly, if the following syntax is used

[SNR1] = fluct(.8, 1-e-6, 5, 1)

then the value *SNR1* will be that of Swerling 1. Of course, if one would use this *SNR1* value in the function *"pd_swerling1.m"* with following syntax

[pd] = pd_swerling1(1e6, 5, .8, SNR1)

the same P_D of 0.8 will be calculated. Therefore, the fluctuation loss for this case, is equal to *SNR0 - SNR1*.

1. Kanter, I., Exact Detection Probability for Partially Correlated Rayleigh Targets, *IEEE Trans*, AES-22, pp. 184-196, March 1986.

7.9. Cumulative Probability of Detection

Denote the range at which the single pulse SNR is unity (0 dB) as R_0, and refer to it as the reference range. Then, for a specific radar, the single pulse SNR at R_0 is defined by the radar equation and is given by

$$(SNR)_{R_0} = \frac{P_t G^2 \lambda^2 \sigma}{(4\pi)^3 k T_0 B F L R_0^4} = 1 \tag{7.99}$$

The single pulse SNR at any range R is

$$SNR = \frac{P_t G^2 \lambda^2 \sigma}{(4\pi)^3 k T_0 B F L R^4} \tag{7.100}$$

Dividing Eq. (7.100) by Eq. (7.99) yields

$$\frac{SNR}{(SNR)_{R_0}} = \left(\frac{R_0}{R}\right)^4 \tag{7.101}$$

Therefore, if the range R_0 is known, then the SNR at any other range R is

$$(SNR)_{dB} = 40\log\left(\frac{R_0}{R}\right) \tag{7.102}$$

Also, define the range R_{50} as the range at which $P_D = 0.5 = P_{50}$. Normally, the radar unambiguous range R_u is set equal to $2R_{50}$.

The cumulative probability of detection refers to detecting the target at least once by the time it is at range R. More precisely, consider a target closing on a scanning radar, where the target is illuminated only during a scan (frame). As the target gets closer to the radar, its probability of detection increases since the SNR is increased. Suppose that the probability of detection during the nth frame is P_{D_n}; then, the cumulative probability of detecting the target at least once during the nth frame (see Fig. 7.15) is given by

$$P_{C_n} = 1 - \prod_{i=1}^{n} (1 - P_{D_i}) \tag{7.103}$$

P_{D_1} is usually selected to be very small. Clearly, the probability of not detecting the target during the nth frame is $1 - P_{C_n}$. The probability of detection for the ith frame, P_{D_i}, is computed as discussed in the previous section.

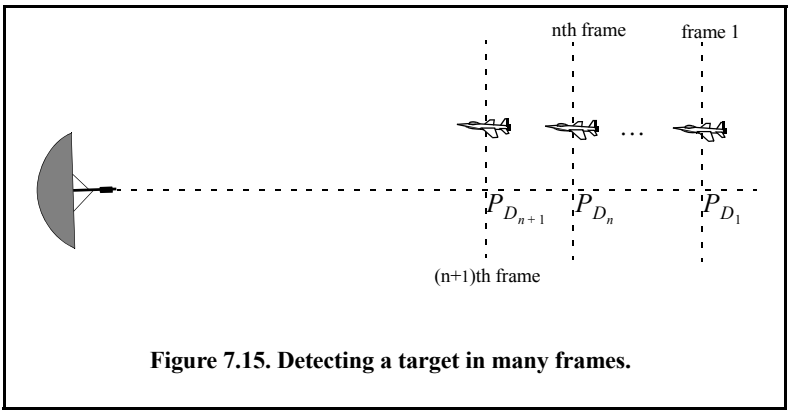

Figure 7.15. Detecting a target in many frames.

Example:

A radar detects a closing target at $R = 10Km$*, with probability of detection* P_D *equal to* 0.5*. Assume* $P_{fa} = 10^{-7}$*. Compute and sketch the single look probability of detection as a function of normalized range (with respect to* $R = 10Km$*), over the interval* $(2-20)Km$*. If the range between two successive frames is* $1Km$*, what is the cumulative probability of detection at* $R = 8Km$*?*

Solution:

From the function "marcumsq.m" the SNR corresponding to $P_D = 0.5$ *and* $P_{fa} = 10^{-7}$ *is approximately 12dB. By using a similar analysis to that which led to Eq. (7.102), we can express the SNR at any range* R *as*

$$(SNR)_R = (SNR)_{10} + 40 \log\frac{10}{R} = 52 - 40 \log R$$

By using the function "marcumsq.m" we can construct the following table:

R *Km*	(SNR) dB	P_D
2	39.09	0.999
4	27.9	0.999
6	20.9	0.999
8	15.9	0.999
9	13.8	0.9
10	12.0	0.5
11	10.3	0.25

R *Km*	(SNR) dB	P_D
12	8.8	0.07
14	6.1	0.01
16	3.8	ε
20	0.01	ε

where ε *is very small. A sketch of* P_D *versus normalized range is shown in Fig. 7.16.*

The cumulative probability of detection is given in Eq. (7.104), where the probability of detection of the first frame is selected to be very small. Thus, we can arbitrarily choose frame 1 to be at $R = 16Km$. *Note that selecting a different starting point for frame 1 would have a negligible effect on the cumulative probability (we only need* P_{D_1} *to be very small). Below is a range listing for frames 1 through 9, where frame 9 corresponds to* $R = 8Km$.

frame	1	2	3	4	5	6	7	8	9
range in Km	16	15	14	13	12	11	10	9	8

The cumulative probability of detection at 8 Km is then

$$P_{C_9} = 1-(1-0.999)(1-0.9)(1-0.5)(1-0.25)(1-0.07)$$
$$(1-0.01)(1-\varepsilon)^2 \approx 0.9998$$

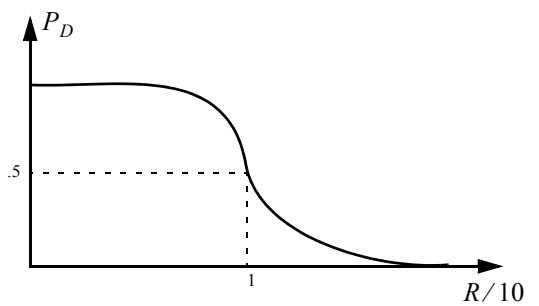

Figure 7.16. Cumulative probability of detection versus normalized range.

7.10. Constant False Alarm Rate (CFAR)

The detection threshold is computed so that the radar receiver maintains a constant predetermined probability of false alarm. Equation (7.20) gives the relationship between the threshold value V_T and the probability of false alarm P_{fa}, and for convenience is repeated here as Eq. (7.104):

$$v_T = \sqrt{2\sigma^2 \ln\left(\frac{1}{P_{fa}}\right)} \tag{7.104}$$

If the noise power σ^2 is constant, then a fixed threshold can satisfy Eq. (7.104). However, due to many reasons this condition is rarely true. Thus, in order to maintain a constant probability of false alarm, the threshold value must be continuously updated based on the estimates of the noise variance. The process of continuously changing the threshold value to maintain a constant probability of false alarm is known as Constant False Alarm Rate (CFAR).

Three different types of CFAR processors are primarily used. They are adaptive threshold CFAR, nonparametric CFAR, and nonlinear receiver techniques. Adaptive CFAR assumes that the interference distribution is known and approximates the unknown parameters associated with these distributions. Nonparametric CFAR processors tend to accommodate unknown interference distributions. Nonlinear receiver techniques attempt to normalize the root-mean-square amplitude of the interference. In this book only analog Cell-Averaging CFAR (CA-CFAR) technique is examined. The analysis presented in this section closely follows Urkowitz[1].

7.10.1. Cell-Averaging CFAR (Single Pulse)

The CA-CFAR processor is shown in Fig. 7.17. Cell averaging is performed on a series of range and/or Doppler bins (cells). The echo return for each pulse is detected by a square-law detector. In analog implementation these cells are obtained from a tapped delay line. The Cell Under Test (CUT) is the central cell. The immediate neighbors of the CUT are excluded from the averaging process due to a possible spillover from the CUT. The output of M reference cells ($M/2$ on each side of the CUT) is averaged. The threshold value is obtained by multiplying the averaged estimate from all reference cells by a constant K_0 (used for scaling). A detection is declared in the CUT if

$$Y_1 \geq K_0 Z \tag{7.105}$$

1. Urkowitz, H., Decision and Detection Theory, unpublished lecture notes. Lockheed Martin Co., Moorestown, NJ.

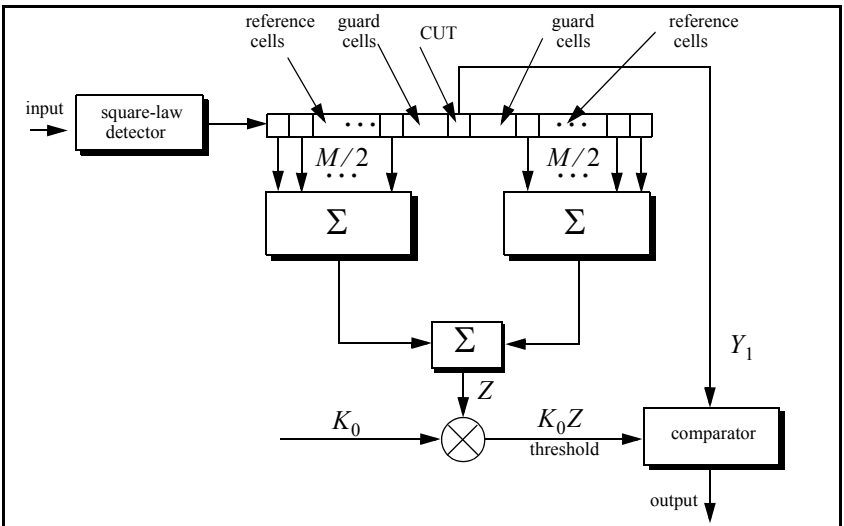

Figure 7.17. Conventional CA-CFAR.

CA-CFAR assumes that the target of interest is in the CUT and all reference cells contain zero-mean independent Gaussian noise of variance ψ^2. Therefore, the output of the reference cells, Z, represents a random variable with gamma probability density function (special case of the chi-square) with $2M$ degrees of freedom. In this case, the gamma *pdf* is

$$f(z) = \frac{z^{(M/2)-1} e^{(-z/2\psi^2)}}{2^{M/2}\ \sigma^M \Gamma(M/2)} \qquad ;\ z > 0 \tag{7.106}$$

The probability of false alarm corresponding to a fixed threshold was derived earlier. When CA-CFAR is implemented, then the probability of false alarm can be derived from the conditional false alarm probability, which is averaged over all possible values of the threshold in order to achieve an unconditional false alarm probability. The conditional probability of false alarm when $y = V_T$ can be written as

$$P_{fa}(v_T = y) = e^{-y/2\sigma^2} \tag{7.107}$$

It follows that the unconditional probability of false alarm is

$$P_{fa} = \int_0^\infty P_{fa}(v_T = y) f(y)\, dy \tag{7.108}$$

where $f(y)$ is the *pdf* of the threshold, which except for the constant K_0 is the same as that defined in Eq. (7.106). Therefore,

$$f(y) = \frac{y^{M-1} e^{(-y/2K_0\psi^2)}}{(2K_0\sigma^2)^M \Gamma(M)} \quad ; \; y \geq 0 \tag{7.109}$$

Performing the integration in Eq. (7.108) yields

$$P_{fa} = 1/(1+K_0)^M \tag{7.110}$$

Observation of Eq. (7.110) shows that the probability of false alarm is now independent of the noise power, which is the objective of CFAR processing.

7.10.2. Cell-Averaging CFAR with Noncoherent Integration

In practice, CFAR averaging is often implemented after noncoherent integration, as illustrated in Fig. 7.18. Now, the output of each reference cell is the sum of n_P squared envelopes. It follows that the total number of summed reference samples is Mn_P. The output Y_1 is also the sum of n_P squared envelopes. When noise alone is present in the CUT, Y_1 is a random variable whose *pdf* is a gamma distribution with $2n_p$ degrees of freedom. Additionally, the summed output of the reference cells is the sum of Mn_P squared envelopes. Thus, Z is also a random variable which has a gamma *pdf* with $2Mn_P$ degrees of freedom.

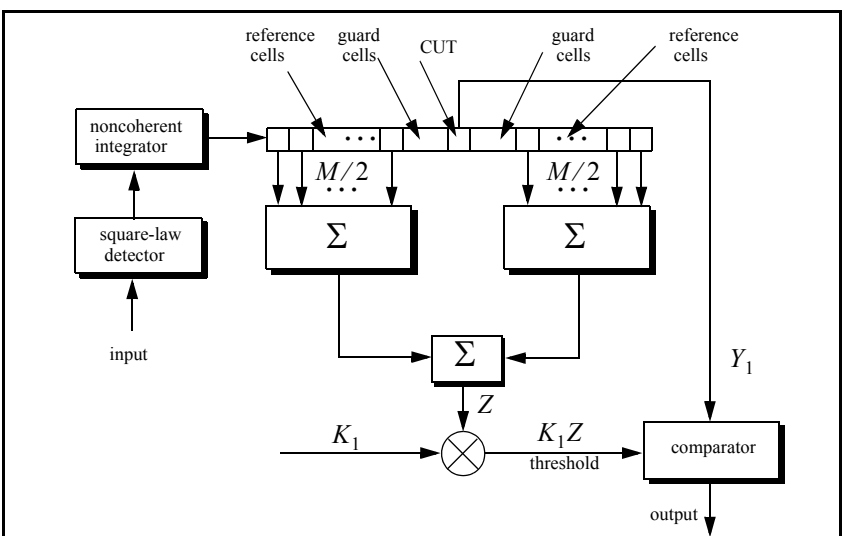

Figure 7.18. Conventional CA-CFAR with noncoherent integration.

The probability of false alarm is then equal to the probability that the ratio Y_1/Z exceeds the threshold. More precisely,

$$P_{fa} = Prob\{Y_1/Z > K_1\} \tag{7.111}$$

Equation (7.111) implies that one must first find the joint *pdf* for the ratio Y_1/Z. However, this can be avoided if P_{fa} is first computed for a fixed threshold value V_T, then averaged over all possible values of the threshold. Therefore, let the conditional probability of false alarm when $y = v_T$ be $P_{fa}(v_T = y)$. It follows that the unconditional false alarm probability is

$$P_{fa} = \int_0^\infty P_{fa}(v_T = y) f(y) dy \tag{7.112}$$

where $f(y)$ is the *pdf* of the threshold. In view of this, the probability density function describing the random variable $K_1 Z$ is given by

$$f(y) = \frac{(y/K_1)^{Mn_P - 1} e^{(-y/2K_0\sigma^2)}}{(2\sigma^2)^{Mn_P} K_1 \ \Gamma(Mn_P)} \quad ; \ y \geq 0 \tag{7.113}$$

It can be shown that in this case the probability of false alarm is independent of the noise power and is given by

$$P_{fa} = \frac{1}{(1+K_1)^{Mn_P}} \sum_{k=0}^{n_P - 1} \frac{1}{k!} \frac{\Gamma(Mn_P + k)}{\Gamma(Mn_P)} \left(\frac{K_1}{1+K_1}\right)^k \tag{7.114}$$

which is identical to Eq. (7.110) when $K_1 = K_0$ and $n_P = 1$.

7.11. MATLAB Programs and Routines

This section presents listings for all the MATLAB programs used to produce all of the MATLAB-generated figures in this chapter. Additionally, other specific MATLAB functions are also presented. They are listed in the same order they appear in the chapter.

7.11.1. MATLAB Function "que_func.m"

The function *"que_func.m"* computes $F(x)$ using Eqs. (7.17) and (7.18). The syntax is as follows:

fofx = que_func (x)

MATLAB Function** "que_func.m" **Listing

```
function fofx = que_func(x)
% This function computes the value of the Q-function
% It uses the approximation in Eqs. (7.17) and (7.18)
if (x >= 0)
   denom = 0.661 * x + 0.339 * sqrt(x^2 + 5.51);
  expo = exp(-x^2 /2.0);
  fofx = 1.0 - (1.0 / sqrt(2.0 * pi)) * (1.0 / denom) * expo;
else
  denom = 0.661 * x + 0.339 * sqrt(x^2 + 5.51);
  expo = exp(-x^2 /2.0);
  value = 1.0 - (1.0 / sqrt(2.0 * pi)) * (1.0 / denom) * expo;
  fofx = 1.0 - value;
end
```

7.11.2. MATLAB Function "marcumsq.m"

This function utilizes Parl's method to compute P_D. The syntax is as follows:

$$Pd = marcumsq(a,b)$$

MATLAB Function *"marcumsq.m"* **Listing**

```
function Pd = marcumsq (a,b); % This function uses Parl's method to compute PD
max_test_value = 5000.;
if (a < b)
  alphan0 = 1.0;
  dn = a / b;
else
  alphan0 = 0.;
  dn = b / a;
end
alphan_1 = 0.;
betan0 = 0.5;
betan_1 = 0.;
D1 = dn;
n = 0;
ratio = 2.0 / (a * b);
r1 = 0.0;
betan = 0.0;
alphan = 0.0;
while betan < 1000.,
  n = n + 1;
  alphan = dn + ratio * n * alphan0 + alphan;
  betan = 1.0 + ratio * n * betan0 + betan;
  alphan_1 = alphan0;
  alphan0 = alphan;
  betan_1 = betan0;
```

```
  betan0 = betan;
  dn = dn * D1;
end
PD = (alphan0 / (2.0 * betan0)) * exp( -(a-b)^2 / 2.0);
if ( a >= b)
  PD = 1.0 - PD;
end
return
```

7.11.3. MATLAB Function "improv_fac.m"

The function *"improv_fac.m"* calculates the improvement factor using Eq. (7.45). The syntax is as follows:

[impr_of_np] = improv_fac (np, pfa, pd)

where

Symbol	Description	Units	Status
np	*number of integrated pulses*	*none*	*input*
pfa	*probability of false alarms*	*none*	*input*
pd	*probability of detection*	*none*	*input*
impr_of_np	*improvement factor*	*output*	*dB*

MATLAB Function** "improv_fac.m" **Listing

```
function impr_of_np = improv_fac (np, pfa, pd)
% This function computes the noncoherent integration improvement
% factor using the empirical formula defined in Eq. (7.54)
fact1 = 1.0 + log10( 1.0 / pfa) / 46.6;
fact2 = 6.79 * (1.0 + 0.235 * pd);
fact3 = 1.0 - 0.14 * log10(np) + 0.0183 * (log10(np))^2;
impr_of_np = fact1 * fact2 * fact3 * log10(np);
return
```

7.11.4. MATLAB Function "threshold.m"

The function *"threshold.m"* calculates the threshold value given the algorithm described in Section 7.6. The syntax is as follows:

[pfa, vt] = threshold (nfa, np)

where

Symbol	Description	Units	Status
nfa	*number of false alarm*	*none*	*input*
np	*number of pulses*	*none*	*input*
pfa	*probability of alarm*	*none*	*output*
vt	*threshold value*	*none*	*output*

MATLAB Function** "threshold.m" **Listing

```
function [pfa, vt] = threshold (nfa, np)
% This function calculates the threshold value from nfa and np.
% The Newton-Raphson  recursive formula is used
% This function uses "gammainc.m".
delmax = .00001;
eps = 0.000000001;
delta =10000.;
pfa = np * log(2) / nfa;
sqrtpfa = sqrt(-log10(pfa));
sqrtnp = sqrt(np);
vt0 = np - sqrtnp + 2.3 * sqrtpfa * (sqrtpfa + sqrtnp - 1.0);
vt = vt0;
while (abs(delta) >= vt0)
  igf = gammainc(vt0,np);
  num = 0.5^(np/nfa) - igf;
  temp = (np-1) * log(vt0+eps) - vt0 - factor(np-1);
  deno = exp(temp);
  vt = vt0 + (num / (deno+eps));
  delta = abs(vt - vt0) * 10000.0;
  vt0 = vt;
```

7.11.5. MATLAB Function "pd_swerling5.m"

The function *"pd_swerling5.m"* calculates the probability of detection for Swerling 0 targets. The syntax is as follows:

[pd] = pd_swerling5 (input1, indicator, np, snr)

where

Symbol	Description	Units	Status
input1	P_{fa} *or* n_{fa}	*none*	*input*
indicator	*1 when input1* = P_{fa} *2 when input1* = n_{fa}	*none*	*input*

Symbol	Description	Units	Status
np	*number of integrated pulses*	*none*	*input*
snr	*SNR*	*dB*	*input*
pd	*probability of detection*	*none*	*output*

MATLAB Function** "pd_swerling5.m" **Listing

```
function pd = pd_swerling5 (input1, indicator, np, snrbar)
% This function is used to calculate the probability of detection
% for Swerling 5 or 0 targets for np>1.
if(np == 1)
  'Stop, np must be greater than 1'
  return
end
format long
snrbar = 10.0.^(snrbar./10.);
eps = 0.00000001;
delmax = .00001;
delta =10000.;
% Calculate the threshold Vt
if (indicator ~=1)
  nfa = input1;
  pfa =  np * log(2) / nfa;
else
  pfa = input1;
  nfa = np * log(2) / pfa;
end
sqrtpfa = sqrt(-log10(pfa));
sqrtnp = sqrt(np);
vt0 = np - sqrtnp + 2.3 * sqrtpfa * (sqrtpfa + sqrtnp - 1.0);
vt = vt0;
while (abs(delta) >= vt0)
  igf = incomplete_gamma(vt0,np);
  num = 0.5^(np/nfa) - igf;
  temp = (np-1) * log(vt0+eps) - vt0 - factor(np-1);
  deno = exp(temp);
  vt = vt0 + (num / (deno+eps));
  delta = abs(vt - vt0) * 10000.0;
  vt0 = vt;
end
% Calculate the Gram-Chrlier coefficients
temp1 = 2.0 .* snrbar + 1.0;
omegabar = sqrt(np .* temp1);
c3 = -(snrbar + 1.0 / 3.0) ./ (sqrt(np) .* temp1.^1.5);
c4 = (snrbar + 0.25) ./ (np .* temp1.^2.);
c6 = c3 .* c3 ./2.0;
```

```
V = (vt - np .* (1.0 + snrbar)) ./ omegabar;
Vsqr = V .*V;
val1 = exp(-Vsqr ./ 2.0) ./ sqrt( 2.0 * pi);
val2 = c3 .* (V.^2 -1.0) + c4 .* V .* (3.0 - V.^2) -...
  c6 .* V .* (V.^4 - 10. .* V.^2 + 15.0);
q = 0.5 .* erfc (V./sqrt(2.0));
pd =  q - val1 .* val2;
return
```

7.11.6. MATLAB Function "pd_swerling1.m"

The function *"pd_swerling1.m"* calculates the probability of detection for Swerling I type targets. The syntax is as follows:

[pd] = pd_swerling1 (nfa, np, snr)

where

Symbol	Description	Units	Status
nfa	*Marcum's false alarm number*	*none*	*input*
np	*number of integrated pulses*	*none*	*input*
snr	*SNR*	*dB*	*input*
pd	*probability of detection*	*none*	*output*

MATLAB Function** "pd_swerling1.m" **Listing

```
function [pd] = pd_swerling1 (nfa, np, snrbar)
% This function is used to calculate the probability of detection
% for Swerling 1 targets.
format long
snrbar = 10.0^(snrbar/10.);
eps = 0.00000001;
delmax = .00001;
delta =10000.;
% Calculate the threshold Vt
pfa =  np * log(2) / nfa;
sqrtpfa = sqrt(-log10(pfa));
sqrtnp = sqrt(np);
vt0 = np - sqrtnp + 2.3 * sqrtpfa * (sqrtpfa + sqrtnp - 1.0);
vt = vt0;
while (delta < (vt0/10000));
  igf = gammainc(vt0,np);
  num = 0.5^(np/nfa) - igf;
  deno = -exp(-vt0) * vt0^(np-1) /factorial(np-1);
  vt = vt0 - (num / (deno+eps));
  delta = abs(vt - vt0);
  vt0 = vt;
```

```
end
if (np == 1)
  temp = -vt / (1.0 + snrbar);
  pd = exp(temp);
  return
end
  temp1 = 1.0 + np * snrbar;
  temp2 = 1.0 / (np *snrbar);
  temp = 1.0 + temp2;
  val1 = temp^(np-1.);
  igf1 = gammainc(vt,np-1);
  igf2 = gammainc(vt/temp,np-1);
  pd = 1.0 - igf1 + val1 * igf2 * exp(-vt/temp1);
  return
```

7.11.7. MATLAB Function "pd_swerling2.m"

The function *"pd_swerling2.m"* calculates P_D for Swerling II type targets. The syntax is as follows:

[pd] = pd_swerling2 (nfa, np, snr)

where

Symbol	Description	Units	Status
nfa	*Marcum's false alarm number*	*none*	*input*
np	*number of integrated pulses*	*none*	*input*
snr	*SNR*	*dB*	*input*
pd	*probability of detection*	*none*	*output*

MATLAB Function** "pd_swerling2.m" **Listing

```
function [pd] = pd_swerling2 (nfa, np, snrbar)
% This function is used to calculate the probability of detection
% for Swerling 2 targets.
format long
snrbar = 10.0^(snrbar/10.);
eps = 0.00000001;
delmax = .00001;
delta =10000.;
% Calculate the threshold Vt
pfa =  np * log(2) / nfa;
sqrtpfa = sqrt(-log10(pfa));
sqrtnp = sqrt(np);
vt0 = np - sqrtnp + 2.3 * sqrtpfa * (sqrtpfa + sqrtnp - 1.0);
vt = vt0;
while (delta < (vt0/10000));
```

```
  igf = gammainc(vt0,np);
  num = 0.5^(np/nfa) - igf;
  deno = -exp(-vt0) * vt0^(np-1) /factorial(np-1);
  vt = vt0 - (num / (deno+eps));
  delta = abs(vt - vt0);
  vt0 = vt;
end
if (np <= 50)
  temp = vt / (1.0 + snrbar);
  pd = 1.0 - gammainc(temp,np);
  return
else
  temp1 = snrbar + 1.0;
  omegabar = sqrt(np) * temp1;
  c3 = -1.0 / sqrt(9.0 * np);
  c4 = 0.25 / np;
  c6 = c3 * c3 /2.0;
  V = (vt - np * temp1) / omegabar;
  Vsqr = V *V;
  val1 = exp(-Vsqr / 2.0) / sqrt( 2.0 * pi);
  val2 = c3 * (V^2 -1.0) + c4 * V * (3.0 - V^2) - ...
    c6 * V * (V^4 - 10. * V^2 + 15.0);
  q = 0.5 * erfc (V/sqrt(2.0));
  pd =  q - val1 * val2;
end
return
```

7.11.8. MATLAB Function "pd_swerling3.m"

The function *"pd_swerling3.m"* calculates P_D for Swerling III type targets. The syntax is as follows:

[pd] = pd_swerling3 (nfa, np, snr)

where

Symbol	Description	Units	Status
nfa	*Marcum's false alarm number*	*none*	*input*
np	*number of integrated pulses*	*none*	*input*
snr	*SNR*	*dB*	*input*
pd	*probability of detection*	*none*	*output*

MATLAB Function** "pd_swerling3.m" **Listing

```
function [pd] = pd_swerling3 (nfa, np, snrbar)
% This function is used to calculate the probability of detection
% for Swerling 3 targets.
```

```
format long
snrbar = 10.0^(snrbar/10.);
eps = 0.00000001;
delmax = .00001;
delta =10000.;
% Calculate the threshold Vt
pfa =  np * log(2) / nfa;
sqrtpfa = sqrt(-log10(pfa));
sqrtnp = sqrt(np);
vt0 = np - sqrtnp + 2.3 * sqrtpfa * (sqrtpfa + sqrtnp - 1.0);
vt = vt0;
while (delta < (vt0/10000));
  igf = gammainc(vt0,np);
  num = 0.5^(np/nfa) - igf;
  deno = -exp(-vt0) * vt0^(np-1) /factorial(np-1);
  vt = vt0 - (num / (deno+eps));
  delta = abs(vt - vt0);
  vt0 = vt;
end
temp1 = vt / (1.0 + 0.5 * np *snrbar);
temp2 = 1.0 + 2.0 / (np * snrbar);
temp3 = 2.0 * (np - 2.0) / (np * snrbar);
ko = exp(-temp1) * temp2^(np-2.) * (1.0 + temp1 - temp3);
if (np <= 2)
  pd = ko;
  return
else
  ko = exp(-temp1) * temp2^(np-2.) * (1.0 + temp1 - temp3);
  temp4 = vt^(np-1.) * exp(-vt) / (temp1 * (factorial(np-2.)));
  temp5 =  vt / (1.0 + 2.0 / (np *snrbar));
  pd = temp4 + 1.0 - gammainc(vt,np-1.) + ko * gammainc(temp5,np-1.);
end
return
```

7.11.9. MATLAB Function "pd_swerling4.m"

The function *"pd_swerling4.m"* calculates P_D for Swerling IV type targets. The syntax is as follows:

[pd] = pd_swerling4 (nfa, np, snr)

where

Symbol	Description	Units	Status
nfa	*Marcum's false alarm number*	*none*	*input*
np	*number of integrated pulses*	*none*	*input*

Symbol	Description	Units	Status
snr	*SNR*	*dB*	*input*
pd	*probability of detection*	*none*	*output*

MATLAB Function** "pd_swerling4.m" **Listing

```
function [pd] = pd_swerling4 (nfa, np, snrbar)
% This function is used to calculate the probability of detection
% for Swerling 4 targets.
format long
snrbar = 10.0^(snrbar/10.);
eps = 0.00000001;
delmax = .00001;
delta =10000.;
% Calculate the threshold Vt
pfa =  np * log(2) / nfa;
sqrtpfa = sqrt(-log10(pfa));
sqrtnp = sqrt(np);
vt0 = np - sqrtnp + 2.3 * sqrtpfa * (sqrtpfa + sqrtnp - 1.0);
vt = vt0;
while (delta < (vt0/10000));
  igf = gammainc(vt0,np);
  num = 0.5^(np/nfa) - igf;
  deno = -exp(-vt0) * vt0^(np-1) /factorial(np-1);
  vt = vt0 - (num / (deno+eps));
  delta = abs(vt - vt0);
  vt0 = vt;
end
h8 = snrbar /2.0;
beta = 1.0 + h8;
beta2 = 2.0 * beta^2 - 1.0;
beta3 = 2.0 * beta^3;
if (np >= 50)
  temp1 = 2.0 * beta -1;
  omegabar = sqrt(np * temp1);
  c3 = (beta3 - 1.) / 3.0 / beta2 / omegabar;
  c4 = (beta3 * beta3 - 1.0) / 4. / np /beta2 /beta2;
  c6 = c3 * c3 /2.0;
  V = (vt - np * (1.0 + snrbar)) / omegabar;
  Vsqr = V *V;
  val1 = exp(-Vsqr / 2.0) / sqrt( 2.0 * pi);
  val2 = c3 * (V^2 -1.0) + c4 * V * (3.0 - V^2) - c6 * V * (V^4 - 10. * V^2 + 15.0);
  q = 0.5 * erfc (V/sqrt(2.0));
  pd =  q - val1 * val2;
  return
else
```

```
  gamma0 = gammainc(vt/beta,np);
  a1 = (vt / beta)^np / (factorial(np) * exp(vt/beta));
  sum = gamma0;
  for i = 1:1:np
    temp1 = gamma0;
    if (i == 1)
      ai = a1;
    else
      ai = (vt / beta) * a1 / (np + i -1);
    end
    gammai = gamma0 - ai;
    gamma0 = gammai;
    a1 = ai;
    for ii = 1:1:i
      temp1 = temp1 * (np + 1 - ii);
    end
    term = (snrbar /2.0)^i * gammai * temp1 / (factorial(i));
    sum = sum + term;
  end
  pd = 1.0 - (sum / beta^np);
end
pd = max(pd,0.);
return
```

7.11.10. MATLAB Function "fluct_loss.m"

This functions has been described in Section 7.8.

MATLAB Function *"fluct_loss.m"*

```
function [SNR] = fluct(pd, nfa, np, sw_case)
% This function calculates the SNR fluctuation loss for Swerling models
% A negative Lf value indicates SNR gain instead of loss
format long
% *************** Swerling 5 case ****************
% check to make sure that np>1
pfa =  np * log(2) / nfa;
if (sw_case == 0)
if (np ==1)
   nfa = 1/pfa;
   b = sqrt(-2.0 * log(pfa));
   Pd_Sw5 = 0.001;
   snr_inc = 0.1 - 0.005;
   while(Pd_Sw5 <= pd)
      snr_inc = snr_inc + 0.005;
      a = sqrt(2.0 * 10^(.1*snr_inc));
      Pd_Sw5 = marcumsq(a,b);
   end
```

```
    PD_SW5 = Pd_Sw5;
    SNR = snr_inc;
else
    % np > 1 use MATLAB function pd_swerling5.m
    snr_inc = 0.1 - 0.001;
    Pd_Sw5 = 0.001;
    while(Pd_Sw5 <= pd)
        snr_inc = snr_inc + 0.001;
        Pd_Sw5 = pd_swerling5(pfa, 1, np, snr_inc);
    end
    PD_SW5 = Pd_Sw5;
    SNR = snr_inc;
end
end
 % ********** End Swerling 5 case ***********
% ********** Swerling 1 case **********
% compute the false alarm number
if (sw_case==1)
    Pd_Sw1 = 0.001;
    snr_inc = 0.1 - 0.001;
    while(Pd_Sw1 <= pd)
        snr_inc = snr_inc + 0.001;
        Pd_Sw1 = pd_swerling1(nfa, np, snr_inc);
    end
    PD_SW1 = Pd_Sw1;
    SNR = snr_inc;
end
 % ********** End Swerling 1 case ***********
% ********** Swerling 2 case **********
if (sw_case == 2)
    Pd_Sw2 = 0.001;
    snr_inc = 0.1 - 0.001;
    while(Pd_Sw2 <= pd)
        snr_inc = snr_inc + 0.001;
        Pd_Sw2 = pd_swerling2(nfa, np, snr_inc);
    end
    PD_SW2 = Pd_Sw2;
    SNR = snr_inc;
end
 % ********** End Swerling 2 case ***********
% ********** Swerling 3 case **********
if (sw_case == 3)
    Pd_Sw3 = 0.001;
    snr_inc = 0.1 - 0.001;
    while(Pd_Sw3 <= pd)
        snr_inc = snr_inc + 0.001;
        Pd_Sw3 = pd_swerling3(nfa, np, snr_inc);
    end
```

```
   PD_SW3 = Pd_Sw3;
   SNR = snr_inc;
end
 % *************** End Swerling 3 case ************
% *************** Swerling 4 case ****************
if (sw_case == 4)
   Pd_Sw4 = 0.001;
   snr_inc = 0.1 - 0.001;
   while(Pd_Sw4 <= pd)
      snr_inc = snr_inc + 0.001;
      Pd_Sw4 = pd_swerling4(nfa, np, snr_inc);
   end
   PD_SW4 = Pd_Sw4;
   SNR = snr_inc;
end
```

Appendix 7.A The Incomplete Gamma Function

The Gamma Function

Define the Gamma function (not the incomplete Gamma function) of the variable z (generally complex) as

$$\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx \tag{7.115}$$

and when z is a positive integer, then

$$\Gamma(z) = (z-1)! \tag{7.116}$$

One very useful and frequently used property is

$$\Gamma(z+1) = z\Gamma(z) \tag{7.117}$$

The Incomplete Gamma Function

The incomplete gamma function. $\Gamma_I(u, q)$ used in this text is given by

$$\Gamma_I(u, q) = \int_0^{u\sqrt{q+1}} \frac{e^{-x} x^q}{q!} dx \tag{7.118}$$

Another definition, which is often used in the literature, for the incomplete Gamma function is

$$\Gamma_I[z, q] = \int_q^{\infty} x^{z-1} e^{-x} dx \tag{7.119}$$

It follows that

$$\Gamma(z) = \Gamma_I[z, 0] = \int_0^{\infty} x^{z-1} e^{-x} dx \tag{7.120}$$

which is the same as Eq. (7.115). Furthermore, for a positive integer n, the incomplete Gamma function can be represented by

$$\Gamma_I[n, z] = (n-1)! e^{-z} \sum_{k=0}^{n-1} \frac{z^k}{k!} \tag{7.121}$$

In order to relate $\Gamma_I[n, z]$ and $\Gamma_I(u, q)$ compute the following relation

$$\Gamma_I[a, 0] - \Gamma_I[a, z] = \int_0^{\infty} x^{a-1} e^{-x} dx - \int_z^{\infty} x^{a-1} e^{-x} dx = \int_0^{z} x^{a-1} e^{-x} dx \tag{7.122}$$

Applying the change of variables $a = q + 1$ and $z = u\sqrt{q+1}$ yields

$$\Gamma_I[q+1, 0] - \Gamma_I[q+1, u\sqrt{q+1}] = \int_0^{u\sqrt{q+1}} x^q e^{-x} dx \tag{7.123}$$

and if q is a positive integer then

$$\frac{\Gamma_I[q+1, 0] - \Gamma_I[q+1, u\sqrt{q+1}]}{q!} = \int_0^{u\sqrt{q+1}} \frac{x^q e^{-x}}{q!} dx = \Gamma_I(u, q) \tag{7.124}$$

Using Eq. (7.116) and (7.121) in Eq. (7.124) yields

$$\Gamma_I(u, q) = 1 - \frac{(q+1-1)! e^{-u\sqrt{q+1}}}{q!} \sum_{k=0}^{q} \frac{(u\sqrt{q+1})^k}{k!} \tag{7.125}$$

Finally, the incomplete Gamma function can be written as

$$\Gamma_I(u, q) = 1 - e^{-u\sqrt{q+1}} \sum_{k=0}^{q} \frac{(u\sqrt{q+1})^k}{k!} \tag{7.126}$$

The two limiting values for Eq. (7.126) are

$$\Gamma_I(0, q) = 0 \qquad \Gamma_I(\infty, q) = 1 \tag{7.127}$$

Figure 7A.1 shows the incomplete gamma function for $q = 1, 3, 5, 8$. This figure can be reproduced using the following MATLAB code which utilizes the built-in MATLAB function *"gammainc.m"*.

```
% This program can be used to reproduce Fig. 7A.1
close all; clear all
x=linspace(0,20,200);
y1 = gammainc(x,1);
y2 = gammainc(x,3);
y3 = gammainc(x,5);
y4 = gammainc(x,8);
plot(x,y1,'k',x,y2,'k:',x,y3,'k--',x,y4,'k-.')
legend('q = 1','q = 3','q = 5','q = 8')
xlabel('x'); ylabel('Incomplete Gamma function (x,q)')
grid
```

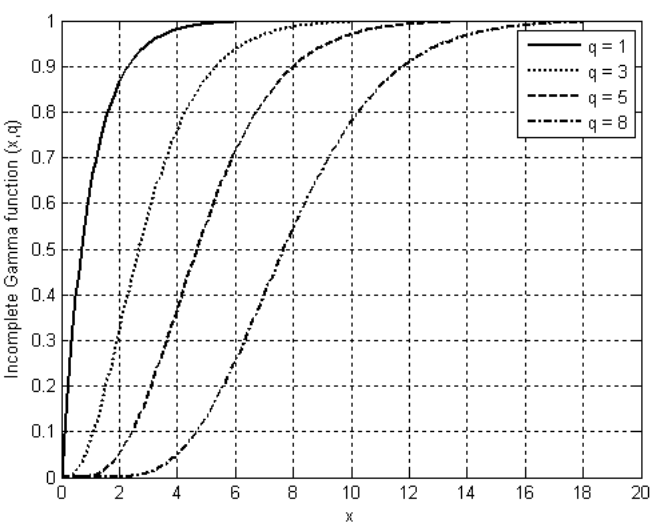

Figure 7A.1. The incomplete Gamma function for four values of q.

Problems

7.1. In the case of noise alone, the quadrature components of a radar return are independent Gaussian random variables with zero mean and variance σ^2. Assume that the radar processing consists of envelope detection followed by threshold decision. (a) Write an expression for the *pdf* of the envelope; (b) determine the threshold V_T as a function of σ that ensures a probability of false alarm $P_{fa} \leq 10^{-8}$.

7.2. (a) Derive Eq. (7.13); (b) derive Eq. (7.15).

7.3. A pulsed radar has the following specifications: time of false alarm $T_{fa} = 10\ min$, probability of detection $P_D = 0.95$, operating bandwidth $B = 1MHz$. (a) What is the probability of false alarm P_{fa}? (b) What is the single pulse SNR? (c) Assuming noncoherent integration of 100 pulses, what is the SNR reduction so that P_D and P_{fa} remain unchanged?

7.4. An L-band radar has the following specifications: operating frequency $f_0 = 1.5GHz$, operating bandwidth $B = 2MHz$, noise figure $F = 8dB$, system losses $L = 4dB$, time of false alarm $T_{fa} = 12\ minutes$, detection range $R = 12Km$, probability of detection $P_D = 0.5$, antenna gain $G = 5000$, and target RCS $\sigma = 1m^2$. (a) Determine the PRF f_r, the pulse width τ, the peak power P_t, the probability of false alarm P_{fa}, and the minimum detectable signal level S_{min}. (b) How can you reduce the transmitter power to achieve the same performance when 10 pulses are integrated noncoherently? (c) If the radar operates at a shorter range in the single pulse mode, find the new probability of detection when the range decreases to $9Km$.

7.5. (a) Show how you can use the radar equation to determine the PRF f_r, the pulse width τ, the peak power P_t, the probability of false alarm P_{fa}, and the minimum detectable signal level S_{min}. Assume the following specifications: operating frequency $f_0 = 1.5MHz$, operating bandwidth $B = 1MHz$, noise figure $F = 10dB$, system losses $L = 5dB$, time of false alarm $T_{fa} = 20\ min$, detection range $R = 12Km$, probability of detection $P_D = 0.5$ (three pulses). (b) If post detection integration is assumed, determine the SNR.

7.6. Show that when computing the probability of detection at the output of an envelope detector, it is possible to use Gaussian probability approximation when the SNR is very large.

7.7. A radar system uses a threshold detection criterion. The probability of false alarm $P_{fa} = 10^{-10}$. (a) What must be the average SNR at the input of a linear detector so that the probability of miss is $P_m = 0.15$? Assume large SNR approximation. (b) Write an expression for the *pdf* at the output of the envelope detector.

7.8. An X-band radar has the following specifications: received peak power $10^{-10}W$, probability of detection $P_D = 0.95$, time of false alarm $T_{fa} = 8\ min$, pulse width $\tau = 2\mu s$, operating bandwidth $B = 2MHz$, operating frequency $f_0 = 10GHz$, and detection range $R = 100Km$. Assume single pulse processing. (a) Compute the probability of false alarm P_{fa}. (b) Determine the SNR at the output of the IF amplifier. (c) At what SNR would the probability of detection drop to 0.9 (P_{fa} does not change)? (d) What is the increase in range that corresponds to this drop in the probability of detection?

7.9. A certain radar utilizes 10 pulses for noncoherent integration. The single pulse SNR is $15dB$ and the probability of miss is $P_m = 0.15$. (a) Compute the probability of false alarm P_{fa}. (b) Find the threshold voltage V_T.

7.10. Consider a scanning low PRF radar. The antenna half-power beam width is $1.5°$, and the antenna scan rate is $35°$ per second. The pulse width is $\tau = 2\mu s$, and the PRF is $f_r = 400Hz$. (a) Compute the radar operating bandwidth. (b) Calculate the number of returned pulses from each target illumination. (c) Compute the SNR improvement due to post-detection integration (assume 100% efficiency). (d) Find the number of false alarms per minute for a probability of false alarm $P_{fa} = 10^{-6}$.

7.11. Using the equation

$$P_D = 1 - e^{-SNR} \int_{P_{fa}}^{1} I_0(\sqrt{-4SNR \ln u})du$$

calculate P_D when $SNR = 10dB$ and $P_{fa} = 0.01$. Perform the integration numerically.

7.12. Write a MATLAB program to compute the CA-CFAR threshold value. Use similar approach to that used in the case of a fixed threshold.

7.13. A certain radar has the following specifications: single pulse SNR corresponding to a reference range $R_0 = 200Km$ is $10dB$. The probability of detection at this range is $P_D = 0.95$. Assume a Swerling I type target. Use the radar equation to compute the required pulse widths at ranges $R = 220Km, 250Km, 175Km$, so that the probability of detection is maintained.

7.14. Repeat Problem 7.14 for Swerling IV type target.

7.15. Utilizing the MATLAB functions presented in this chapter, plot the actual value for the improvement factor versus the number of integrated pulses. Pick three different values for the probability of false alarm.

7.16. Develop a MATLAB program to calculate the cumulative probability of detection.

7.17. A certain radar has the following parameters: Peak power $P_t = 500KW$, total losses $L = 12dB$, operating frequency $f_o = 5.6GHZ$, PRF $f_r = 2KHz$, pulse width $\tau = 0.5\mu s$, antenna beamwidth $\theta_{az} = 2°$ and $\theta_{el} = 7°$, noise figure $F = 6dB$, scan time $T_{sc} = 2s$. The radar can experience one false alarm per scan. (a) What is the probability of false alarm? Assume that the radar searches a minimum range of 10 Km to its maximum unambiguous range. (b) Plot the detection range versus RCS in dBsm. The detection range is defined as the range at which the single scan probability of detection is equal to 0.94. Generate curves for Swerling I, II, III, and IV type targets. (c) Repeat part (b) above when noncoherent integration is used.

7.18. A certain circularly scanning radar with a fan beam has a rotation rate of 3 seconds per revolution. The azimuth beamwidth is 3 degrees and the radar uses a PRI of 600 microseconds. The radar pulse width is 2 microseconds and the radar searches a range window that extends from 15 Km to 100 Km. It is desired that the false alarm rate not be higher than two false alarms per revolution. What is the required probability of false alarm? What is the minimum SNR so that minimum probability of false alarm can be maintained?

7.19. Derive Eq(7.63).