Chapter 4 Radar Detection

4.1. 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. 4.1. The input signal to the receiver is composed of the radar echo signal $s(t)$ and additive zero mean white Gaussian noise $n(t)$, with variance ψ^2. The input noise is assumed to be spatially incoherent and uncorrelated with the signal.

The output of the band pass IF filter is the signal $v(t)$, which can be written as

$$
\begin{aligned}
v(t) &= v_I(t)\cos\omega_0 t + v_Q(t)\sin\omega_0 = r(t)\cos(\omega_0 t - \varphi(t)) \\
v_I(t) &= r(t)\cos\varphi(t) \\
v_Q(t) &= r(t)\sin\varphi(t)
\end{aligned}
\tag{4.1}
$$

where $\omega_0 = 2\pi f_0$ is the radar operating frequency, $r(t)$ is the envelope of $v(t)$, the phase is $\varphi(t) = \operatorname{atan}(v_Q/v_I)$, and the subscripts I, 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

$$
\begin{aligned}
&s(t) + n(t) > V_T \qquad \textit{Detection} \\
&n(t) > V_T \qquad \textit{False alarm}
\end{aligned}
$$

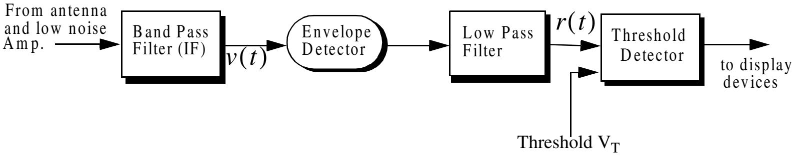


Figure 4.1. Simplified block diagram of an envelope detector and threshold receiver.

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 first case are

$$\begin{aligned} v_I(t) &= n_I(t) \\ v_Q(t) &= n_Q(t) \end{aligned} \tag{4.2}$$

and for the second case,

$$\begin{aligned} v_I(t) = A + n_I(t) = r(t)\cos\varphi(t) \Rightarrow n_I(t) &= r(t)\cos\varphi(t) - A \\ v_Q(t) = n_Q(t) &= r(t)\sin\varphi(t) \end{aligned} \tag{4.3}$$

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

$$\begin{aligned} f(n_I, n_Q) &= \frac{1}{2\pi\psi^2}\exp\left(-\frac{n_I^2 + n_Q^2}{2\psi^2}\right) \\ &= \frac{1}{2\pi\psi^2}\exp\left(-\frac{(r\cos\varphi - A)^2 + (r\sin\varphi)^2}{2\psi^2}\right) \end{aligned} \tag{4.4}$$

The *pdfs* of the random variables $r(t)$ and $\varphi(t)$, respectively, represent the modulus and phase of $v(t)$. The joint *pdf* for the two random variables $r(t);\varphi(t)$ is given by

$$f(r, \varphi) = f(n_I, n_Q)|J| \tag{4.5}$$

where $[J]$ is a matrix of derivatives defined by

$$[J] = \begin{bmatrix} \frac{\partial n_I}{\partial r} & \frac{\partial n_I}{\partial \varphi} \\ \frac{\partial n_Q}{\partial r} & \frac{\partial n_Q}{\partial \varphi} \end{bmatrix} = \begin{bmatrix} \cos\varphi & -r\sin\varphi \\ \sin\varphi & r\cos\varphi \end{bmatrix} \quad \textbf{(4.6)}$$

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

$$|J| = r(t) \quad \textbf{(4.7)}$$

Substituting Eqs. (4.4) and (4.7) into Eq. (4.5) and collecting terms yield

$$f(r, \varphi) = \frac{r}{2\pi\psi^2}\exp\left(-\frac{r^2 + A^2}{2\psi^2}\right)\exp\left(\frac{rA\cos\varphi}{\psi^2}\right) \quad \textbf{(4.8)}$$

The *pdf* for r alone is obtained by integrating Eq. (4.8) over φ

$$f(r) = \int_0^{2\pi} f(r, \varphi)d\varphi = \frac{r}{\psi^2}\exp\left(-\frac{r^2 + A^2}{2\psi^2}\right) \frac{1}{2\pi}\int_0^{2\pi} \exp\left(\frac{rA\cos\varphi}{\psi^2}\right)d\varphi \quad \textbf{(4.9)}$$

where the integral inside Eq. (4.9) is known as the modified Bessel function of zero order,

$$I_0(\beta) = \frac{1}{2\pi}\int_0^{2\pi} e^{\beta\cos\theta}\, d\theta \quad \textbf{(4.10)}$$

Thus,

$$f(r) = \frac{r}{\psi^2} I_0\left(\frac{rA}{\psi^2}\right)\exp\left(-\frac{r^2 + A^2}{2\psi^2}\right) \quad \textbf{(4.11)}$$

which is the Rice probability density function. If $A/\psi^2 = 0$ (noise alone), then Eq. (4.11) becomes the Rayleigh probability density function

$$f(r) = \frac{r}{\psi^2}\exp\left(-\frac{r^2}{2\psi^2}\right) \quad \textbf{(4.12)}$$

Also, when (A/ψ^2) is very large, Eq. (4.11) becomes a Gaussian probability density function of mean A and variance ψ^2:

$$f(r) \approx \frac{1}{\sqrt{2\pi\psi^2}} \exp\left(-\frac{(r-A)^2}{2\psi^2}\right) \tag{4.13}$$

Fig. 4.2 shows plots for the Rayleigh and Gaussian densities.

The density function for the random variable φ is obtained from

$$f(\varphi) = \int_0^r f(r, \varphi) \ dr \tag{4.14}$$

While the detailed derivation is left as an exercise, the result of Eq. (4.14) is

$$f(\varphi) = \frac{1}{2\pi} \ \exp\left(\frac{-A^2}{2\psi^2}\right) + \frac{A\cos\varphi}{\sqrt{2\pi\psi^2}} \ \exp\left(\frac{-(A\sin\varphi)^2}{2\psi^2}\right) \ F\left(\frac{A\cos\varphi}{\psi}\right) \tag{4.15}$$

where

$$F(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} \ e^{-\zeta^2/2} \ d\xi \tag{4.16}$$

The function $F(x)$ can be found tabulated in most mathematical formulas and tables reference books. Note that for the case of noise alone ($A = 0$), Eq. (4.15) collapses to a uniform *pdf* over the interval $\{0, 2\pi\}$.

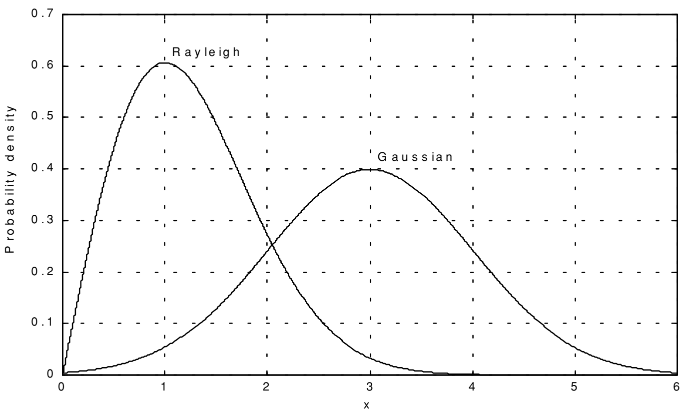


Figure 4.2. Gaussian and Rayleigh probability densities.

One excellent approximation for the function $F(x)$ is

$$F(x) = 1 - \left(\frac{1}{0.661x + 0.339\sqrt{x^2 + 5.51}}\right)\frac{1}{\sqrt{2\pi}}e^{-x^2/2} \qquad x \ge 0 \tag{4.17}$$

and for negative values of x

$$F(-x) = 1 - F(x) \tag{4.18}$$

MATLAB Function "que_func.m"

The function *"que_func.m"* computes $F(x)$ using Eqs. (4.17) and (4.18) and is given in Listing 4.1 in Section 4.10. The syntax is as follows:

fofx = que_func (x)

4.2. *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,

$$P_{fa} = \int_{V_T}^{\infty} \frac{r}{\psi^2}\exp\left(-\frac{r^2}{2\psi^2}\right) dr = \exp\left(\frac{-V_T^2}{2\psi^2}\right) \tag{4.19a}$$

$$V_T = \sqrt{2\psi^2 \ln\left(\frac{1}{P_{fa}}\right)} \tag{4.19b}$$

Fig. 4.3 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

$$T_{fa} = \frac{t_{int}}{P_{fa}} \tag{4.20}$$

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}, then by substituting Eq. (4.19) into Eq. (4.20) we can write T_{fa} as

$$T_{fa} = \frac{1}{B}\exp\left(\frac{V_T^2}{2\psi^2}\right) \tag{4.21}$$

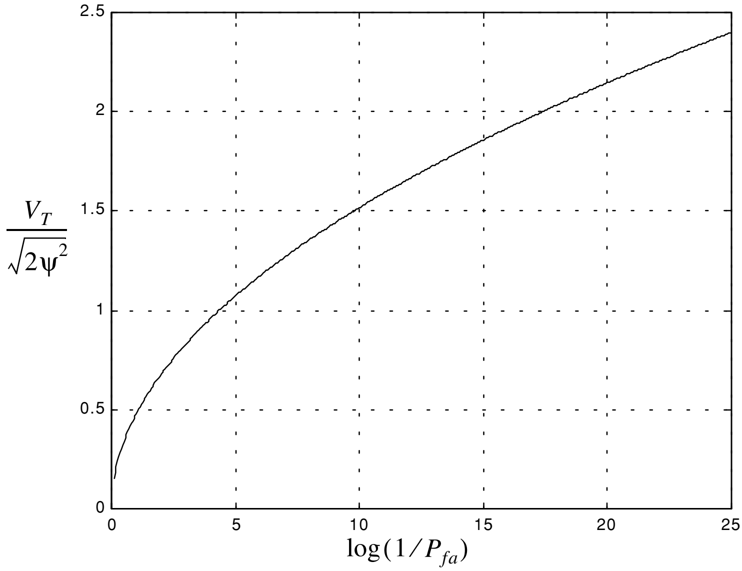


Figure 4.3. Normalized detection threshold versus probability of false alarm.

Minimizing T_{fa} means increasing the threshold value, and as a result the radar maximum detection range is decreased. Therefore, the choice of an acceptable value for T_{fa} becomes a compromise depending on the radar mode of operation. The false alarm number n_{fa} was defined by Marcum (see bibliography) as the reciprocal of P_{fa}. Using Marcum's definition of the false alarm number, the probability of false alarm is given by $P_{fa} \approx \ln(2)(n_p/n_{fa})$, where $n_p > 1$ is the number of pulses and $P_{fa} < 0.007$.

4.3. 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,

$$P_D = \int_{V_T}^{\infty} \frac{r}{\psi^2} I_0\left(\frac{rA}{\psi^2}\right) \exp\left(-\frac{r^2 + A^2}{2\psi^2}\right) dr \tag{4.22}$$

If we assume that the radar signal is a sine waveform with amplitude A, then its power is $A^2/2$. Now, by using $SNR = A^2/2\psi^2$ (single-pulse SNR) and $(V_T^2/2\psi^2) = \ln(1/P_{fa})$, then Eq. (4.22) can be rewritten as

$$P_D = \int_{\sqrt{2\psi^2 \ln(1/p_{fa})}}^{\infty} \frac{r}{\psi^2} I_0\left(\frac{rA}{\psi^2}\right) \exp\left(-\frac{r^2 + A^2}{2\psi^2}\right) dr = Q\left[\sqrt{\frac{A^2}{\psi^2}}, \sqrt{2\ln\left(\frac{1}{P_{fa}}\right)}\right] \quad \textbf{(4.23)}$$

$$Q[\alpha, \beta] = \int_{\beta}^{\infty} \zeta I_0(\alpha\zeta) e^{-(\zeta^2 + \alpha^2)/2} \, d\zeta \quad \textbf{(4.24)}$$

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. (4.24) can be approximated by

$$P_D \approx F\left(\frac{A}{\psi} - \sqrt{2\ln\left(\frac{1}{P_{fa}}\right)}\right) \quad \textbf{(4.25)}$$

where $F(x)$ is given by Eq. (4.16).

Many approximations for computing Eq. (4.23) can be found throughout the literature. One very accurate approximation presented by North (see bibliography) is given by

$$P_D \approx 0.5 \times erfc(\sqrt{-\ln P_{fa}} - \sqrt{SNR + 0.5}) \quad \textbf{(4.26)}$$

where the complementary error function is

$$erfc(z) = 1 - \frac{2}{\sqrt{\pi}} \int_0^z e^{-\nu^2} d\nu \quad \textbf{(4.27)}$$

Table 4.1 gives samples of the single pulse SNR corresponding to few values of P_D and P_{fa}, using Eq. (4.26). For example, if $P_D = 0.99$ and $P_{fa} = 10^{-10}$, then the minimum single pulse SNR required to accomplish this combination of P_D and P_{fa} is $SNR = 16.12dB$.

MATLAB Function "marcumsq.m"

The integral given in Eq. (4.23) 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:

$$Q[a, b] = \begin{cases} \dfrac{\alpha_n}{2\beta_n} \exp\left(\dfrac{(a-b)^2}{2}\right) & a < b \\[2ex] 1 - \left(\dfrac{\alpha_n}{2\beta_n} \exp\left(\dfrac{(a-b)^2}{2}\right) \right. & \left. a \geq b \right) \end{cases} \quad \textbf{(4.28)}$$

1. Parl, S., A New Method of Calculating the Generalized Q Function, *IEEE Trans. Information Theory,* Vol. IT-26, No. 1, January 1980, pp. 121-124.

TABLE 4.1. Single pulse SNR (dB).

P_D	P_{fa} 10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}
.1	4.00	6.19	7.85	8.95	9.94	10.44	11.12	11.62	12.16	12.65
.2	5.57	7.35	8.75	9.81	10.50	11.19	11.87	12.31	12.85	13.25
.3	6.75	8.25	9.50	10.44	11.10	11.75	12.37	12.81	13.25	13.65
.4	7.87	8.85	10.18	10.87	11.56	12.18	12.75	13.25	13.65	14.00
.5	8.44	9.45	10.62	11.25	11.95	12.60	13.11	13.52	14.00	14.35
.6	8.75	9.95	11.00	11.75	12.37	12.88	13.50	13.87	14.25	14.62
.7	9.56	10.50	11.50	12.31	12.75	13.31	13.87	14.20	14.59	14.95
.8	10.18	11.12	12.05	12.62	13.25	13.75	14.25	14.55	14.87	15.25
.9	10.95	11.85	12.65	13.31	13.85	14.25	14.62	15.00	15.45	15.75
.95	11.50	12.40	13.12	13.65	14.25	14.64	15.10	15.45	15.75	16.12
.98	12.18	13.00	13.62	14.25	14.62	15.12	15.47	15.85	16.25	16.50
.99	12.62	13.37	14.05	14.50	15.00	15.38	15.75	16.12	16.47	16.75
.995	12.85	13.65	14.31	14.75	15.25	15.71	16.06	16.37	16.65	17.00
.998	13.31	14.05	14.62	15.06	15.53	16.05	16.37	16.7	16.89	17.25
.999	13.62	14.25	14.88	15.25	15.85	16.13	16.50	16.85	17.12	17.44
.9995	13.84	14.50	15.06	15.55	15.99	16.35	16.70	16.98	17.35	17.55
.9999	14.38	14.94	15.44	16.12	16.50	16.87	17.12	17.35	17.62	17.87

$$\alpha_n = d_n + \frac{2n}{ab}\alpha_{n-1} + \alpha_{n-2} \tag{4.29}$$

$$\beta_n = 1 + \frac{2n}{ab}\beta_{n-1} + \beta_{n-2} \tag{4.30}$$

$$d_{n+1} = d_n d_1 \tag{4.31}$$

$$\alpha_0 = \begin{Bmatrix} 1 & a < b \\ 0 & a \ge b \end{Bmatrix} \tag{4.32}$$

$$d_1 = \begin{Bmatrix} a/b & a < b \\ b/a & a \ge b \end{Bmatrix} \tag{4.33}$$

$\alpha_{-1} = 0.0$, $\beta_0 = 0.5$, and $\beta_{-1} = 0$. The recursive Eqs. (4.29) through (4.31) are computed continuously until $\beta_n > 10^p$ for some value $p \ge 3$. The accuracy of the algorithm is enhanced as the value of p is increased. The MATLAB function *"marcumsq.m"* given in Listing 4.2 in Section 4.10 implements Parl's algorithm to compute the probability of detection defined in Eq. (4.23). The syntax is as follows:

$$Pd = marcumsq\ (alpha, beta)$$

where *alpha* and *beta* are from Eq. (4.24). Fig. 4.4 shows plots of the probability of detection, P_D, versus the single pulse SNR, with the P_{fa} as a parameter. This figure can be reproduced using the MATLAB program *"prob_snr1.m"* given in Listing 4.3 in Section 4.10. This program uses the function *"marcumsq.m"*.

Example 4.1: A pulsed radar has the following specification: time of false alarm $T_{fa} = 16.67$ minutes; probability of detection $P_D = 0.9$ and bandwidth $B = 1\ GHz$. Find the radar integration time t_{int}, the probability of false alarm P_{fa}, and the SNR of a single pulse.

Solution:

$$t_{int} = \frac{1}{B} = \frac{1}{10^9} = 1n\sec$$

$$P_{fa} = \frac{1}{T_{fa}B} = \frac{1}{10^9 \times 16.67 \times 60} \approx 10^{-12}$$

and from Table 4.1 or from Fig. 4.4, we read

$$(SNR)_1 \approx 15.75dB.$$

4.4. Pulse Integration

When a target is illuminated by the radar beam it normally reflects numerous pulses. The radar probability of detection is normally enhanced by summing all (or most) of the returned pulses. The process of adding radar echoes 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 pre-detection integration. Coherent integration preserves the phase relationship between the received pulses, thus a build up in the signal amplitude is achieved. Alternatively, pulse integration performed after the envelope detector (where the phase relation is destroyed) is called non-coherent or post-detection integration.

4.4.1. Coherent Integration

In coherent integration, if a perfect integrator is used (100% efficiency), then integrating n_p pulses would improve the SNR by the same factor. Otherwise, integration loss occurs which is always the case for non-coherent integration. In order to demonstrate this signal buildup, consider the case where the radar return signal contains both signal plus additive noise. The m^{th} pulse is

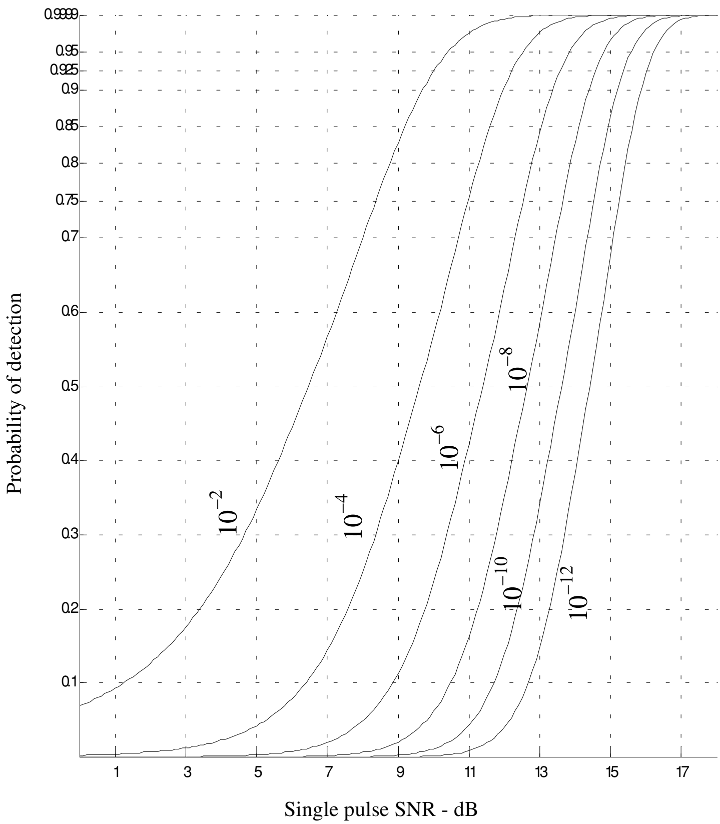


Figure 4.4. Probability of detection versus single pulse SNR, for several values of P_{fa}.

$$y_m(t) = s(t) + n_m(t) \tag{4.34}$$

where $s(t)$ is the radar return of interest and $n_m(t)$ is white uncorrelated additive noise signal. Coherent integration of n_p pulses yields

$$z(t) = \frac{1}{n_p}\sum_{m=1}^{n_p} y_m(t) = \sum_{m=1}^{n_p}\frac{1}{n_p}[s(t) + n_m(t)] = s(t) + \sum_{m=1}^{n_p}\frac{1}{n_p}n_m(t) \tag{4.35}$$

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

$$\psi_{nz}^2 = E\left[\left(\sum_{m=1}^{n_p}\frac{1}{n_p}n_m(t)\right)\left(\sum_{l=1}^{n_p}\frac{1}{n_p}n_l(t)\right)^*\right] \qquad \textbf{(4.36)}$$

where $E[\ ]$ is the expected value operator. It follows that

$$\psi_{nz}^2 = \frac{1}{n_p^2}\sum_{m,l=1}^{n_p} E[n_m(t)n_l^*(t)] = \frac{1}{n_p^2}\sum_{m,l=1}^{n_p}\psi_{ny}^2\delta_{ml} = \frac{1}{n_p}\psi_{ny}^2 \qquad \textbf{(4.37)}$$

where ψ_{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. (4.35) and (4.37) 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.

Denote the single pulse SNR required to produce a given probability of detection as $(SNR)_1$. Also, denote $(SNR)_{n_p}$ as the SNR required to produce the same probability of detection when n_p pulses are integrated. It follows that

$$(SNR)_{n_p} = \frac{1}{n_p}(SNR)_1 \qquad \textbf{(4.38)}$$

The requirements of remembering the phase of each transmitted pulse as well as maintaining coherency during propagation is very costly and challenging to achieve. In practice, most radar systems utilize non-coherent integration.

4.4.2. Non-Coherent Integration

Non-coherent integration is often implemented after the envelope detector, also known as the quadratic detector. A block diagram of radar receiver utilizing a square law detector and non-coherent integration is illustrated in Fig. 4.5. In practice, the square law detector is normally used as an approximation to the optimum receiver.

The *pdf* for the signal $r(t)$ was derived earlier and it is given in Eq. (4.11). Define a new dimensionless variable y as

$$y_n = r_n/\psi \qquad \textbf{(4.39)}$$

and also define

$$\Re_p = \frac{A^2}{\psi^2} = 2SNR \qquad \textbf{(4.40)}$$

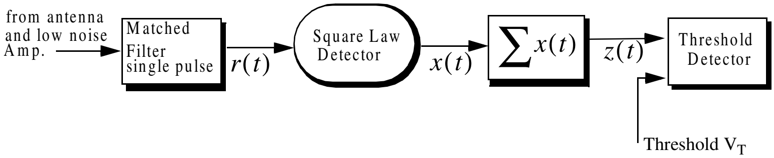


Figure 4.5. Simplified block diagram of a square law detector and non-coherent integration.

It follows that the *pdf* for the new variable is then given by

$$f(y_n) = f(r_n)\left|\frac{dr_n}{dy_n}\right| = y_n \ I_0(y_n\sqrt{\Re_p}) \ \exp\left(\frac{-(y_n^2+\Re_p)}{2}\right) \tag{4.41}$$

The output of a square law detector for the n^{th} pulse is proportional to the square of its input, which, after the change of variable in Eq. (4.39), is proportional to y_n. Thus, it is convenient to define a new change variable,

$$x_n = \frac{1}{2}y_n^2 \tag{4.42}$$

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

$$f(x_n) = f(y_n)\left|\frac{dy_n}{dx_n}\right| = \exp\left(-\left(x_n+\frac{\Re_p}{2}\right)\right)I_0(\sqrt{2x_n\Re_p}) \tag{4.43}$$

Non-coherent integration of n_p pulses is implemented as

$$z = \sum_{n=1}^{n_p} x_n \tag{4.44}$$

Since the random variables x_n are independent, the *pdf* for the variable z is

$$f(z) = f(x_1)\bullet f(x_2)\bullet \ldots \bullet f(x_{n_p}) \tag{4.45}$$

the operator $\bullet$ symbolically indicates convolution. The characteristic functions for the individual *pdf*s can then be used to compute the joint *pdf* in Eq. (4.45). The details of this development are left as an exercise. The result is

$$f(z) = \left(\frac{2z}{n_p\Re_p}\right)^{(n_p-1)/2} \exp\left(-z-\frac{1}{2}n_p\Re_p\right)I_{n_p-1}(\sqrt{2n_pz\Re_p}) \tag{4.46}$$

where $I_{n_p - 1}$ is the modified Bessel function of order $n_p - 1$. Therefore, the probability of detection is obtained by integrating $f(z)$ from the threshold value to infinity. Alternatively, the probability of false alarm is obtained by letting $\Re_p$ be zero and integrating the *pdf* from the threshold value to infinity. Closed form solutions to these integrals are not easily available. Therefore, numerical techniques are often utilized to generate tables for the probability of detection.

The non-coherent integration efficiency $E(n_p)$ is defined as

$$E(n_p) = \frac{(SNR)_1}{n_p (SNR)_{n_p}} \le 1 \qquad \textbf{(4.47)}$$

The integration improvement factor $I(n_p)$ for a specific P_{fa} is defined as the ratio of $(SNR)_1$ to $(SNR)_{n_p}$

$$I(n_p) = \frac{(SNR)_1}{(SNR)_{n_p}} = n_p E(n_p) \le n_p \qquad \textbf{(4.48)}$$

Note that $(SNR)_{n_p}$ corresponds to the SNR needed to produce the same P_D as in the case of a single pulse when n_p pulses are used. It follows that $(SNR)_{n_p} < (SNR)_1$.

An empirically derived expression for the improvement factor that is accurate within $0.8dB$ is reported in Peebles[1] as

$$[I(n_p)]_{dB} = 6.79(1 + 0.235 P_D)\left(1 + \frac{\log(1/P_{fa})}{46.6}\right)\log(n_p) \qquad \textbf{(4.49)}$$

$$(1 - 0.140\log(n_p) + 0.018310(\log n_p)^2)$$

Fig. 4.6 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. (4.49). This plot can be reproduced using the MATLAB program *"fig4_5.m"* given in Listing 4.4 in Section 4.10.

Example 4.2: Consider the same radar defined in Example 4.1. Assume non-coherent integration of 10 pulses. Find the reduction in the SNR.

Solution: The integration improvement factor is calculated using the function "improv_fac.m". It is $I(10) \cong 9.20dB$*, and from Eq. (4.48) we get*

$$(SNR)_{n_p} = \frac{(SNR)_1}{I(n_p)} \Rightarrow (SNR)_{n_p} = 15.75 - 9.20 = 6.55dB$$

---

1. Peebles Jr., P. Z., *Radar Principles*, John Wiley & Sons, Inc., 1998.

Thus, non-coherent integration of 10 pulses where $(SNR)_{10} = 6.55 dB$ *provides the same detection performance as* $(SNR)_1 = 15.75 dB$ *of a single pulse and no integration.*

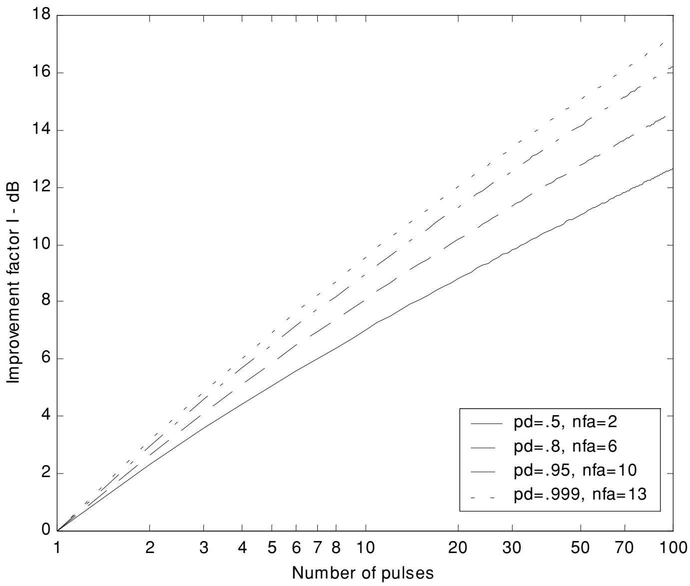


Figure 4.6. Improvement factor versus number of pulses (non-coherent integration). These plots were generated using the empirical approximation in Eq. (4.49).

MATLAB Function "improv_fac.m"

The function *"improv_fac.m"* calculates the improvement factor using Eq. (4.49). It is given in Listing 4.5 in Section 4.10. 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 alarm*	*none*	*input*
pd	*probability of detection*	*none*	*input*
impr_of_np	*improvement factor*	*output*	*dB*

4.5. Detection of Fluctuating Targets

So far when we addressed the probability of detection, we assumed a constant target cross section (non-fluctuating target). However, when target scintillation is present, the probability of detection decreases, or equivalently the SNR is reduced.

4.5.1. Detection Probability Density Function

The probability density functions for fluctuating targets were given in Chapter 2. And for convenience, they are repeated here as Eqs. (4.50) and (4.51):

$$f(A) = \frac{1}{A_{av}} \exp\left(-\frac{A}{A_{av}}\right) \qquad A \geq 0 \tag{4.50}$$

for Swerling I and II type targets, and

$$f(A) = \frac{4A}{A_{av}^2} \exp\left(-\frac{2A}{A_{av}}\right) \qquad A \geq 0 \tag{4.51}$$

for Swerling III and IV type targets, where A_{av} denotes the average RCS over all target fluctuations.

The probability of detection for a scintillating target is computed in a similar fashion to Eq. (4.22), except in this case $f(r)$ is replaced by the conditional *pdf* $f(r/A)$. Performing the analysis for the general case (i.e., using Eq. (4.46)) yields

$$f(z/A) = \left(\frac{2z}{n_p A^2/\psi^2}\right)^{(n_p-1)/2} \exp\left(-z - \frac{1}{2} n_p \frac{A^2}{\psi^2}\right) I_{n_p-1}\left(\sqrt{2n_p z \frac{A^2}{\psi^2}}\right) \tag{4.52}$$

To obtain $f(z)$ use the relations

$$f(z, A) = f(z/A) f(A) \tag{4.53}$$

$$f(z) = \int f(z, A)\, dA \tag{4.54}$$

Finally, using Eq. (4.54) in Eq. (4.53) produces

$$f(z) = \int f(z/A) f(A)\, dA \tag{4.55}$$

where $f(z/A)$ is defined in Eq. (4.52) and $f(A)$ is in either Eq. (4.50) or (4.51). The probability of detection is obtained by integrating the *pdf* derived from Eq. (4.55) from the threshold value to infinity. Performing the integration in Eq. (4.55) leads to the incomplete Gamma function.

4.5.2. Threshold Selection

In practice, the detection threshold, V_T, is found from the probability of false alarm P_{fa}. DiFranco and Rubin[1] give a general form relating the threshold and P_{fa} for any number of pulses and non-coherent integration,

$$P_{fa} = 1 - \Gamma_I\left(\frac{V_T}{\sqrt{n_p}}, n_p - 1\right) \tag{4.56}$$

where Γ_I is used to denote the incomplete Gamma function, and it is given by

$$\Gamma_I\left(\frac{V_T}{\sqrt{n_p}}, n_p - 1\right) = \int_0^{V_T/\sqrt{n_p}} \frac{e^{-\gamma}\ \gamma^{n_p - 1 - 1}}{(n_p - 1 - 1)!}\ d\gamma \tag{4.57}$$

For our purposes, the incomplete Gamma function can be approximated by

$$\Gamma_I\left(\frac{V_T}{\sqrt{n_p}}, n_p - 1\right) = 1 - \frac{V_T^{n_p - 1}\ e^{-V_T}}{(n_p - 1)!}\left[1 + \frac{n_p - 1}{V_T} + \frac{(n_P - 1)(n_p - 2)}{V_T^2} + \ldots + \frac{(n_p - 1)!}{V_T^{n_p - 1}}\right] \tag{4.58}$$

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

$$V_{T,m} = V_{T,m-1} - \frac{G(V_{T,m-1})}{G'(V_{T,m-1})} \qquad ; \ m = 1, 2, 3, \ldots \tag{4.59}$$

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

$$G(V_{T,m}) = (0.5)^{n_P/n_{fa}} - \Gamma_I(V_T, n_p) \tag{4.60}$$

$$G'(V_{T,m}) = -\frac{e^{-V_T}\ V_T^{n_p - 1}}{(n_p - 1)!} \tag{4.61}$$

The initial value for the recursion is

$$V_{T,0} = n_p - \sqrt{n_p} + 2.3\ \sqrt{-\log P_{fa}}\ \left(\sqrt{-\log P_{fa}} + \sqrt{n_p}\ - 1\right) \tag{4.62}$$

---

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

MATLAB Function "incomplete_gamma.m"

In general, the incomplete Gamma function for some integer N is

$$\Gamma_I(x, N) = \int_0^x \frac{e^{-v} v^{N-1}}{(N-1)!} dv \tag{4.63}$$

The function *"incomplete_gamma.m"* implements Eq. (4.63). It is given in Listing 4.6 in Section 4.10. The syntax for this function is as follows:

[value] = incomplete_gamma (x, N)

where

Symbol	Description	Units	Status
x	*variable input to* $\Gamma_I(x, N)$	*units of x*	*input*
N	*variable input to* $\Gamma_I(x, N)$	*none / integer*	*input*
value	$\Gamma_I(x, N)$	*none*	*output*

Fig. 4.7 shows the incomplete Gamma function for $N = 1, 3, 10$. Note that the limiting values for the incomplete Gamma function are

$$\Gamma_I(0, N) = 0 \qquad \Gamma_I(\infty, N) = 1 \tag{4.64}$$

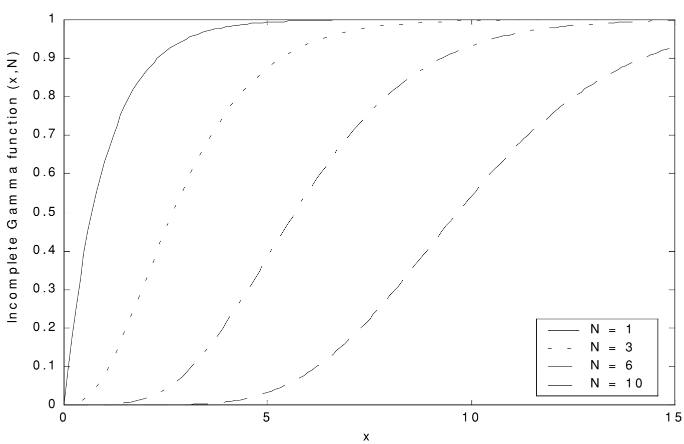


Figure 4.7. The incomplete Gamma function for four values of N.

MATLAB Function "threshold.m"

The function *"threshold.m"* calculates the threshold using the recursive formula used in the Newton-Raphson method. It is given in Listing 4.7 in Section 4.10. The syntax is as follows:

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

where

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

Fig. 4.8 shows plots for the threshold value versus the number of integrated pulses for several values of n_{fa}; remember that $P_{fa} \approx \ln(2)(n_p/n_{fa})$.

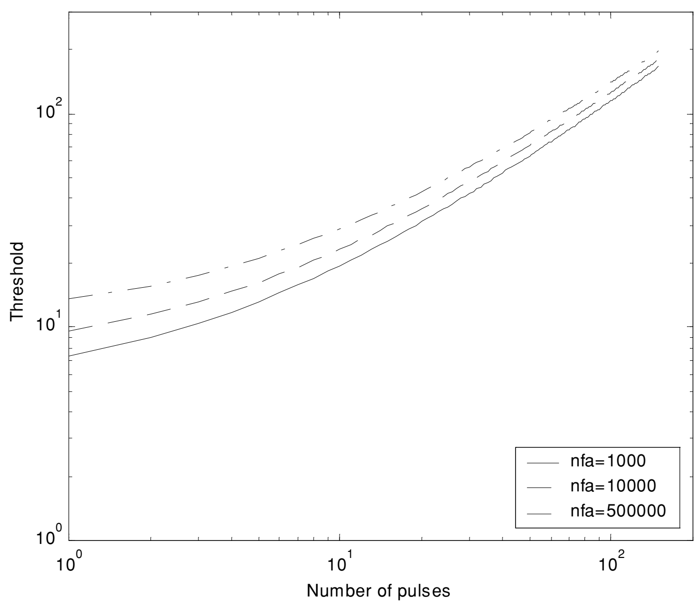


Figure 4.8. Threshold V_T versus n_p for several values of n_{fa}.

4.6. Probability of Detection Calculation

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

$$(SNR)_{R_0} = \frac{P_t G^2 \lambda^2 \sigma}{(4\pi)^3 k T_0 B F L R_0^4} = 1 \tag{4.65}$$

The single pulse SNR at any range R is

$$SNR = \frac{P_t G^2 \lambda^2 \sigma}{(4\pi)^3 k T_0 B F L R^4} \tag{4.66}$$

Dividing Eq. (4.66) by Eq. (4.65) yields

$$\frac{SNR}{(SNR)_{R_0}} = \left(\frac{R_0}{R}\right)^4 \tag{4.67}$$

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

$$(SNR)_{dB} = 40\log\left(\frac{R_0}{R}\right) \tag{4.68}$$

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

4.6.1. Detection of Swerling V Targets

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

$$P_{fa} = 1 - (P_{50})^{n_p/n_{fa}} \approx \ln(2)(n_p/n_{fa}) \tag{4.69}$$

The single pulse probability of detection for non-fluctuating targets is given in Eq. (4.23). When $n_p > 1$, the probability of detection is computed using the Gram-Charlier series. In this case, the probability of detection is

$$P_D \cong \frac{erfc(V/\sqrt{2})}{2} - \frac{e^{-V^2/2}}{\sqrt{2\pi}}[C_3(V^2 - 1) + C_4 V(3 - V^2) \tag{4.70}$$
$$- C_6 V(V^4 - 10V^2 + 15)]$$

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

$$V = \frac{V_T - n_p(1 + SNR)}{\varpi} \tag{4.71}$$

In general, values for C_3, C_4, C_6, and ϖ vary depending on the target fluctuation type. In the case of Swerling V targets, they are

$$C_3 = -\frac{SNR + 1/3}{\sqrt{n_p}(2SNR + 1)^{1.5}} \tag{4.72}$$

$$C_4 = \frac{SNR + 1/4}{n_p(2SNR + 1)^2} \tag{4.73}$$

$$C_6 = C_3^2/2 \tag{4.74}$$

$$\varpi = \sqrt{n_p(2SNR + 1)} \tag{4.75}$$

MATLAB Function "pd_swerling5.m"

The function *"pd_swerling5.m"* calculates the probability of detection for Swerling V targets using Eq. (4.70). It is given in Listing 4.8 in Section 4.10. 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*
np	*number of integrated pulses*	*none*	*input*
snr	*SNR*	*dB*	*input*
pd	*probability of detection*	*none*	*output*

Fig. 4.9 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 $I(10) \approx 8.55dB$. Observation of Fig. 4.9 shows that the ten pulse SNR is about $5.03dB$. Therefore, the single pulse SNR is about (from Eq. (4.48)) $14.5dB$, which can be read from the figure. This figure can be reproduced using MATLAB program *"fig4_9.m"*, which is part of the companion software of this book.

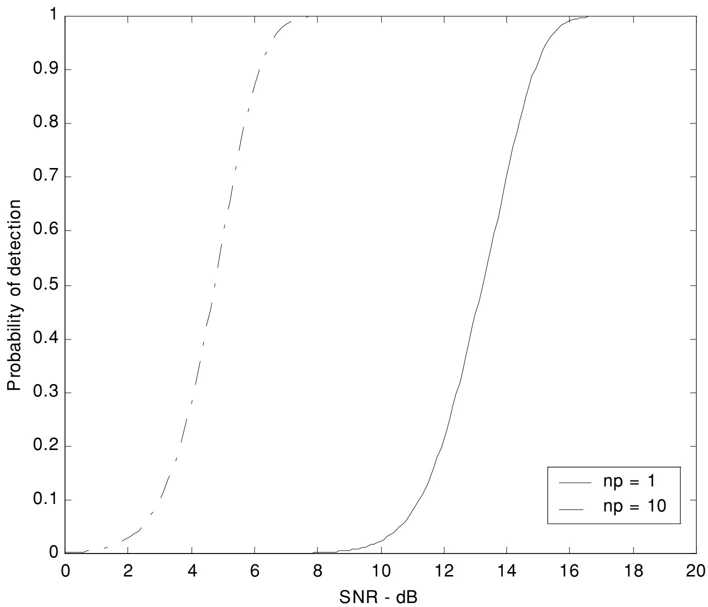


Figure 4.9. Probability of detection versus SNR, $P_{fa} = 10^{-9}$, and non-coherent integration.

4.6.2. Detection of Swerling I Targets

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

$$P_D = e^{-V_T/(1 + SNR)} \qquad ; \; n_p = 1 \tag{4.76}$$

$$P_D = 1 - \Gamma_I(V_T, n_p - 1) + \left(1 + \frac{1}{n_p SNR}\right)^{n_p - 1} \Gamma_I\left(\frac{V_T}{1 + \frac{1}{n_p SNR}}, n_p - 1\right) \times e^{-V_T/(1 + n_p SNR)} \qquad ; \; n_p > 1 \tag{4.77}$$

MATLAB Function "pd_swerling1.m"

The function *"pd_swerling1.m"* calculates the probability of detection for Swerling I type targets. It is given in Listing 4.9 in Section 4.10. 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*

Fig. 4.10 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 type fluctuating. Note that it requires more SNR, with fluctuation, to achieve the same P_D as in the case with no fluctuation. Fig. 4.11a shows a plot of the probability of detection versus SNR for $n_p = 1, 10, 50, 100$, where $P_{fa} = 10^{-6}$. Fig. 4.11b is similar to Fig. 4.11a; in this case $P_{fa} = 10^{-12}$.

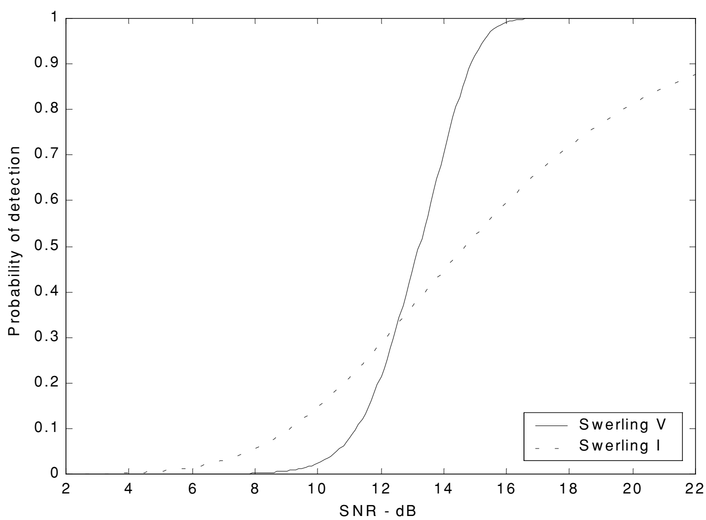


Figure 4.10. Probability of detection versus SNR, single pulse. $P_{fa} = 10^{-9}$.

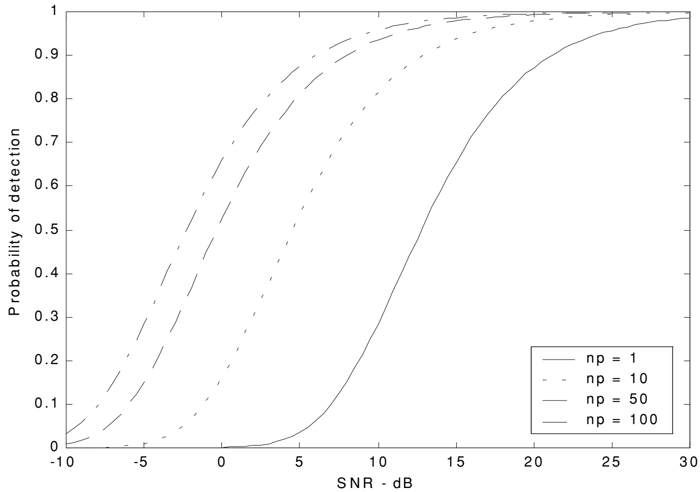


Figure 4.11a. Probability of detection versus SNR. Swerling I. $P_{fa} = 10^{-6}$.

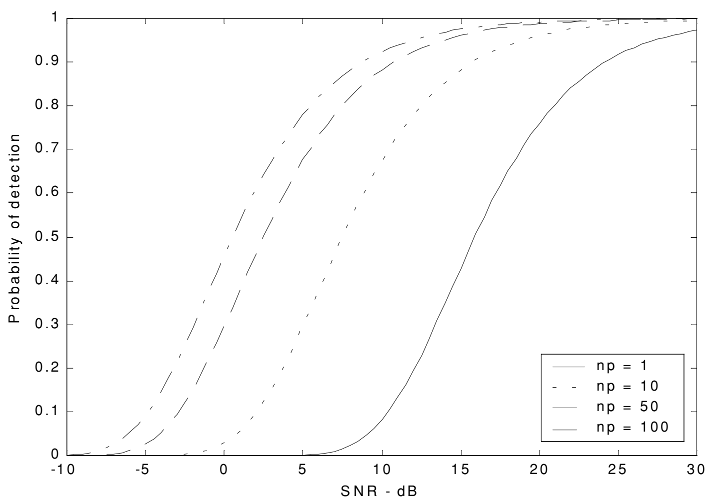


Figure 4.11b. Probability of detection versus SNR. Swerling I. $P_{fa} = 10^{-12}$.

4.6.3. Detection of Swerling II Targets

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

$$P_D = 1 - \Gamma_I\left(\frac{V_T}{(1+SNR)}\ , n_p\right) \qquad ; \ n_p \le 50 \tag{4.78}$$

For the case when $n_P > 50$ Eq. (4.70) is used to compute the probability of detection. In this case,

$$C_3 = -\frac{1}{3\sqrt{n_p}} \qquad , \ C_6 = \frac{C_3^2}{2} \tag{4.79}$$

$$C_4 = \frac{1}{4n_p} \tag{4.80}$$

$$\varpi = \sqrt{n_p}\ (1 + SNR) \tag{4.81}$$

MATLAB Function "pd_swerling2.m"

The function *"pd_swerling2.m"* calculates P_D for Swerling II type targets. It is given in Listing 4.10 in Section 4.10. 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*

Fig. 4.12 shows a plot of the probability of detection as a function of SNR for $n_p = 1, 10, 50, 100$, where $P_{fa} = 10^{-9}$.

4.6.4. Detection of Swerling III Targets

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

$$P_D = \exp\left(\frac{-V_T}{1 + n_p SNR/2}\right)\left(1 + \frac{2}{n_p SNR}\right)^{n_p - 2} \times \left(1 + \frac{V_T}{1 + n_p SNR/2} - \frac{2}{n_p SNR}(n_p - 2)\right) = K_0 \tag{4.82}$$

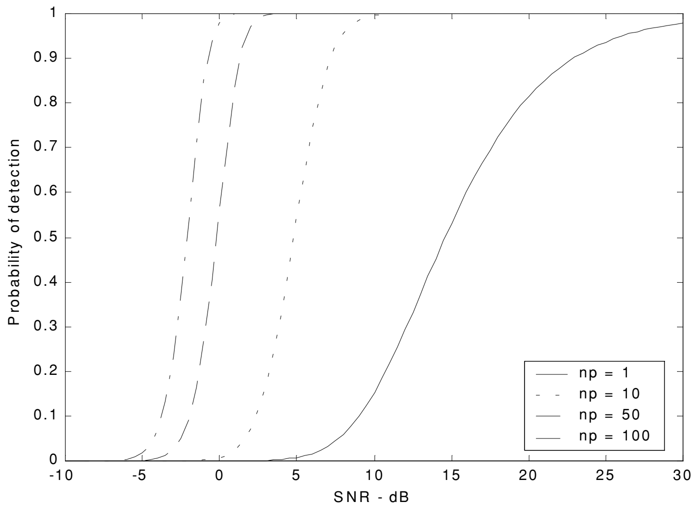


Figure 4.12. Probability of detection versus SNR. Swerling II. $P_{fa} = 10^{-9}$**.**

For $n_p > 2$ the expression is

$$P_D = \frac{V_T^{n_p - 1} \quad e^{-V_T}}{(1 + n_p SNR/2)(n_p - 2)!} + 1 - \Gamma_I(V_T, n_p - 1) + K_0 \qquad \textbf{(4.83)}$$

$$\Gamma_I\left(\frac{V_T}{1 + 2/n_p SNR}, n_p - 1\right)$$

MATLAB Function "pd_swerling3.m"

The function *"pd_swerling3.m"* calculates P_D for Swerling II type targets. It is given in Listing 4.11 in Section 4.10. 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*

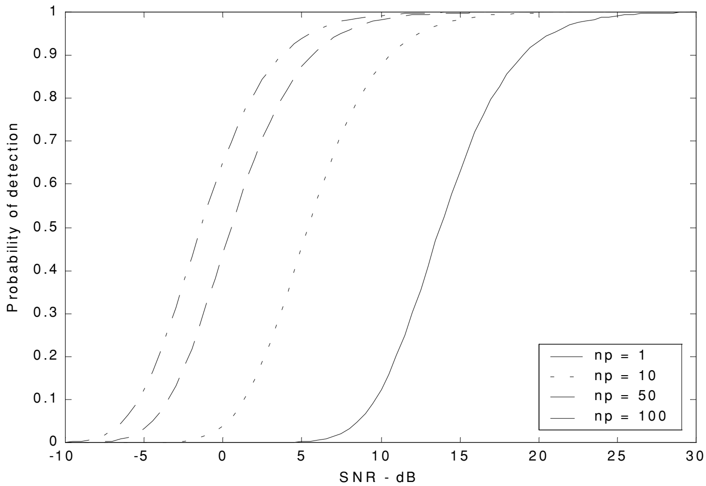


Figure 4.13. Probability of detection versus SNR. Swerling III. $P_{fa} = 10^{-9}$**.**

Fig. 4.13 shows a plot of the probability of detection as a function of SNR for $n_p = 1, 10, 50, 100$, where $P_{fa} = 10^{-9}$.

4.6.5. Detection of Swerling IV Targets

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

$$P_D = 1 - \left[\gamma_0 + \left(\frac{SNR}{2}\right)n_p\gamma_1 + \left(\frac{SNR}{2}\right)^2 \frac{n_p(n_p-1)}{2!}\gamma_2 + \ldots + \left(\frac{SNR}{2}\right)^{n_p}\gamma_{n_p}\right]\left(1 + \frac{SNR}{2}\right)^{-n_p} \quad \textbf{(4.84)}$$

where

$$\gamma_i = \Gamma_I\left(\frac{V_T}{1 + (SNR)/2}, n_p + i\right) \quad \textbf{(4.85)}$$

By using the recursive formula

$$\Gamma_I(x, i+1) = \Gamma_I(x, i) - \frac{x^i}{i!\exp(x)} \quad \textbf{(4.86)}$$

then only γ_0 needs to be calculated using Eq. (4.85) and the rest of γ_i are calculated from the following recursion:

$$\gamma_i = \gamma_{i-1} - A_i \qquad ;i > 0 \tag{4.87}$$

$$A_i = \frac{V_T/(1 + (SNR)/2)}{n_p + i - 1} A_{i-1} \qquad ;i > 1 \tag{4.88}$$

$$A_1 = \frac{(V_T/(1 + (SNR)/2))^{n_p}}{n_p! \exp(V_T/(1 + (SNR)/2))} \tag{4.89}$$

$$\gamma_0 = \Gamma_I\left(\frac{V_T}{(1 + (SNR)/2)}\ , n_p\right) \tag{4.90}$$

For the case when $n_p \geq 50$, the Gram-Charlier series and Eq. (4.70) can be used to calculate the probability of detection. In this case,

$$C_3 = \frac{1}{3\sqrt{n_p}} \frac{2\beta^3 - 1}{(2\beta^2 - 1)^{1.5}} \qquad ;\ C_6 = \frac{C_3^2}{2} \tag{4.91}$$

$$C_4 = \frac{1}{4n_p} \frac{2\beta^4 - 1}{(2\beta^2 - 1)^2} \tag{4.92}$$

$$\varpi = \sqrt{n_p(2\beta^2 - 1)} \tag{4.93}$$

$$\beta = 1 + \frac{SNR}{2} \tag{4.94}$$

MATLAB Function "pd_swerling4.m"

The function *"pd_swerling4.m"* calculates P_D for Swerling II type targets. It is given in Listing 4.12 in Section 4.10. 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*
snr	*SNR*	*dB*	*input*
pd	*probability of detection*	*none*	*output*

Fig. 4.14 shows a plot of the probability of detection as a function of SNR for $n_p = 1, 10, 50, 100$, where $P_{fa} = 10^{-9}$.

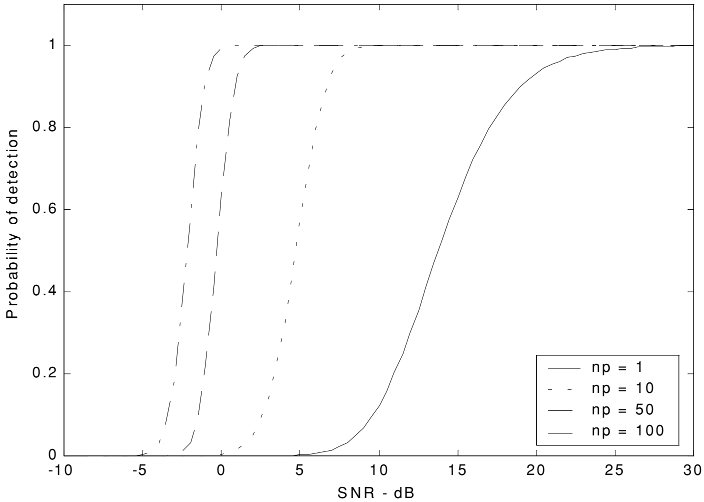


Figure 4.14. Probability of detection versus SNR. Swerling IV. $P_{fa} = 10^{-9}$.

4.7. Cumulative Probability of Detection

The cumulative probability of detection refers to detecting the target at least once by the time it is 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 also 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. 4.15) is given by

$$P_{C_n} = 1 - \prod_{i=1}^{n} (1 - P_{D_i}) \tag{4.95}$$

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.

Example 4.3: A radar detects a closing target at $R = 10Km$, *with probability of detection 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 $R = 8Km$?

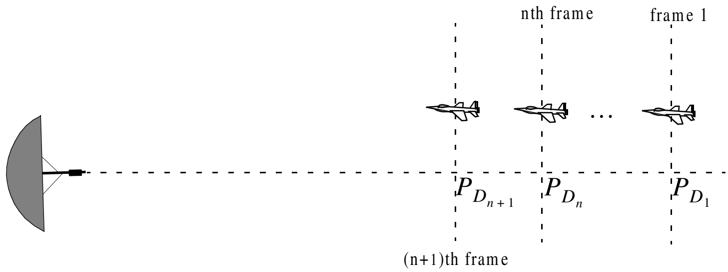


Figure 4.15. Detecting a target in many frames.

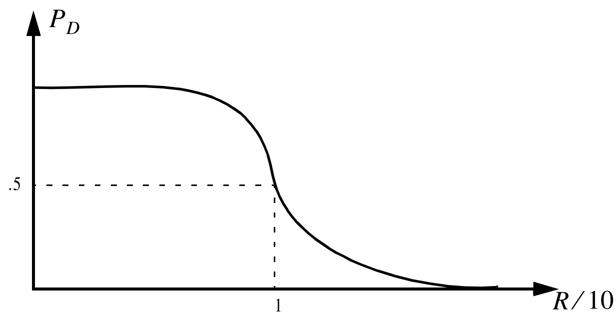
Solution: From the function "marcumsq.m" or from Table 4.1 the SNR corresponding to $P_D = 0.5$ and $P_{fa} = 10^{-7}$ is approximately 12dB. By using a similar analysis to that which led to Eq. (4.68), we can express the SNR at any range R as

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

Then with the help of 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
12	8.8	0.07
14	6.1	0.01
16	3.8	ε
20	0.01	ε

where ε is very small. Below is a sketch of P_D versus normalized range.



The cumulative probability of detection is given in Eq. (4.95), 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

$$P_{C_9} = 1 - (1 - 0.999)(1 - 0.9)(1 - 0.5)(1 - 0.25)(1 - 0.07)(1 - 0.01)(1 - \varepsilon)^3$$
$$\approx 0.9998$$

4.8. Solving the Radar Equation

The radar equation was developed in Chapter 1. It is given by

$$R = \left(\frac{P_t \tau f_r T_i G_t G_r \lambda^2 \sigma}{(4\pi)^3 k T_e F L (SNR)_o}\right)^{\frac{1}{4}} \tag{4.96}$$

where P_t is peak transmitted power, τ is pulse width, f_r is PRF, T_i is dwell interval, G_t is transmitting antenna gain, G_r is receiving antenna gain, λ is wavelength, σ is target cross section, k is Boltzman's constant, T_e is effective noise temperature, F is system noise figure, L is total system losses, and $(SNR)_o$ is the minimum SNR required for detection.

Assuming that the radar parameters such as power, antenna gain, wavelength, losses, bandwidth, effective temperature, and noise figure are known, the steps one should follow to solve for range are shown in Fig. 4.16. Note that both sides of the bottom half of Fig. 4.16 are identical. Nevertheless, we purposely show two paths so that a distinction between scintillating and non-fluctuating targets is made.

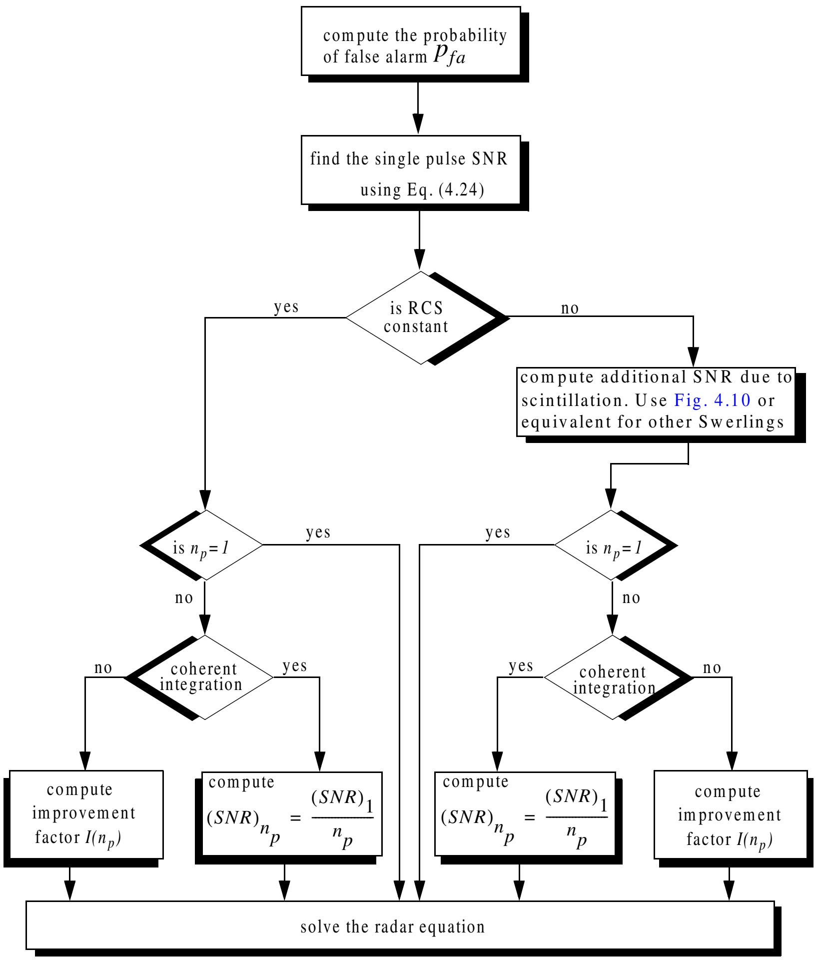


Figure 4.16. Solving the radar equation.

4.9. Constant False Alarm Rate (CFAR)

The detection threshold is computed so that the radar receiver maintains a constant pre-determined probability of false alarm. Eq. (4.19b) 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. (4.97):

$$V_T = \sqrt{2\psi^2 \ln\left(\frac{1}{P_{fa}}\right)} \tag{4.97}$$

If the noise power ψ^2 is assumed to be constant, then a fixed threshold can satisfy Eq. (4.97). 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].

4.9.1. Cell-Averaging CFAR (Single Pulse)

The CA-CFAR processor is shown in Fig. 4.17. 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 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

$$Y_1 \geq K_0 Z \tag{4.98}$$

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

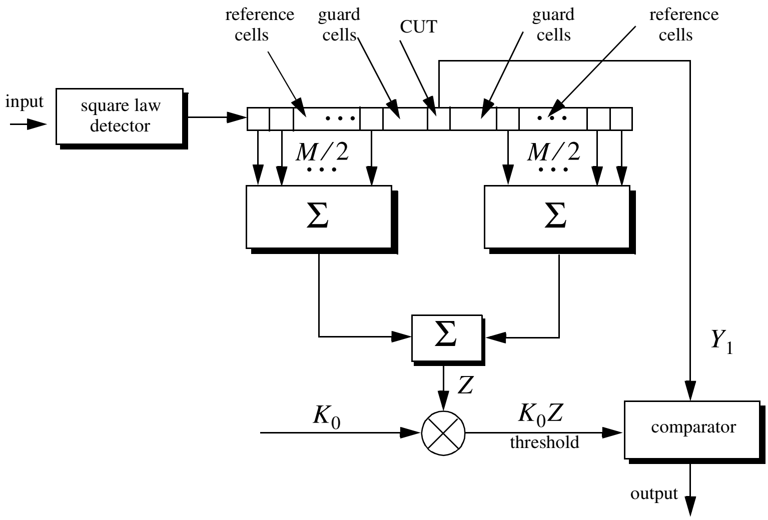


Figure 4.17. Conventional CA-CFAR.

Cell-averaging 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

$$f(z) = \frac{z^{(M/2)-1} e^{(-z/2\psi^2)}}{2^{M/2} \psi^M \Gamma(M/2)} \quad ; \; z > 0 \tag{4.99}$$

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

$$P_{fa}(V_T = y) = e^{-y/2\psi^2} \tag{4.100}$$

It follows that the unconditional probability of false alarm is

$$P_{fa} = \int_0^{\infty} P_{fa}(V_T = y) f(y) dy \tag{4.101}$$

where $f(y)$ is the *pdf* of the threshold, which except for the constant K_0 is the same as that defined in Eq. (4.99). Therefore,

$$f(y) = \frac{y^{M-1} e^{(-y/2K_0\psi^2)}}{(2K_0\psi^2)^M \Gamma(M)} \qquad ; \; y \geq 0 \tag{4.102}$$

Substituting Eqs. (4.102) and (4.100) into Eq. (4.101) yields

$$P_{fa} = \frac{1}{(1 + K_0)^M} \tag{4.103}$$

Observation of Eq. (4.103) shows that the probability of false alarm is now independent of the noise power, which is the objective of CFAR processing.

4.9.2. Cell-Averaging CFAR with Non-Coherent Integration

In practice, CFAR averaging is often implemented after non-coherent integration, as illustrated in Fig. 4.18. 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 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 who has a gamma *pdf* with $2Mn_p$ degrees of freedom.

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

$$P_{fa} = Prob\{Y_1/Z > K_1\} \tag{4.104}$$

Eq. (4.104) 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 value of the threshold. Therefore, let the conditional probability of false when $y = V_T$ be $P_{fa}(V_T = y)$. It follows that the unconditional false alarm probability is given by

$$P_{fa} = \int_0^\infty P_{fa}(V_T = y) f(y) dy \tag{4.105}$$

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

$$f(y) = \frac{(y/K_1)^{Mn_p - 1} e^{(-y/2K_0\psi^2)}}{(2\psi^2)^{Mn_p} K_1 \ \Gamma(Mn_p)} \qquad ; \; y \geq 0 \tag{4.106}$$

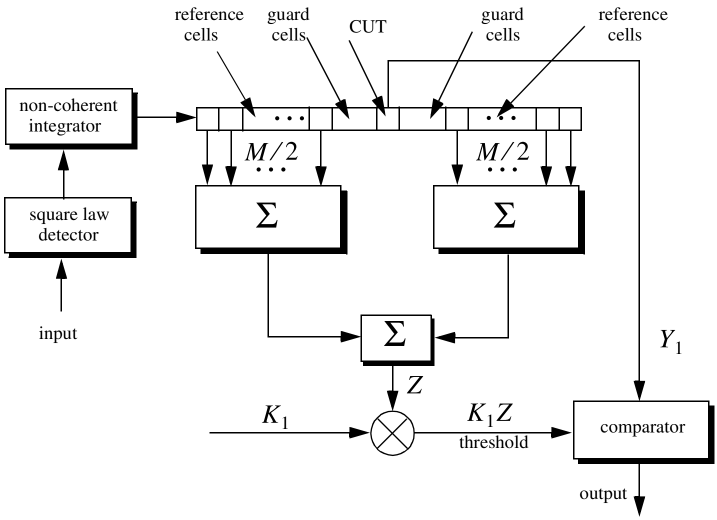


Figure 4.18. Conventional CA-CFAR with non-coherent integration.

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

$$P_{fa} = \frac{1}{(1+K_1)^{Mn_p}} \sum_{k=0}^{n_p-1} \frac{1}{k!} \frac{\Gamma(Mn_p+k)}{\Gamma(Mn_p)} \left(\frac{K_1}{1+K_1}\right)^k \tag{4.107}$$

which is identical to Eq. (4.103) when $K_1 = K_0$ and $n_p = 1$.

4.10. MATLAB Function and Program Listings

This section presents listings for all MATLAB programs/functions used in this chapter. The user is advised to rerun these programs with different input parameters. All functions have companion MATLAB *"filename_driver.m"* files that utilize MATLAB Graphical User Interface (GUI).

Listing 4.1. MATLAB Function "que_func.m"

```
function fofx = que_func(x)
% This function computes the value of the Q-function
% listed in Eq.(4.16). It uses the approximation in Eq.s (4.17) and (4.18)
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
```

Listing 4.2. MATLAB Function "marcumsq.m"

```
function PD = marcumsq (a,b)
% This function uses Parl's method to compute PD
max_test_value = 1000.; % increase to more than 1000 for better results
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 < max_test_value,
   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
```

Listing 4.3. MATLAB Program "prob_snr1.m"

```
% This program is used to produce Fig. 4.3
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 - dB')
ylabel ('Probability of detection')
grid
```

Listing 4.4. MATLAB Program "fig4_5.m"

```
% This program is used to produce Fig. 4.5
% It uses the function "improv_fac"
pfa1 = 1.0e-2;
pfa2 = 1.0e-6;
pfa3 = 1.0e-10;
pfa4 = 1.0e-13;
pd1 = .5;
pd2 = .8;
pd3 = .95;
pd4 = .999;
index = 0;
for np = 1:1:100
  index = index + 1;
  I1(index) = improv_fac (np, pfa1, pd1);
  I2(index) = improv_fac (np, pfa2, pd2);
  I3(index) = improv_fac (np, pfa3, pd3);
  I4(index) = improv_fac (np, pfa4, pd4);
end
np = 1:1:100;
semilogx (np, I1, 'k', np, I2, 'k--', np, I3, 'k-.', np, I4, 'k:')
set (gca,'xtick',[1 2 3 4 5 6 7 8  10 20 30  50 70 100]);
xlabel ('Number of pulses');
```

```
ylabel ('Improvement factor I - dB')
legend ('pd=.5, nfa=2','pd=.8, nfa=6','pd=.95, nfa=10','pd=.999, nfa=13');
```

Listing 4.5. MATLAB Function "improv_fac.m"

```
function impr_of_np = improv_fac (np, pfa, pd)
% This function computes the non-coherent integration improvement
% factor using the empirical formula defined in Eq. (4.49)
fact1 = 1.0 + log10( 1.0 / pfa) / 46.6;
fact2 = 6.79 * (1.0 + 0.253 * pd);
fact3 = 1.0 - 0.14 * log10(np) + 0.0183 * (log10(np)^2);
impr_of_np = fact1 * fact2 * fact3 * log10(np);
return
```

Listing 4.6. MATLAB Function "incomplete_gamma.m"

```
function [value] = incomplete_gamma ( vt, np)
% This function implements Eq. (4.63) to compute the Incomplete Gamma Function
format long
eps = 1.000000001;
% Test to see if np = 1
if (np == 1)
  value1 = vt * exp(-vt);
  value = 1.0 - exp(-vt);
  return
end
sumold = 1.0;
sumnew =1.0;
calc1 = 1.0;
calc2 = np;
xx = np * log(vt) - vt - factor(calc2);
temp1 = exp(xx);
temp2 = np / vt;
diff = .0;
ratio = 1000.0;
if (vt >= np)
  while (ratio >= eps)
    diff = diff + 1.0;
    calc1 = calc1 * (calc2 - diff) / vt ;
    sumnew = sumold + calc1;
    ratio = sumnew / sumold;
    sumold = sumnew;
  end
  value = 1.0 - temp1 * sumnew * temp2;
  return
else
  diff = 0.;
  sumold = 1.;
```

```
   ratio = 1000.;
   calc1 = 1.;
   while(ratio >= eps)
      diff = diff + 1.0;
      calc1 = calc1 * vt / (calc2 + diff);
      sumnew = sumold + calc1;
      ratio = sumnew / sumold;
      sumold = sumnew;
   end
   value = temp1 * sumnew;
end
```

Listing 4.7. MATLAB Function "threshold.m"

```
function [pfa, vt] = threshold (nfa, np)
% This function calculates the threshold value from nfa and np.
% The newton-Raphson recursive formula is used (Eq. (4.59)
% This function uses "incomplete_gamma.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 = 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);
   delta = abs(vt - vt0) * 10000.0;
   vt0 = vt;
end
```

Listing 4.8. MATLAB Function "pd_swerling5.m"

```
function pd = pd_swerling5 (input1, indicator, np, snrbar)
% This function is used to calculate the probability of
% 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 + 2.*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;
```

Listing 4.9. MATLAB Function "pd_swerling1.m"

```
function pd = pd_swerling1 (nfa, np, snrbar)
% This function is used to calculate the probability of
% 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 (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
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 = incomplete_gamma(vt,np-1);
   igf2 = incomplete_gamma(vt/temp,np-1);
   pd = 1.0 - igf1 + val1 * igf2 * exp(-vt/temp1);
```

Listing 4.10. MATLAB Function “pd_swerling2.m”

```
function pd = pd_swerling2 (nfa, np, snrbar)
% This function is used to calculate the probability of
% 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 (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
if (np <= 50)
   temp = vt / (1.0 + snrbar);
   pd = 1.0 - incomplete_gamma(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
```

Listing 4.11. MATLAB Function “pd_swerling3.m”

```
function pd = pd_swerling3 (nfa, np, snrbar)
% This function is used to calculate the probability of
% 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 (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
```

```
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
   temp4 = vt^(np-1.) * exp(-vt) / (temp1 * exp(factor(np-2.)));
   temp5 =  vt / (1.0 + 2.0 / (np *snrbar));
   pd = temp4 + 1.0 - incomplete_gamma(vt,np-1.) + ko * ...
      incomplete_gamma(temp5,np-1.);
end
```

Listing 4.12. MATLAB Function "pd_swerling4.m"

```
function pd = pd_swerling4 (nfa, np, snrbar)
% This function is used to calculate the probability of
% 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 (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
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
   snr = 1.0;
   gamma0 = incomplete_gamma(vt/beta,np);
   a1 = (vt / beta)^np / (exp(factor(np)) * exp(vt/beta));
   sum = gamma0;
   for i = 1:1:np
      temp1 = 1;
      if (i == 1)
         ai = a1;
      else
         ai = (vt / beta) * a1 / (np + i -1);
      end
      a1 = ai;
      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 / exp(factor(i));
      sum = sum + term;
   end
   pd = 1.0 - sum / beta^np;
end
pd = max(pd,0.);
```

Problems

4.1. In the case of noise alone, the quadrature components of a radar return are independent Gaussian random variables with zero mean and variance ψ^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 ψ that ensures a probability of false alarm $P_{fa} \leq 10^{-8}$.

4.2. (a) Derive Eq. (4.13); (b) derive Eq. (4.15).

4.3. A pulsed radar has the following specifications: time of false alarm $T_{fa} = 10\ minutes$, 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 non-coherent integration of 100 pulses, what is the SNR reduction so that P_D and P_{fa} remain unchanged?

4.4. 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 non-coherently? (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 $9Km$.

4.5. (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 $T_{fa} = 20\ minutes$, detection range $R = 12Km$, probability of detection $P_D = 0.5$ (three pulses). (b) If post detection integration is assumed, determine the SNR.

4.6. 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.

4.7. 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 (see Problem 4.6). (b) Write an expression for the *pdf* at the output of the envelope detector.

4.8. 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 $T_{fa} = 8\ minutes$, 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?

4.9. A certain radar utilizes 10 pulses for non-coherent 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.

4.10. 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 $P_{fa} = 10^{-6}$.

4.11. 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.

4.12. Repeat Example 4.3 with $P_D = 0.8$ and $P_{fa} = 10^{-5}$.

4.13. Derive Eq. (4.107).

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

4.15. 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.

4.16. Repeat Problem 4.15 for swerling IV type target.

4.17. 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.

4.18. Reproduce Fig. 4.10 for Swerling II, III, and IV type targets.

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