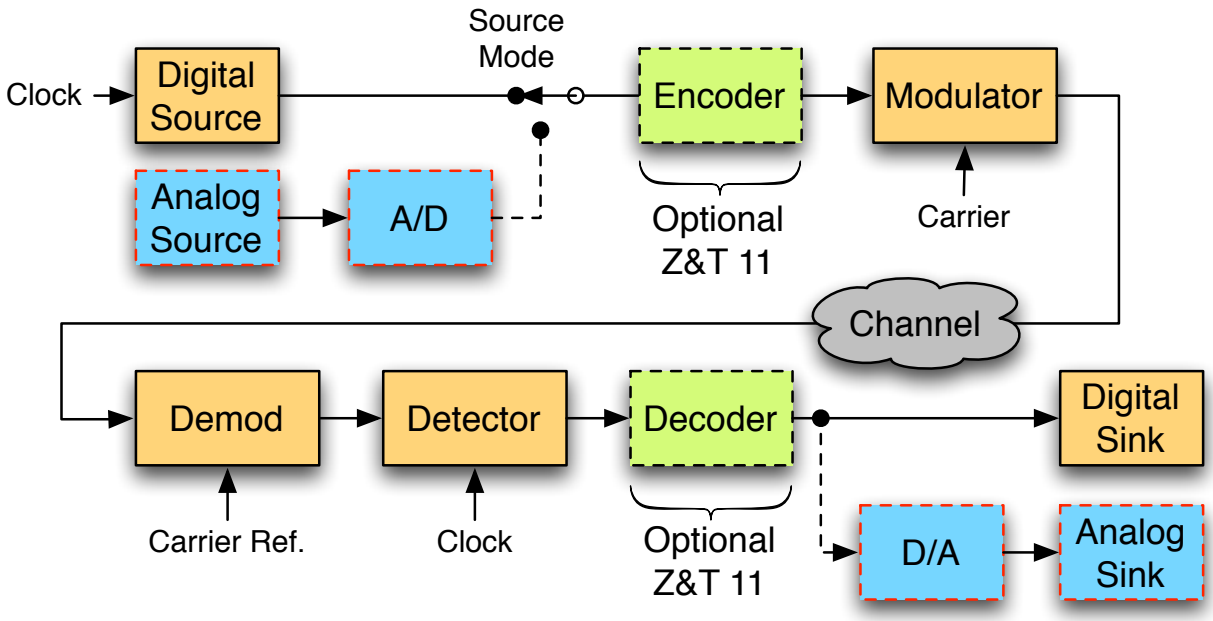
Chapter 5

Principles of Data Transmission in Noise (text chapter 9)

Contents

5.1 Introduction

In this chapter (Z&T Chapter 9) we formally jump into the business of digital communication systems. The focus of this chapter is the digital data transmission in a channel composed of additive WGN (AWGN), distortion caused by linear filtering, and carrier phase and symbol timing errors. Coding theory will not be developed until a later chapter.



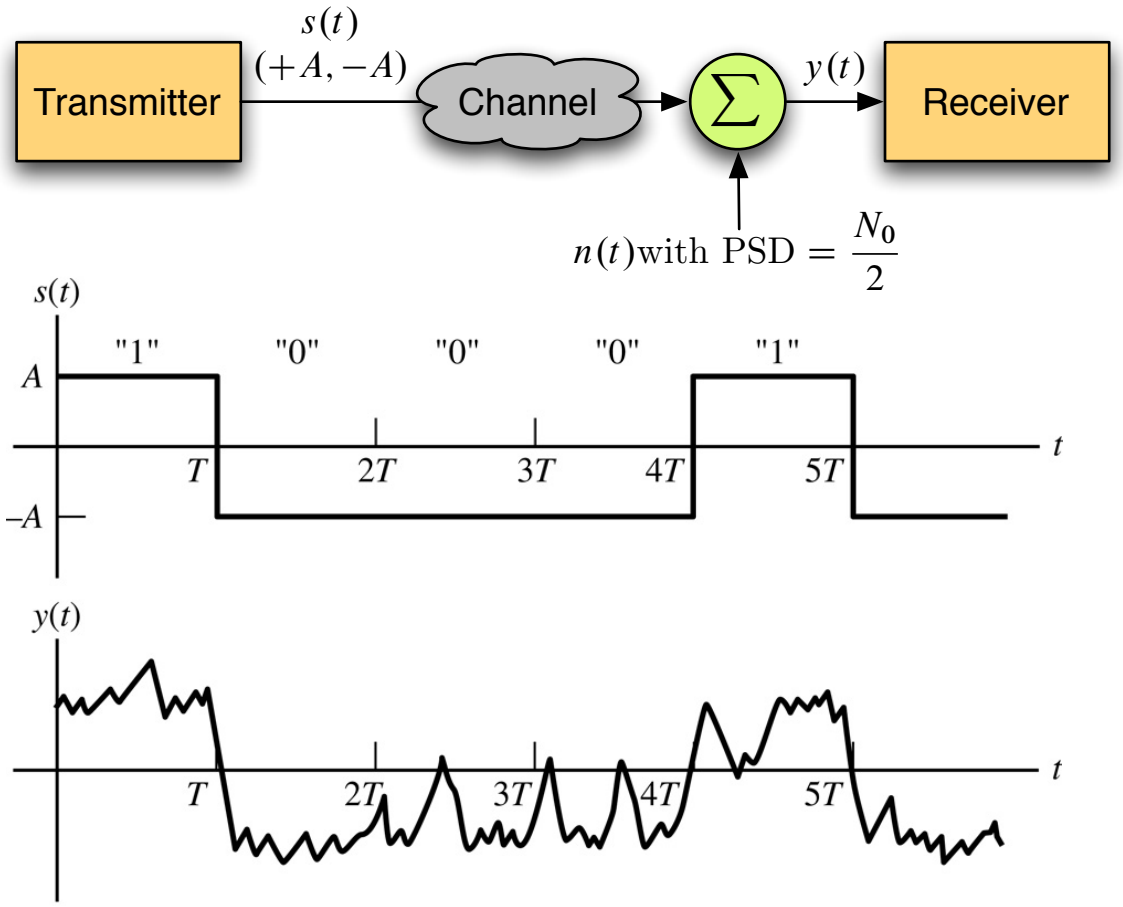
Block diagram of digital transmission system

- We begin with binary modulation schemes where the waveform signaling consists of one of two basic types
- Next we consider binary signaling with arbitrary signal shapes, this leads to the study of the *matched filter*

- Basic digital modulation schemes that follow from the matched filter are
  - *Amplitude-shift keying* (ASK)
  - *Phase-shift keying* (PSK)
  - *Frequency -shift keying* (FSK)
- A special case of PSK for example is the well known *quadriphase-shift keying* (QPSK)
- Modulation schemes not requiring a coherent reference are studied next
- We then move on to basic M-ary schemes, where $M > 2$ possible waveform types are sent over the channel
- Next pulse shaping for zero-ISI is considered
- Multipath interference, flat fading channels, and equalization are the final topics considered
- More digital modulation theory topics are taken up in the following chapter (Z&T Chapter 10)

5.2 Baseband Data Transmission in White Gaussian Noise

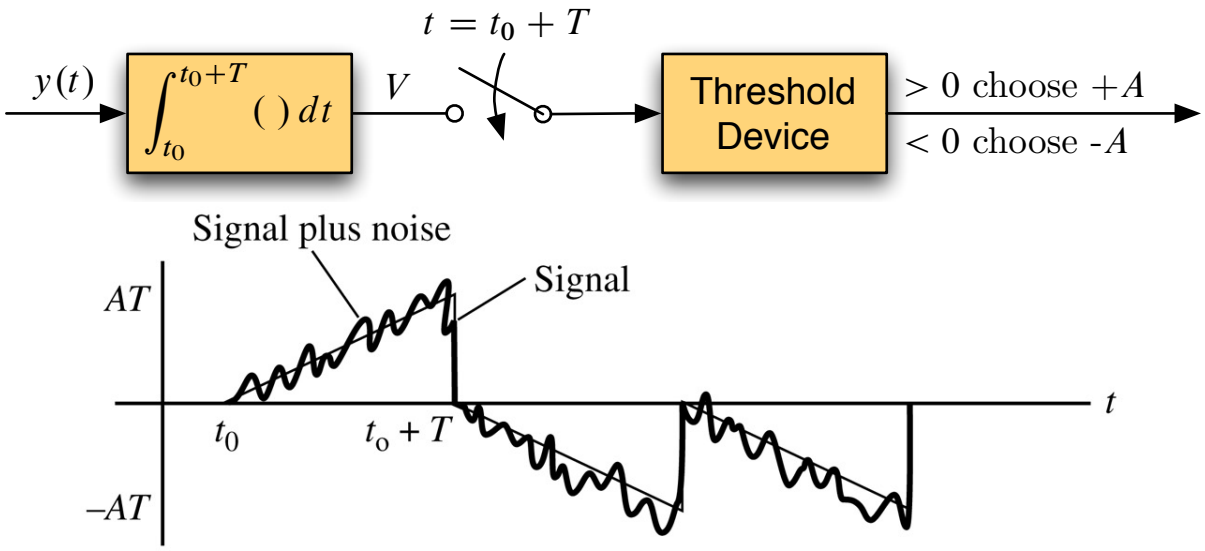
- As a starting point, in part motivated by the random binary pulse train example of Notes Chapter 4, we consider the following baseband transmission system:



Block diagram of a baseband binary transceiver

- We desire a receiver that can obtain the lowest bit error probability (BEP) for a given noise level
- We model the channel as additive WGN (AWGN)

- A receiver capable of the minimum BEP is the *integrate-and-dump* (I&D) receiver



Integrate and dump receiver

- To analyze the performance we start by writing the integrator output at the sampling instants as

$$V = \int_{t_0}^{t_0+T} \left[s(t) + n(t)\right] dt = \begin{cases} +AT + N, & +A \text{ sent} \\ -AT + N, & -A \text{ sent} \end{cases}$$

where $N = \int_{t_0}^{t_0+T} n(t)\,dt$ is the integrated noise voltage

- We know that N is a Gaussian random variable since integration is a linear operation, so we must find m_N and σ_N^2 to com-

plete the characterization

$$m_N = E[N] = E\left[\int_{t_0}^{t_0+T} n(t)\,dt\right] = \int_{t_0}^{t_0+T} E[n(t)]\,dt = 0$$

$$\sigma_N^2 = E\left[\left(\int_{t_0}^{t_0+T} n(t)\,dt\right)^2\right] - 0^2$$

$$= \int_{t_0}^{t_0+T}\int_{t_0}^{t_0+T} E[n(t)n(\sigma)]\,dt\,d\sigma$$

- Since $n(t)$ is white noise and WSS, we know that

$$E[n(t)n(\sigma)] = R_n(t-\sigma) = \frac{N_0}{2}\delta(t-\sigma)$$

- Returning to the calculation of σ_N^2, we have using the sifting property of the delta function

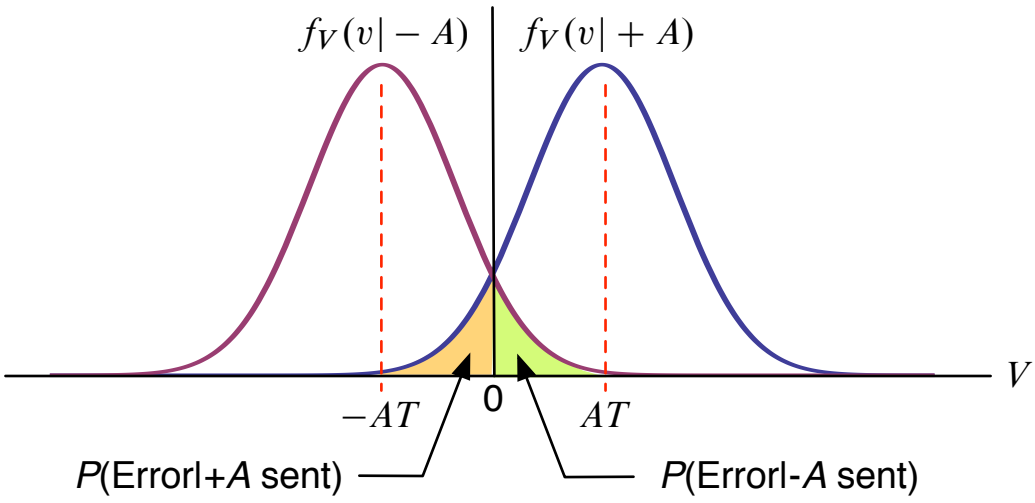
$$\sigma_N^2 = \frac{N_0}{2}\int_{t_0}^{t_0+T} 1\,d\sigma = \frac{1}{2}N_0T$$

- We see that the pdf on V is conditional Gaussian, i.e.,

$$f_{V|+A}(v|+A) = \frac{1}{\sqrt{\pi N_0 T}}e^{-(v-AT)^2/N_0T}$$

$$f_{V|-A}(v|-A) = \frac{1}{\sqrt{\pi N_0 T}}e^{-(v+AT)^2/N_0T}$$

- The threshold is set at $v_t = 0$, so the probability of error can now be calculated



Error probability analysis

$$
\begin{aligned}
P(\text{error}|A \text{ sent}) &= P(E|A) \\
&= \int_{-\infty}^{0} \frac{1}{\sqrt{\pi N_0 T}} e^{-\frac{(v-AT)^2}{N_0 T}}\, dv \\
P(\text{error}|\text{-}A \text{ sent}) &= P(E|-A) \\
&= \int_{0}^{\infty} \frac{1}{\sqrt{\pi N_0 T}} e^{-\frac{(v+AT)^2}{N_0 T}}\, dv
\end{aligned}
$$

- In the first line we make the substitution $u = (v-AT)/\sqrt{N_0 T/2}$ and in the second line $u = (v+AT)/\sqrt{N_0 T/2}$
- Making the substitutions and recalling the definition of the Gaussian $Q(\,)$ function we can write

$$
P(E|A) = \int_{-\infty}^{\sqrt{2A^2T/N_0}} \frac{e^{-u^2/2}}{\sqrt{2\pi}}\, du = Q\left(\sqrt{\frac{2A^2T}{N_0}}\right)
$$

$$
P(E|-A) = \int_{\sqrt{2A^2T/N_0}}^{\infty} \frac{e^{-u^2/2}}{\sqrt{2\pi}}\, du = Q\left(\sqrt{\frac{2A^2T}{N_0}}\right)
$$

- The average bit error probability (BEP) is

$$P_E = P(E|+A)P(+A) + P(E|-A)P(-A)$$

- Since $P(+A) + P(-A) = 1$, it follows that

$$P_E = Q\left(\sqrt{\frac{2A^2T}{N_0}}\right)$$

- There are two important physical interpretations of the $Q(\,)$ function argument:

  1. Note that under the present signaling scheme, the energy per bit is

$$E_b = \int_{t_0}^{t_0+T} A^2\,dt = A^2T,$$

  so we can form the ratio

$$z = \frac{A^2T}{N_0} = \frac{E_b}{N_0} = \frac{\text{energy per bit}}{\text{noise power spectral density}}$$

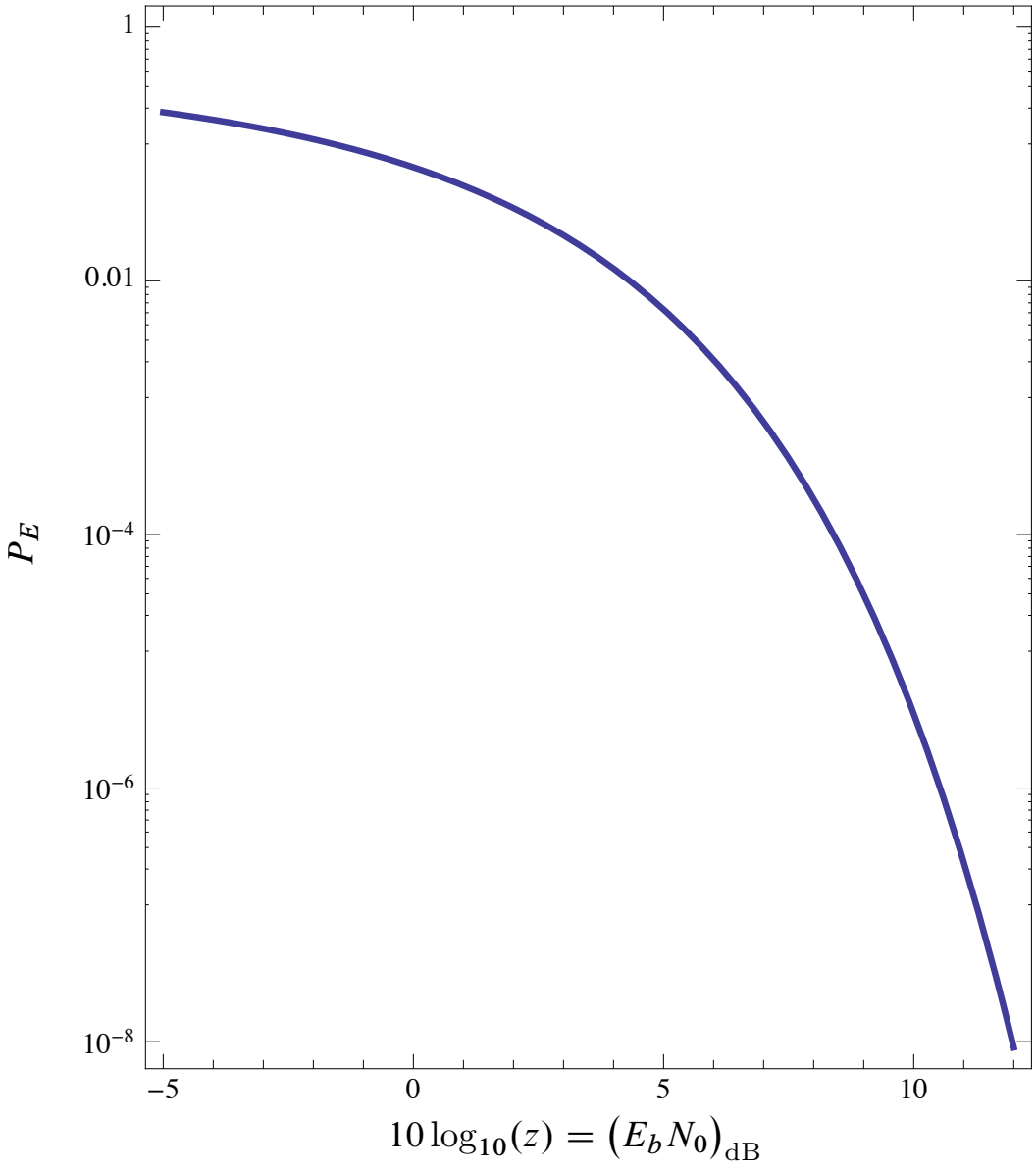
  2. We may also view z as the ratio of signal power to noise power, i.e.,

$$z = \frac{A^2}{N_0(1/T)} = \frac{A^2}{N_0B_p} = \frac{A^2}{N_0R_b},\ R_b = \text{bit rate}$$

  where $B_p = 1/T$ is a rough measure of the signal low-pass bandwidth or *bit-rate bandwidth* (recall the PSD of the random binary pulse train is $AT\,\text{sinc}^2(fT)$

- Note that z is referred to as the SNR or E_b/N_0 for this system

- Typically P_E is plotted as $\log_{10} P_E$ versus E_b/N_0 in dB



BEP plot for baseband signaling

Example 5.1: **Find** P_E

- Given $N_0 = 10^{-10}$ W/Hz, $A = 50$mv, and the bit rate is $R_b = 1/T = 5$ Mbps, find P_E

- We need to form the ratio $z = E_b/N_0$ with the units properly resolved

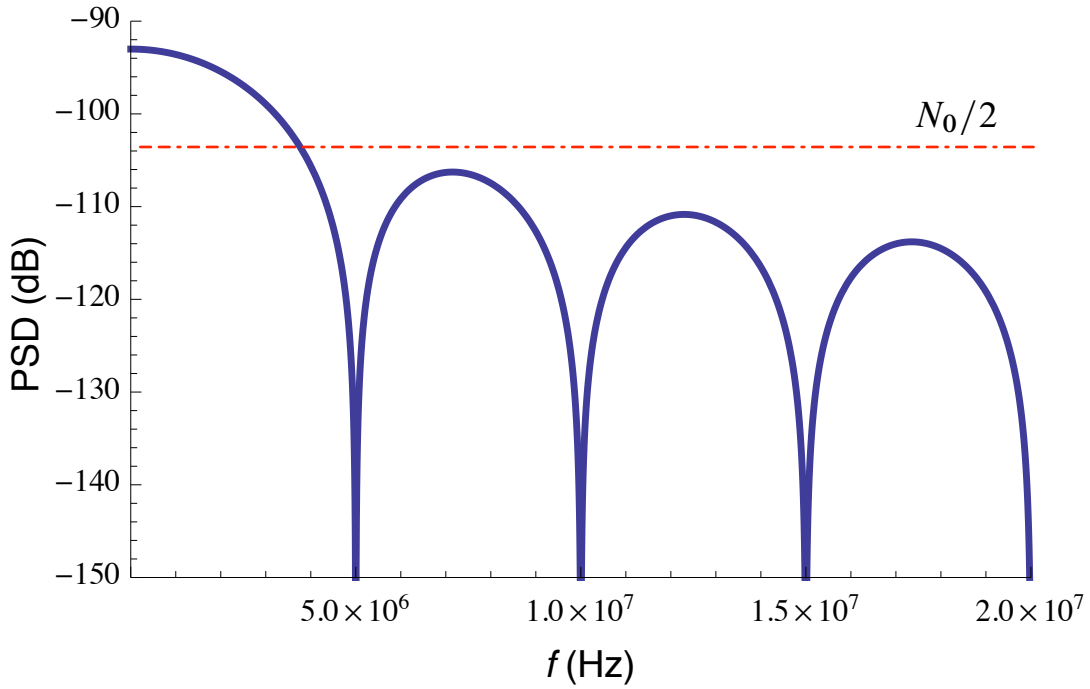
$$z = \frac{E_b}{N_0} = \frac{A^2 T}{N_0} = \frac{(0.05)^2 \cdot 1/(5 \times 10^6)}{10^{-10}} = 5.0$$

thus

$$P_E = Q\left(\sqrt{2 \cdot 5}\right) = \frac{1}{2}\text{erfc}\left(\sqrt{\frac{2 \cdot 5}{2}}\right) = 7.827 \times 10^{-4}$$

- Sketch the signal power spectrum

$$A^2\text{sinc}^2(fT) = \frac{A^2}{R_b}\text{sinc}^2(f/R_b)$$



5 Mbps signal spectrum

Example 5.2: Find E_b/N_0 in dB

- Find $(E_b/N_0)_{\text{dB}}$ to achieve $P_E = 10^{-5}$ and 10^{-6}
- Both Mathematica and MATLAB have inverse erfc functions, `InverseErfc[ ]` and `erfcinv( )` respectively
- Given the P_E expression in terms of erfc( ) we can write

$$\frac{E_b}{N_0} = \left(\text{erfcinv}(2P_E)\right)^2$$

or in dB

$$\left(\frac{E_b}{N_0}\right)_{\text{dB}} = 20\log_{10}\left[\text{erfcinv}(2P_E)\right]$$

- The specific answers are

$$\left(\frac{E_b}{N_0}\right)_{\text{dB, }10^{-5}} = 9.59 \text{ dB}$$

$$\left(\frac{E_b}{N_0}\right)_{\text{dB, }10^{-6}} = 10.53 \text{ dB}$$

5.3 Binary Data Transmission with Arbitrary Signal Shapes

- We now expand the analysis to include arbitrary signal shapes of finite energy on Ts, and establish an optimal receiver structure from a BEP standpoint

- This framework will be useful for the analysis of a variety of basic binary modulation schemes
- Let $s_1(t)$ be the signal/pulse sent to represent a logical '1' and $s_2(t)$ be the signal/pulse sent to represent a logical '0'
- The energies of these signals are denoted

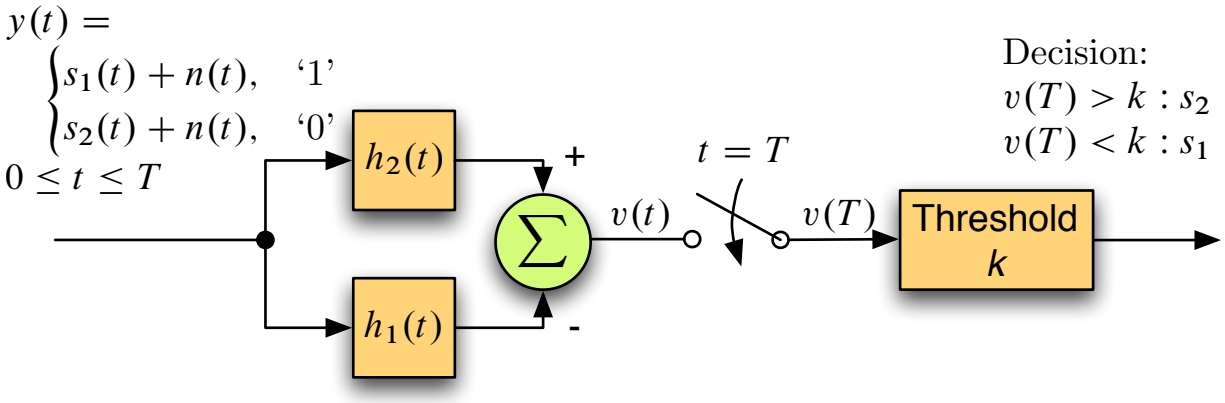
$$E_1 \triangleq \int_{t_0}^{t_0+T} s_1^2(t)\,dt \quad \text{and} \quad E_2 \triangleq \int_{t_0}^{t_0+T} s_2^2(t)\,dt$$

- Note for binary ASK, PSK , and FSK we can define $s_1(t)$ and $s_2(t)$ as follows:

Case	$s_1(t)$	$s_2(t)$	**Notes**
ASK	0	$A\cos(\omega_c t)$	high peak to avg.
PSK	$A\sin(\omega_c t + \beta)$	$A\sin(\omega_c t - \beta)$	$\beta = \cos^{-1} m$, typ $\frac{\pi}{2}$
FSK	$A\cos(\omega_c t)$	$A\cos[(\omega_c + \Delta\omega)t]$	BW set by $\Delta\omega$

5.3.1 Receiver Structure and Error Probability

- What should the receiver look like?
- With AWGN present we expect that a linear filter followed by a decision device should work
- Two filters may make more sense initially, one for each signal type, a difference, and then a comparison
  - Signal space methods of Z&T Chapter 10 or Rice Chapter 5 would make this more structured



Candidate receiver for binary signals in AWGN

- The two filters in the above figure is equivalent to $h(t) = h_2(t) - h_1(t)$, so in either case we have a zero mean Gaussian RP having variance

$$\sigma_0^2 = \int_{-\infty}^{\infty} \frac{N_0}{2} |H(f)|^2 \, df$$

present at the receiver decision point

- The sample voltage $V \triangleq v(T)$ is of the form

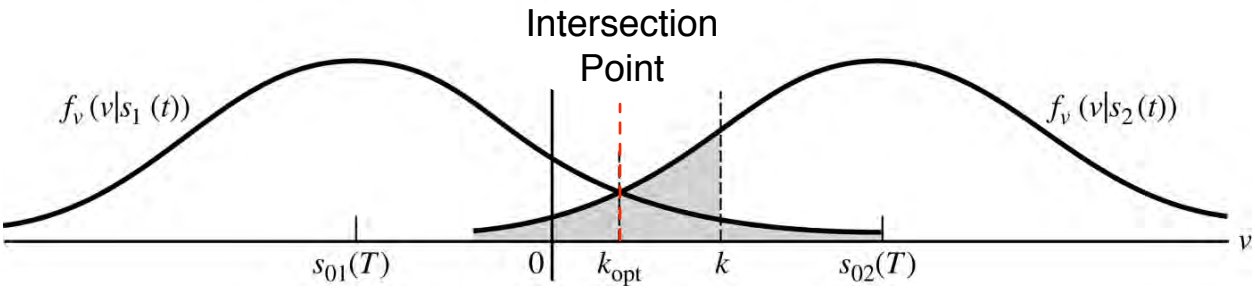
$$V = \begin{cases} s_{01}(T) + N, & \text{'1' transmitted} \\ s_{02}(T) + N, & \text{'0' transmitted} \end{cases}$$

where N is zero mean Gaussian making V a conditional Gaussian RV

- We thus have the conditional error probabilities

$$P(E|s_1(t)) = \int_k^{\infty} \frac{e^{-[v - s_{01}(T)]^2/2\sigma_0^2}}{\sqrt{2\pi\sigma_0^2}} \, dv$$

$$P(E|s_2(t)) = \int_{-\infty}^{k} \frac{e^{-[v - s_{02}(T)]^2/2\sigma_0^2}}{\sqrt{2\pi\sigma_0^2}} \, dv$$



Decision point conditional pdf's

- The average BEP is again

$$\begin{aligned} P_E &= P[E|s_1(t)]P[s_1(t) \text{ sent}] + P[E|s_2(t)]P[s_2(t) \text{ sent}] \\ &\overset{\text{eq}}{=} \frac{1}{2}P[E|s_1(t)] + \frac{1}{2}P[E|s_2(t)] \end{aligned}$$

- Because the conditional pdf's have equal variance, the optimal threshold k_{opt} can be found at the intersection point for the case of equally likely '1's and '0's

$$k_{\text{opt}} = \frac{1}{2}\left[s_{01}(T) + s_{02}(T)\right]$$

- Under this assumption it then follows that P_E reduces to

$$P_E = Q\left(\frac{s_{02}(T) - s_{01}(T)}{2\sigma_0}\right)$$

- We see that as the *distance* (signal space concepts seen later) between the signals increases P_E decreases

- We still do not know how to choose $h(t)$ to minimize P_E

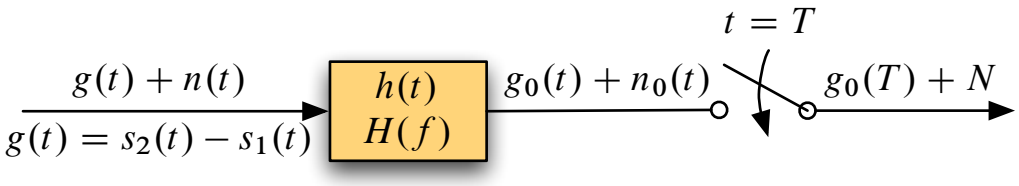
5.3.2 The Matched Filter

- The *matched filter* concept is fundamental to this chapter
- We will see that when we choose $h(t)$ to maximize the SNR at the sampling instant, we will achieve the minimum P_E
- To begin the analysis we let

$$\zeta = \frac{s_{02}(T) - s_{01}(T)}{\sigma_0}$$

  knowing that the Q-function is monotone decreasing

- Furthermore let $g(t) = s_2(t) - s_1(t)$, since the filtered version of this signal normalized by the filtered noise is what we are trying to maximize



Model for finding the optimum $h(t) \leftrightarrow H(f)$

- We now maximize ζ^2 where $\zeta = g_0(T)/\sigma_0$, through choice of $H(f)$
- We will work with frequency domain expressions making use of Parseval's theorem and the *Schwarz's inequality*

$$\zeta^2 = \left. \frac{g_0^2(t)}{E\left[n_0^2(t)\right]} \right|_{t=T}$$

- The denominator can be written as

$$E\left[n_0^2(t)\right] = \frac{N_0}{2}\int_{-\infty}^{\infty} |H(f)|^2\, df$$

- The numerator can be written as

$$g_0(T) = \mathcal{F}^{-1}\left\{G(f)H(f)\right\}\Big|_{t=T} = \int_{-\infty}^{\infty} H(f)G(f)e^{j2\pi fT}\, df$$

so

$$g_0^2(T) = \left|\int_{-\infty}^{\infty} H(f)G(f)e^{j2\pi fT}\, df\right|^2$$

- Now we can write

$$\zeta^2 = \frac{\left|\int_{-\infty}^{\infty} H(f)G(f)e^{j2\pi fT}\, df\right|^2}{\frac{N_0}{2}\int_{-\infty}^{\infty}|H(f)|^2\, df}$$

- The Schwarz inequality states that for vectors **A** and **B**

$$|\mathbf{A}\cdot\mathbf{B}| = |AB\cos\theta| \le |\mathbf{A}||\mathbf{B}|$$

- Here we are dealing with generalized vector spaces with the inner product being defined as

$$X(f)\cdot Y(f) = \langle X(f), Y(f)\rangle = \int_{-\infty}^{\infty} X(f)Y^*(f)\, df$$

- Here Schwarz's inequality is of the form

$$\left|\int_{-\infty}^{\infty} X(f)Y^*(f)\, df\right|^2 \le \int_{-\infty}^{\infty}|X(f)|^2\, df\cdot\int_{-\infty}^{\infty}|Y(f)|^2\, df$$

with the equality holding if and only if $X(f) = mY(f)$ (m a constant)

- We make the substitutions $X(f) = H(f)$ and $Y^*(f) = G(f)e^{j2\pi fT}$

$$\zeta^2 = \frac{2}{N_0}\frac{\left|\int_{-\infty}^{\infty} H(f)G(f)e^{j2\pi fT}\,df\right|^2}{\int_{-\infty}^{\infty}|H(f)|^2\,df} \leq \frac{2}{N_0}\frac{\int_{-\infty}^{\infty}|H(f)|^2\,df\int_{-\infty}^{\infty}|G(f)|^2\,df}{\int_{-\infty}^{\infty}|H(f)|^2\,df}$$

- Finally,

$$\zeta^2_{\max} = \frac{2}{N_0}\int_{-\infty}^{\infty}|G(f)|^2\,df$$

  if and only if

$$H(f) = m'G^*(f)e^{-j2\pi fT}$$

  - Since m' is arbitrary, and both signal and noise are equally amplified by m', we set $m' = 1$

- The impulse response for the matched filter is

$$h_0(t) = \mathcal{F}^{-1}\{H_0(f)\} = \int_{-\infty}^{\infty} G^*(f)e^{-j2\pi fT}\,e^{j2\pi ft}\,df = \int_{-\infty}^{\infty} G(f)e^{-j2\pi f(T-t)}\,df$$

  or

$$h_0(t) = g(T-t) = \underbrace{s_2(T-t)}_{h_2(t)} - \underbrace{s_1(T-t)}_{h_1(t)},$$

  which is just the time reverse of the pulse shapes delayed by T to account for causality (note we may need to make the delay larger to insure realizability)

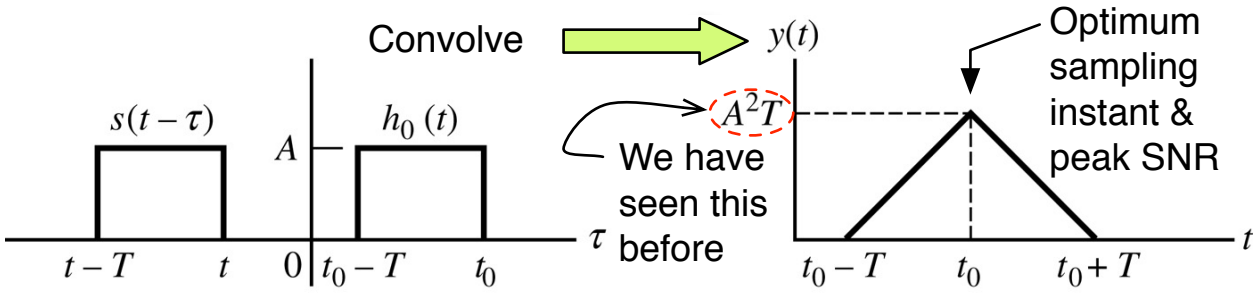
Example 5.3: **REC Pulse Shape**

- Consider the REC pulse shape used in the baseband transmission in noise scheme

$$s(t) = \Pi\left(\frac{t - T/2}{T}\right) = \begin{cases} A, & 0 \le t \le T \\ 0, & \text{otherwise} \end{cases}$$

- The matched filter is

$$h_0(t) = s(t_0 - t) = \begin{cases} A, & t_0 - T \le t \le t_0 \\ 0, & \text{otherwise} \end{cases}$$



The matched filter with an input pulse passed to the output

- The output is

$$y(t) = h_0(t) * s(t) = \int_{-\infty}^{\infty} h_0(\tau) s(t - \tau)\, d\tau$$

- Here it turns out that we want $t_0 = T$

5.3.3 Error Probability for the Matched-Filter Receiver

- The BEP for the matched filter receiver follows immediately, since

$$P_E = Q\left(\frac{s_{02}(T) - s_{01}(T)}{2\sigma_0}\right) = Q\left(\frac{\zeta_{\max}}{2}\right)$$

- We can put the above into a more user friendly form using Parseval's theorem

$$\begin{aligned}\zeta_{\max}^2 &= \frac{2}{N_0}\int_{-\infty}^{\infty} |G(f)|^2\, df = \frac{2}{N_0}\int_{-\infty}^{\infty} |S_2(f) - S_1(f)|^2\, df \\ &= \frac{2}{N_0}\int_{-\infty}^{\infty} \left[s_2(t) - s_1(t)\right]^2 dt \\ &= \frac{2}{N_0}\left[\int_{-\infty}^{\infty} s_2^2(t)\, dt + \int_{-\infty}^{\infty} s_1^2(t)\, dt \right. \\ &\qquad \left. -2\int_{-\infty}^{\infty} s_1(t)s_2(t)\, dt\right]\end{aligned}$$

- The last term in the last line can be viewed as the correlation between the two signals $s_1(t)$ and $s_2(t)$, so we can define the signal correlation coefficient, $\rho_{12} \in [-1, 1]$, as

$$\rho_{12} \triangleq = \frac{1}{\sqrt{E_1 E_2}}\int_{-\infty}^{\infty} s_1(t)s_2(t)\, dt$$

- Finally we can write

$$\zeta_{\max}^2 = \frac{2}{N_0}\left(E_1 + E_2 - 2\sqrt{E_1 E_2}\rho_{12}\right)$$

and inserting into the $Q(\,)$ function

$$P_E = Q\left(\sqrt{\frac{E_1 + E_2 - 2\sqrt{E_1 E_2}\rho_{12}}{2N_0}}\right)$$
$$= Q\left(\sqrt{\frac{E}{N_0}\left(1 - \frac{\sqrt{E_1 E_2}}{E}\rho_{12}\right)}\right)$$

where

$$E = \frac{E_1 + E_2}{2} = \text{Averaged received energy}$$

- **Note**: It is advantageous to chose/make $\rho_{12} = -1$ to obtain the lowest P_E, i.e., maximize the Q-function argument
- Another way of writing P_E is

$$P_E = Q\left(\sqrt{z(1 - R_{12})}\right)$$

where

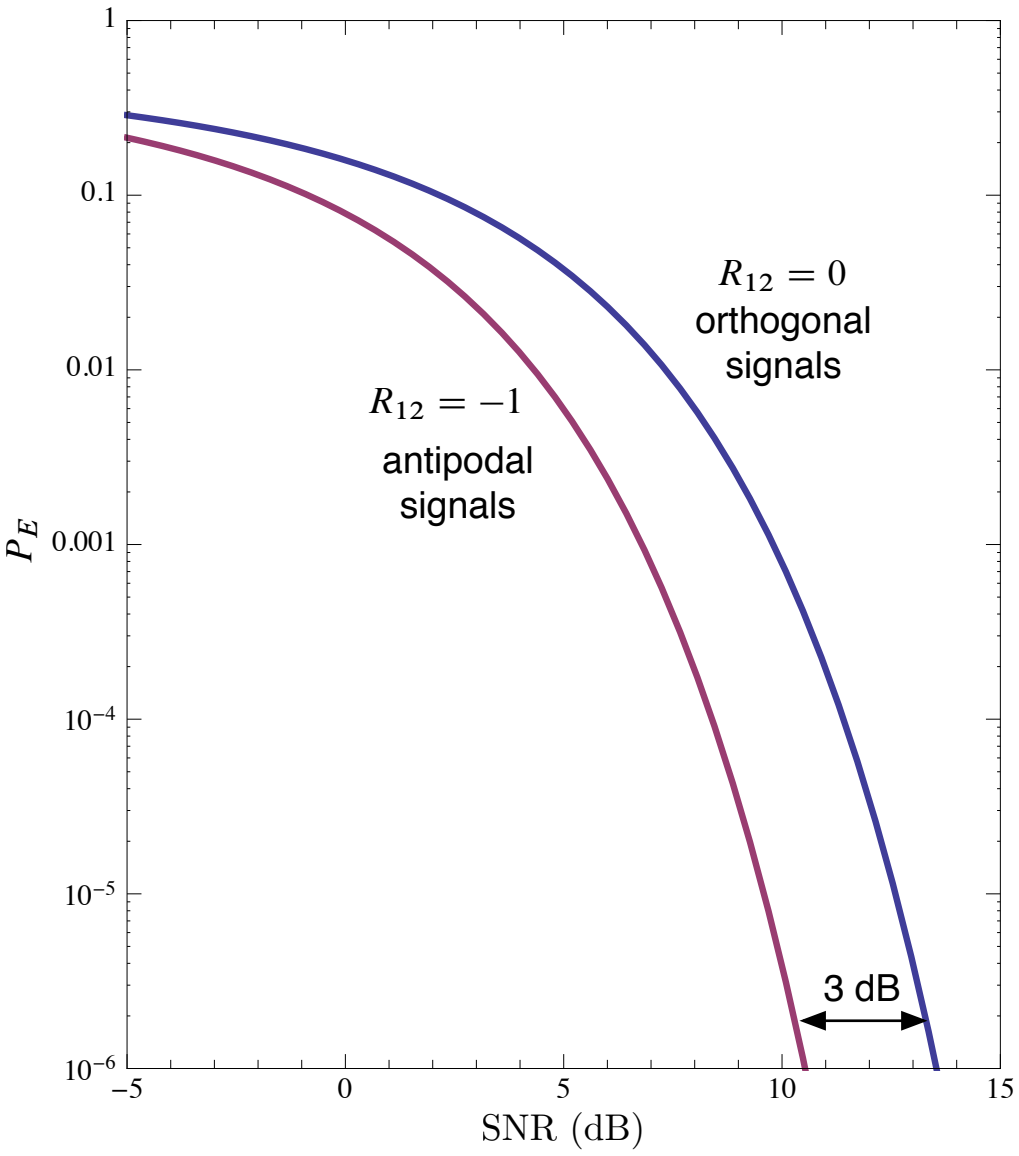
$$R_{12} = \frac{2\sqrt{E_1 E_2}}{E_1 + E_2}\rho_{12} = \frac{\sqrt{E_1 E_2}}{E}\rho_{12}$$

- For the case where $E_1 = E_2$ and $R_{12} = -1$ we have

$$P_E = Q\left(\sqrt{2z}\right)$$

which is identical to the baseband signaling case first considered (recall the integrate and dump receiver)

- Furthermore we can interpret $R_{12} = -1$ as *antipodal* signals and $R_{12} = 0$ as orthogonal signals; which gives better performance?



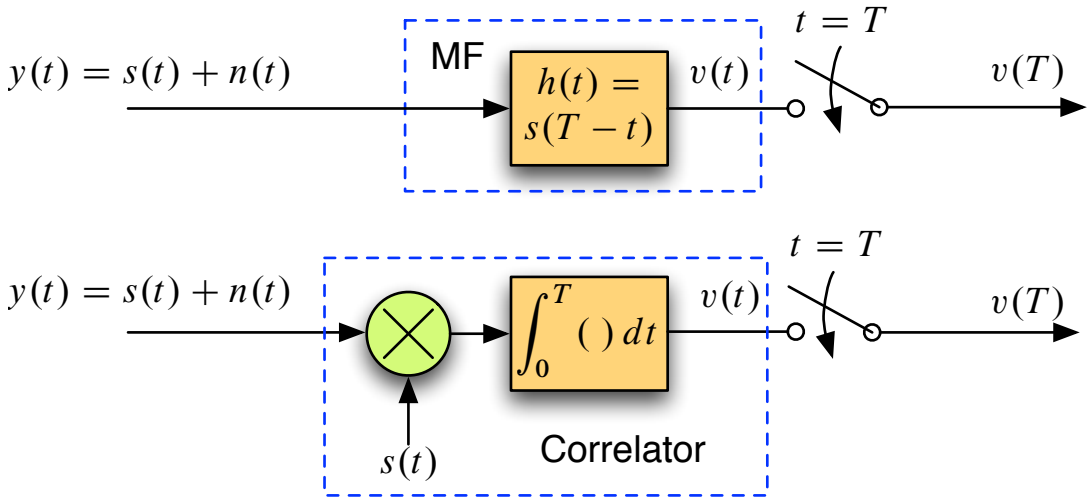
P_E comparison for $R_{12} = 0$ and 1

5.3.4 Correlator Implementation of the MF Receiver

- For the baseband binary signal with REC pulses, we have seen that both the I&D and matched filter receivers give the same performance
- The matched filter receiver works for arbitrary pulse shapes,

but is it the only receiver structure for the general case? No!

- A signal *correlator* also works:



Correlator versus matched filter

- To see that the two implementations are equivalent first consider the matched filter output

$$v_{\text{MF}}(t) = h(t) * y(t) = \int_0^T s(T-\tau)y(t-\tau)\,d\tau,$$

sampled at $t = T$ is

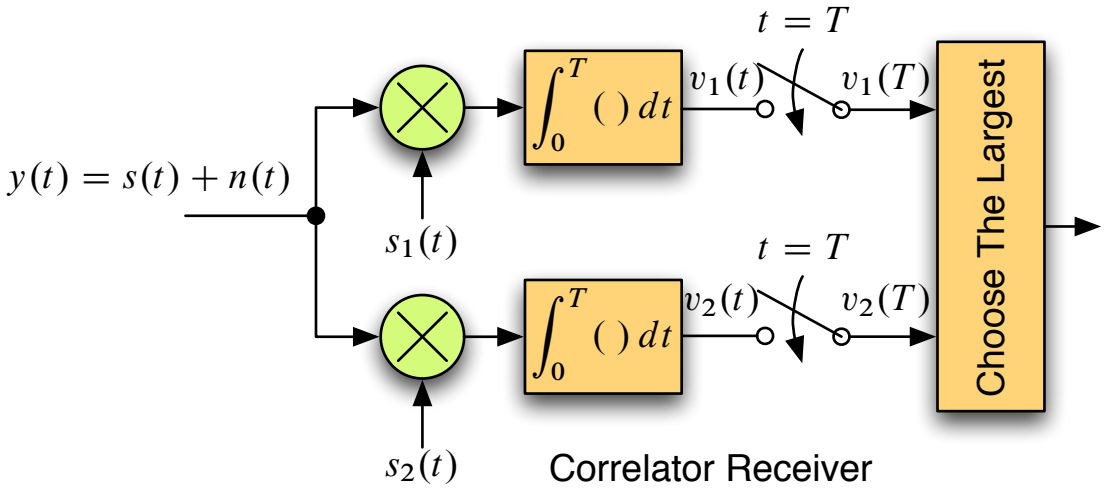
$$v_{\text{MF}}(T) = \int_0^T s(T-\tau)y(T-\tau)\,d\tau = \int_0^T s(\alpha)y(\alpha)\,d\alpha$$

- For the correlator we have

$$v_{\text{corr}}(T) = \int_0^T s(t)y(t)\,dt = v_{\text{MF}}(T)$$

- The I&D receiver is actually a correlation receiver for the special case of $s(t) = 1$ for $t \in [0, T]$

- For binary signaling using $s_1(t)$ and $s_2(t)$ the correlator receiver is of the form



Correlator receiver for arbitrary signals

5.3.5 Optimum Threshold

- We saw earlier that the optimum threshold for equally likely signals was given in terms of $[s_{01}(T) + s_{02}(T)]/2$
- We can write this result in terms of the signal energies using the general matched filter design expression, i.e.,

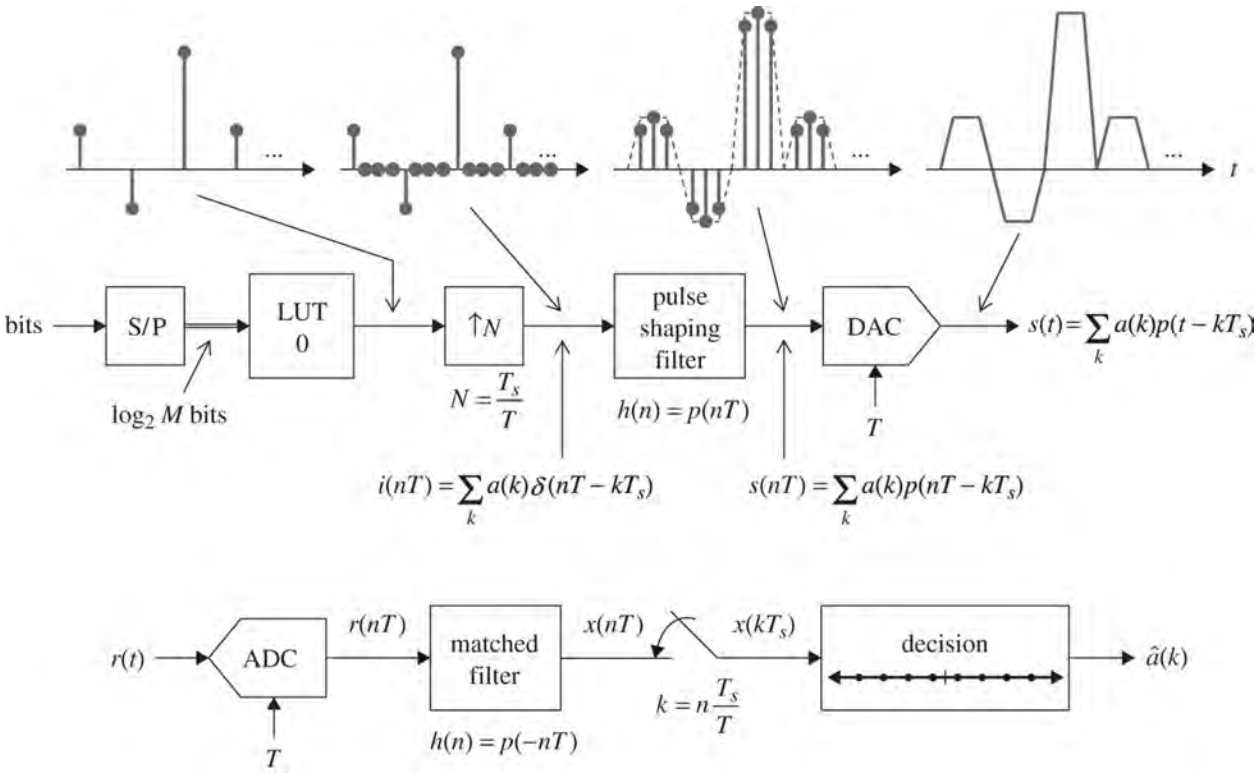
$$\begin{aligned} s_{01}(T) &= \int_{-\infty}^{\infty} h(\tau) s_1(T-\tau)\, d\tau \\ &= \int_{-\infty}^{\infty} \big[s_2(T-\tau) - s_1(T-\tau)\big] s_1(T-\tau)\, d\tau \\ &= \sqrt{E_1 E_2}\rho_{12} - E_1 \\ s_{02}(T) &= E_2 - \sqrt{E_1 E_2}\rho_{12} \end{aligned}$$

- Thus independent of the actual pulse shape

$$k_{\text{opt}} = \frac{1}{2}(E_2 - E_1)$$

Example 5.4: **Baseband Modulation Using DSP [Rice]**

- Baseband modulation for either Python/Julia/MATLAB simulation or true real-time DSP implementation consider the following:



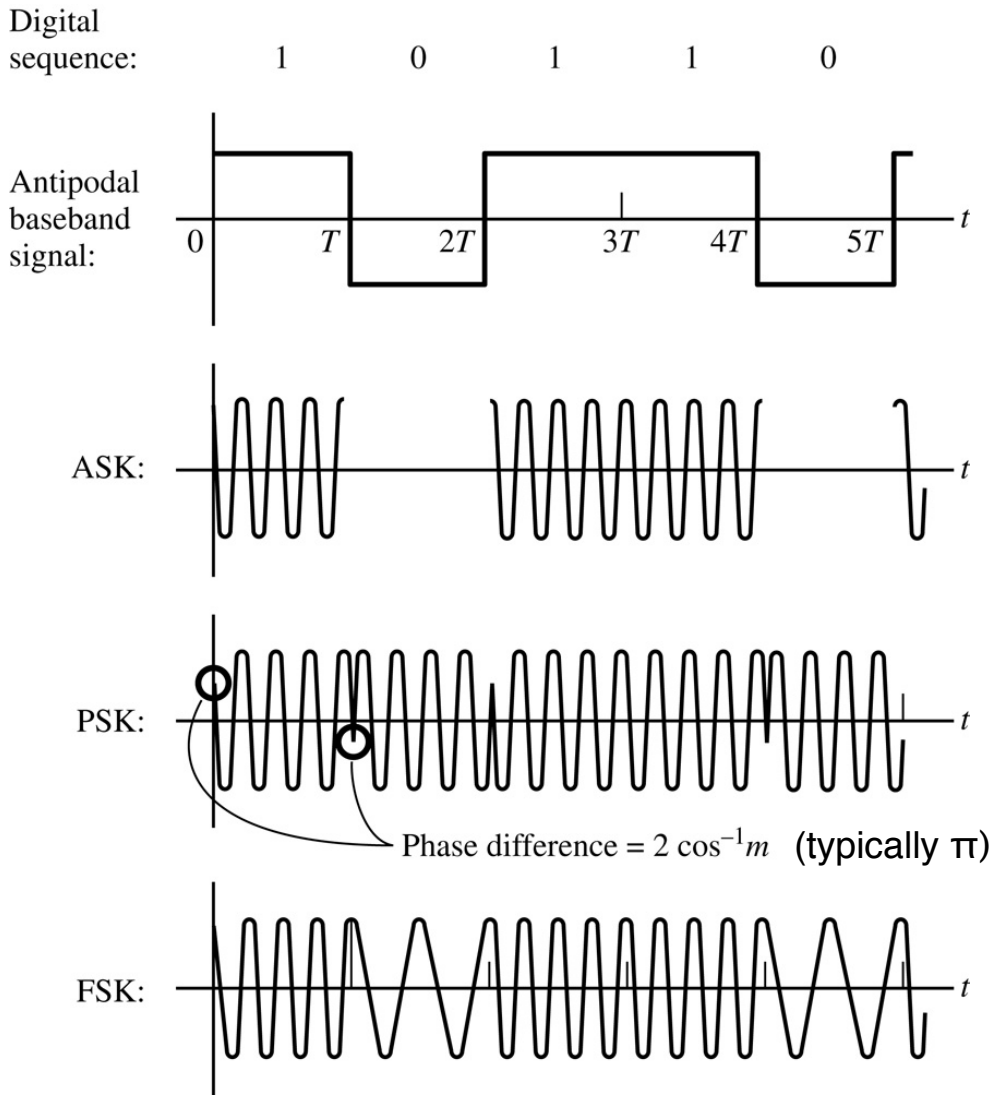
Baseband modulation using DSP

- This is a generalization of the random pulse train MATLAB function `PT_mod()` function described in notes Chapter 4, Example 4.10

- In the context of this Chapter the pulse shaping filter is rectangular, as that is the NRZ waveform we have been dealing with thus far

5.3.6 Error Probabilities for Coherent Binary Signaling

- Using the matched filter results just developed, we now study the BEP of three coherent binary signaling schemes



Binary ASK, PSK, and FSK

- Some receiver imperfections will also be considered

Amplitude-Shift Keying (ASK)

- With this scheme an actual signal, $\cos(\omega_c t)$, is transmitted only when a '1' or '0' is being sent, not both however
- For that reason binary ASK is sometimes called *on-off keying*
- In a correlator-based receiver the correlation signal matches the transmitted signal, so the correlation is either

$$\int_0^T [A\cos(\omega_c t) + n(t)]A\cos(\omega_c t)\,dt$$

$$\text{or} \int_0^T [0 + n(t)]A\cos(\omega_c t)\,dt$$

- The signal energy either $E_1 = A^2T/2$ or 0
- The correlation is

$$R_{12} = \rho_{12} = 0$$

  and the threshold is at $(A^2T/2)/2 = A^2T/4$
- The average P_E is given by

$$P_{E,\text{ASK}} = Q\left(\sqrt{z}\right)$$

- Performance is 3dB worse than the binary antipodal case considered earlier
- This signal also requires a peak power which is twice the average power

Phase-Shift Keying (PSK)

- For binary PSK the transmitted signal is of the form

$$s_k(t) = A \sin\left[\omega_c t - (-1)^k \cos^{-1} m\right],\ 0 \le t \le T,\ k = 1, 2$$

  where as noted earlier $\cos^{-1} m$ is the modulation index, typically given by $\pi/2$, so the carrier phase switches between 0 and π, and we assume for simplicity that $\omega_c = 2\pi n/T$ (e.g., double frequency terms in the correlator will integrate to zero)

- To analyze the receiver for the general case the modulation index, we start by expanding the PSK waveform into a sum carrier and modulation terms

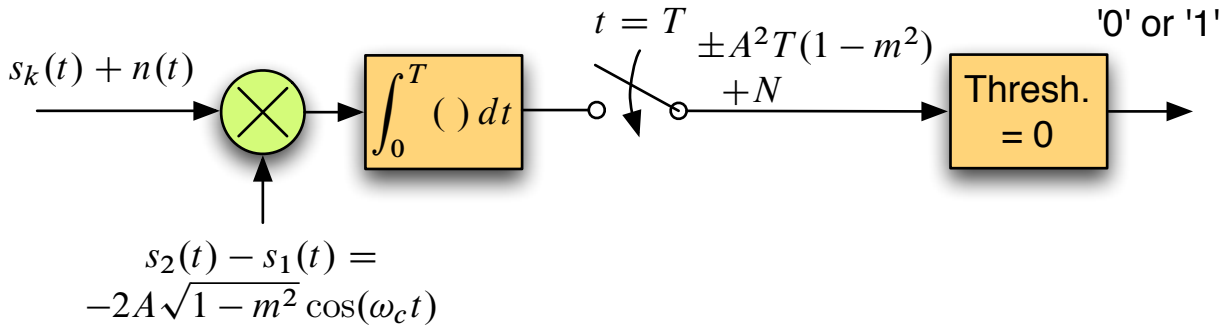
$$s_k(t) = \underbrace{Am \sin(\omega_c t)}_{\text{carrier}} - \underbrace{(-1)^k A\sqrt{1-m^2}\cos(\omega_c t)}_{\text{modulation}},\ 0 \le t \le T,$$

  for , $k = 1, 2$

  - Note: $\cos(\cos^{-1} m) = m$ and $\sin(\cos^{-1} m) = \sqrt{1-m^2}$

- The fraction of power in the carrier is m^2

- The general formulation of two correlators can be replaced by a single correlator (a sin( ) correlator is not needed; why?)



PSK correlation receiver

- Since the carrier component is orthogonal to the modulation, it is of no consequence in the correlator (under ideal conditions)
- We also note that $E_1 = E_1 = \frac{1}{2}A^2(1-m^2)T$ and the cross term leading up to R_{12} is

$$\sqrt{E_1 E_2}\rho_{12} = \int_0^T s_1(t)s_2(t)\,dt \overset{\text{can show}}{=} \frac{1}{2}A^2T(2m^2-1)$$

- Now we can calculate R_{12}

$$R_{12} = \frac{2\sqrt{E_1 E_2}}{E_1 + E_2}\rho_{12} = 2m^2 - 1$$

and

$$P_E = Q\left(\sqrt{2(1-m^2)z}\right)$$

- When $m = 0$ ($\cos^{-1} 0 = \pi/2$) we have formally *biphase-shift keying* (BPSK) and $R_{12} = -1$ and the familiar antipodal P_E equation, $Q(\sqrt{2z})$
- **Note**: BPSK is 3dB superior to ASK

BPSK with an Imperfect Phase Reference

- The need for a coherent carrier reference at the receiver is a downside of BPSK
- We model the received signal as

$$\pm A\cos[\omega_c + \theta(t)] + n(t)$$

and the correlator reference as $A\cos(\omega_c t + \hat{\theta})$

- For a static phase error, i.e. $\theta(t) = \theta$ the correlator output is

$$v(t) = \pm AT\cos(\theta - \hat{\theta}) + N = \pm AT\cos(\phi) + N$$

  where ϕ is the phase error and N has variance σ_0^2 as before

- Tracking this error through the P_E expression, we find that conditionally,

$$P_E(\phi) = Q\left(\sqrt{2z\cos^2\phi}\right)$$

- If we know bounds on ϕ we can find min and max values on P_E

- If we have a pdf model for ϕ, we can form the average P_E with *phase jitter* ϕ as

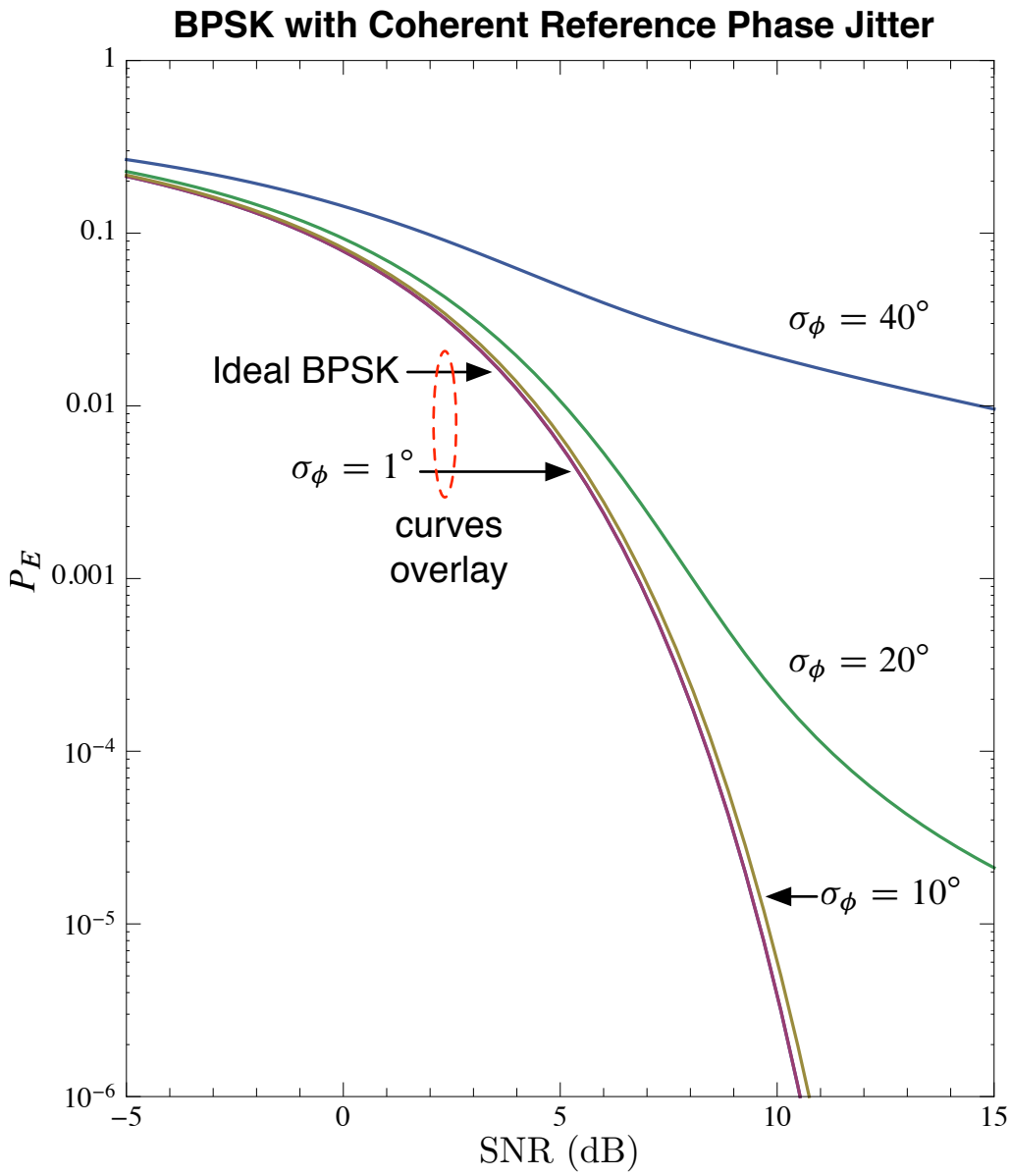
$$P_E = \int_{-\pi}^{\pi} P_E(\phi)\, f_\phi(\phi)\, d\phi$$

  where perhaps a Gaussian model is appropriate, e.g.,

$$f_\phi(\phi) = \frac{e^{-\phi^2/2\sigma_\phi^2}}{\sqrt{2\pi\sigma_\phi^2}},\ |\phi| \le \pi$$

- It might be that a phase-locked loop (PLL) is responsible for estimating the carrier phase, in which case σ_ϕ^2 is related to the design of the PLL, oscillator phase noise, platform dynamics, etc.

```
Q[x_] := 1/2 Erfc[x/Sqrt[2]];

PE[σϕdeg_, SNR_] :=
  Module[{σ2}, σ2 = (σϕdeg * π/180)^2; 2 NIntegrate[Q[Sqrt[2 SNR (Cos[ϕ])^2]] E^(-ϕ^2/(2 σ2))/Sqrt[2 π σ2], {ϕ, 0, π}]];
```



BPSK P_E averaged over a Gaussian jitter phase noise model

Frequency-Shift Keying (FSK)

- With FSK one of two carrier *tones* is sent to represent the binary information

$$s_k(t) = \begin{cases} A\cos(\omega_c t), & k = 1,\ \text{`1'} \\ A\cos(\omega_c + \Delta\omega)t, & k = 2,\ \text{`0'} \end{cases}$$

- We again assume that

$$\omega_c = \frac{2\pi n}{T} \quad \text{and} \quad \Delta\omega = \frac{2\pi m}{T},$$

  $m \neq n$ to insure orthogonality of the double frequency terms and the cross term of the correlator analysis

- The result is

$$\sqrt{E_1 E_2}\rho_{12} = \int_0^T A^2 \cos(\omega_c t)\cos[(\omega_c + \Delta\omega)t]\,dt = 0$$

  and $R_{12} = 0$

- This makes

$$P_E = Q\left(\sqrt{z}\right)$$

  which is equivalent in performance to ASK and 3dB inferior to BPSK

- From a peak power standpoint, ASK is 3dB worse that FSK

5.4 Complex Baseband Modeling

- ASK, PSK, and FSK are all bandpass signals
- If deterministic signal $x(t)$ is bandpass, it has spectrum $X(f)$ zero everywhere except for bandwidth B centered on $\pm f_0$
- From the study of the Hilbert transform the positive spectrum portion of $x(t)$ is given by

$$x_p(t) = x(t) + j\hat{x}(t)$$

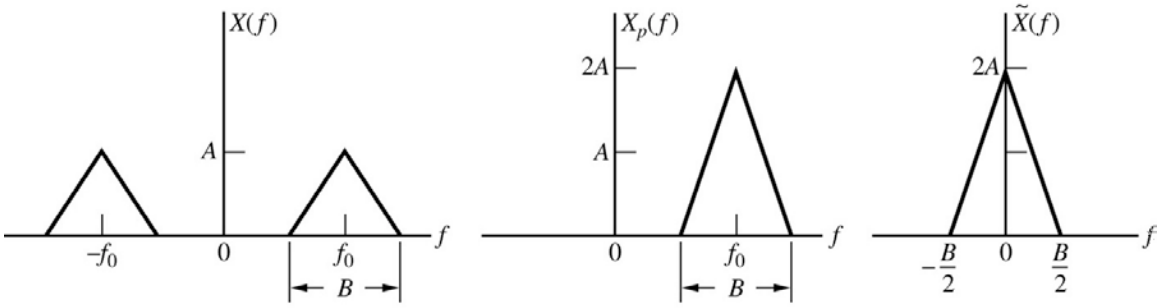
  where $\hat{x}(t)$ denotes the Hilbert transform of $x(t)$

- From the frequency translation theorem

$$x_p(t) = \tilde{x}(t)e^{j2\pi f_0 t}$$

  where $\tilde{x}(t)$ is the *complex envelope*[1] or *lowpass equivalent*[2]

  - Note: $\tilde{x}(t)$ is helpful in building Python/Julia/MATLAB simulation models and in real-time DSP implementation of digital communication systems



Complex envelope spectrum related to bandpass signal spectrum

[1] Z&T Chapter 2, pp. 87–90.

[2] J.G Proakis and M. Selehi, *Digital Communications*, fifth edition, Mc Graw Hill, New York, pp. 21–25, 2008.

- Since $x_p(t) = x(t) + j\hat{x}(t) = \tilde{x}(t)e^{j2\pi f_0 t}$, it follows that

$$x(t) = \mathrm{Re}\{\tilde{x}(t)e^{j2\pi f_0 t}\}$$

and

$$\begin{aligned} x(t) &= \mathrm{Re}\{\tilde{x}(t)\}\cos(2\pi f_0 t) - \mathrm{Im}\{\tilde{x}(t)\}\sin(2\pi f_0 t) \\ &= x_I(t)\cos(2\pi f_0 t) - x_Q(t)\sin(2\pi f_0 t), \end{aligned}$$

- Thus in rectangular form the complex envelope is defined as

$$\tilde{x}(t) \triangleq x_I(t) + jx_Q(t),$$

with $x_I(t)$ the *in-phase* component and $x_Q(t)$ the *quadrature* component

- If the channel rotates the carrier by θ, i.e., $\cos(2\pi f_0 t) \rightarrow \cos(2\pi f_0 t + \theta)$ then in the complex envelope representation of $x(t) \rightarrow y(t)$, we have

$$y(t) = \mathrm{Re}\{\tilde{x}(t)e^{j(2\pi f_0 t+\theta)}\} = \mathrm{Re}\{[\tilde{x}(t)e^{j\theta}]e^{j2\pi f_0 t}\}$$

and the complex envelope of $y(t)$ in terms of $\tilde{x}(t)$ is

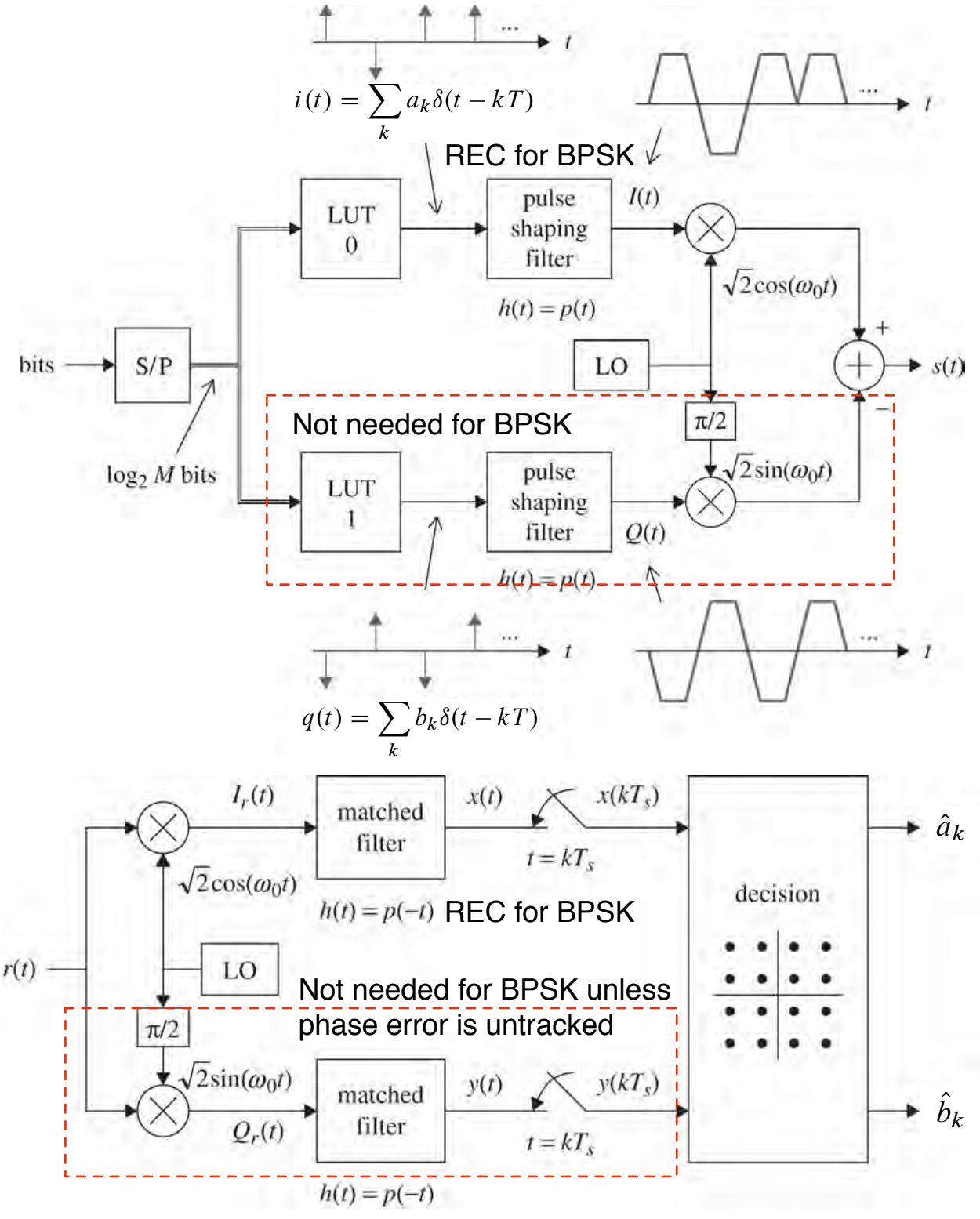
$$\tilde{y}(t) = \tilde{x}(t)e^{j\theta} = [x_I(t) + jx_Q(t)]e^{j\theta}$$

Example 5.5: **Bandpass Modulation using the Complex Envelope [Rice]**

- The complex envelope representation of BPSK ($m = 0$) on $[0, T]$ is

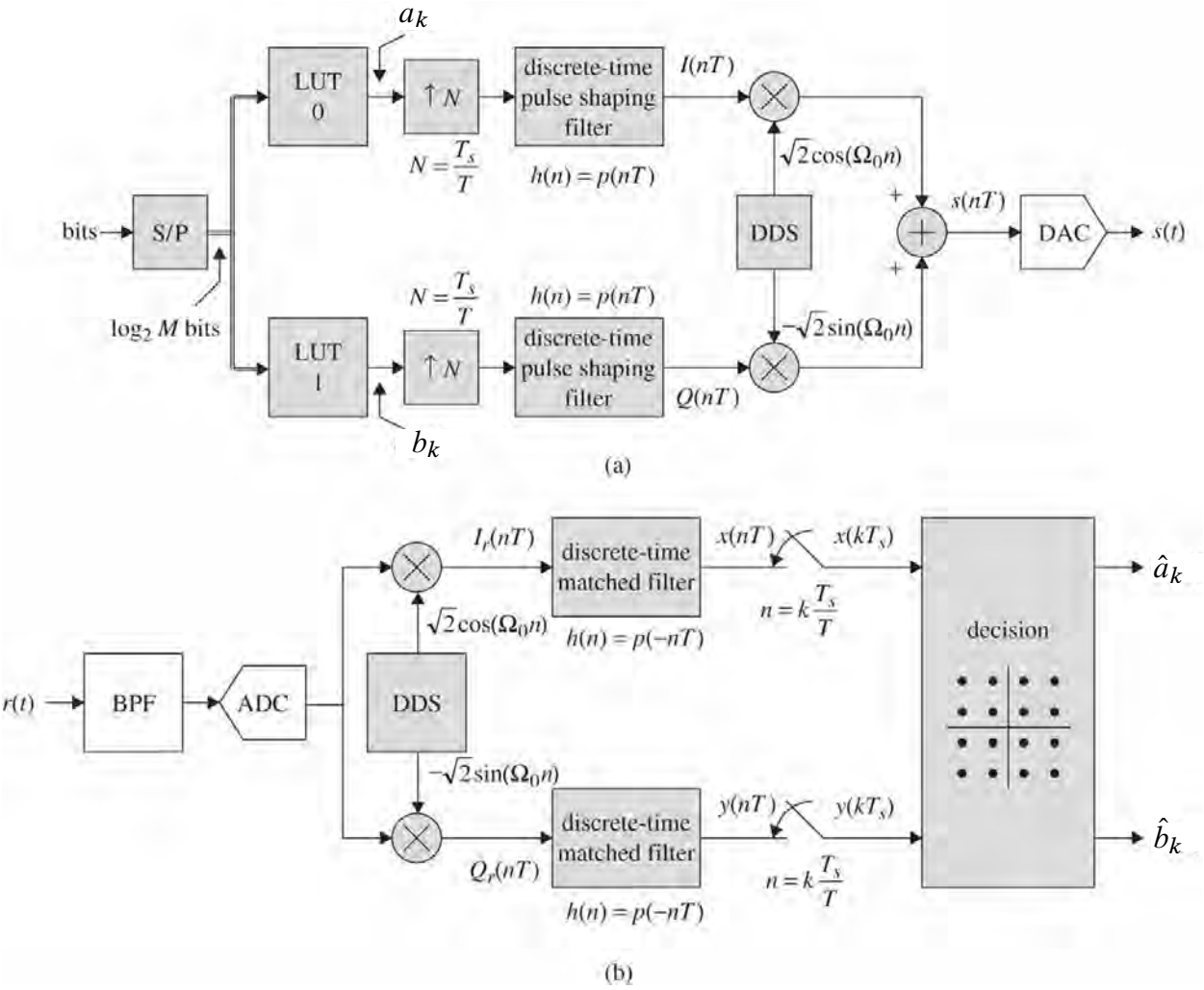
$$s(t) = A\sin(\omega_c t + a_0\pi/2) = Aa_0\cos(\omega_c t) = \mathrm{Re}\{Aa_0 e^{j\omega_c t}\}$$

where $a_k(0) \in \{-1, 1\}$ is the kth data bit, associated with the in-phase component, e.g., $s_Q(t) = 0$



Complex envelope motivated modulation and demodulation

- A full DSP implementation of in-phase/quadrature modulation and demodulation is shown below
- Modulation of the quadrature carrier ($\sin(\omega_c t)$ will be considered in Z&T Chapter 9



Complex envelope motivated DSP-based modulation and demodulation

5.4.1 Useful Complex Baseband Measurements

- Three measurements of interest at this time are

  1. The power spectral density (PSD), also referred to as the power density spectrum
  2. The matched filter output eye plot
  3. The decision statistic scatter plot

PSD Estimation

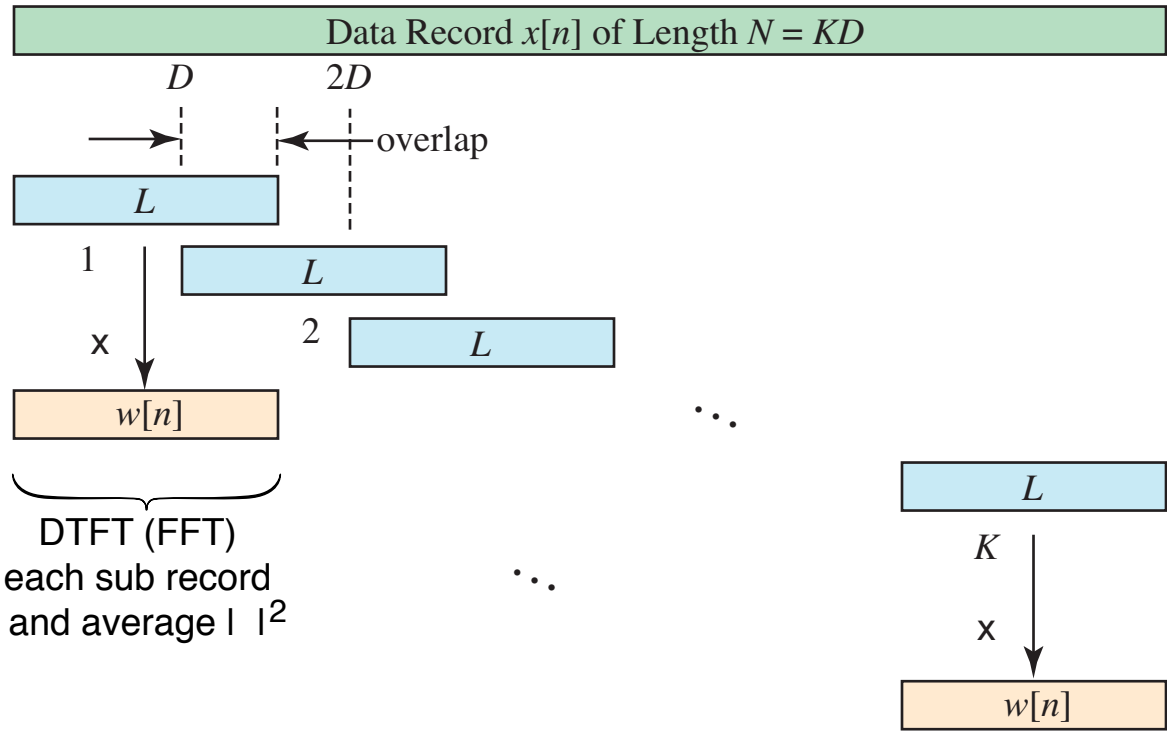
- The PSD as measured at the transmitter output is of particular interest as it allows us to see frequency occupied by the signal
- The theoretical PSD will be considered in more detail in the future
- The PSD of a digital communications waveform is most easily estimated using Welch's method of *averaging modified periodograms*[3]
  - Welch's averaging method (1967) breaks the data record down into overlapping sub-records and applies a window $w[n]$ to each sub-record
  - An overlap of say 50% will increase the correlation between sub-records, and hence impact the variance reduction capability, but it will also allow for longer sub-records, thus increasing the resolution
  - In practice the overlap may go as high as 75%

[3] Monson H. Hayes, Statistical Digital Signal Processing and Modeling, John Wiley, 1996.

– The MATLAB function `pwelch()` forms the estimate

$$\hat{P}_x(e^{j\omega}) = \frac{1}{N}\sum_{i=0}^{K-1}\left|\sum_{n=0}^{L-1} w[n]x[n+iD]e^{-jn\omega}\right|^2$$

where $w[k]$ is a window function, *Hanning*, etc.

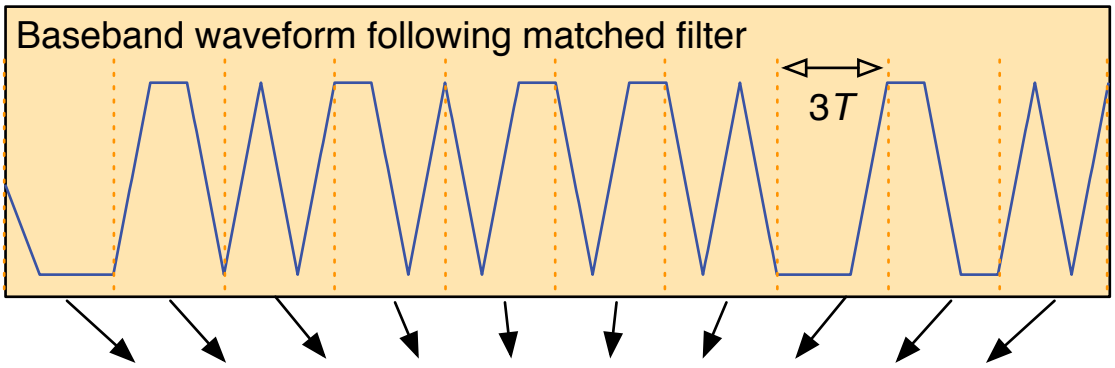


Welch's method of creating sub-records and averaging

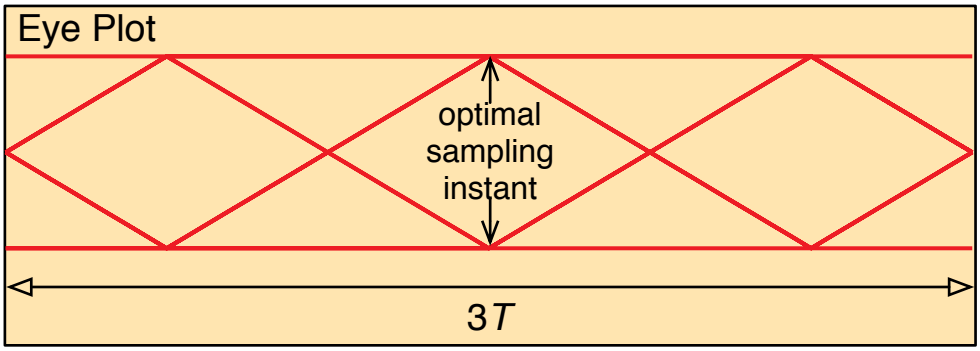
- In the example that follows a wrapper function, `spec_plot(x, NFFT,Ns)`, designed to display the two-sided PSD of both real and complex signals will be explored

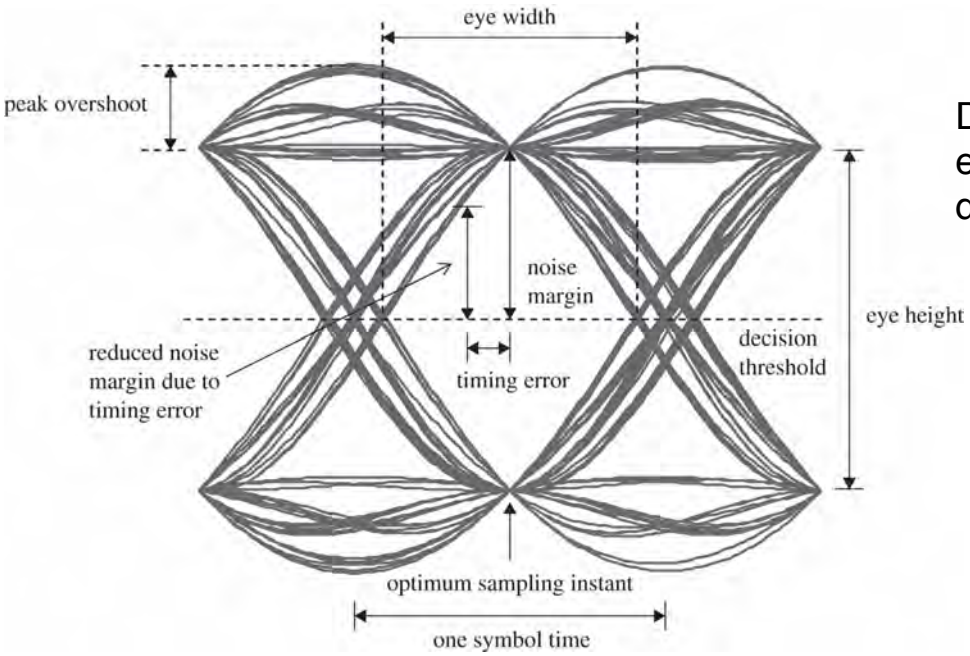
The Eye Plot

- The eye plot operates at the waveform level, typically observing the output of the matched filter or correlator, by overlaying integer multiples of the signaling interval



As shown here, slice the signal into 3-bit (symbol) long continguous segments, and overlay them





Detailed eye plot definitions

Eye plot as obtained at the output of a matched filter

- The bottom portion of the figure, taken from Rice, shows the useful information that can be extracted from a generic eye plot

The Scatter Plot

- The scatter plot typically observes the complex baseband signal at the matched filter output, following the sampler (every T seconds or N_s samples if in the discrete-time domain we have N_s samples per bit)

- Under ideal conditions, the scatter plot points lie at the nominal decision statistic location, e.g., $\pm AT$ in the case of binary antipodal signaling; assuming also that we sample at the maximum eye opening

- The ideal sample point locations constitute what is known as the *signal constellation* (more in the following chapter)

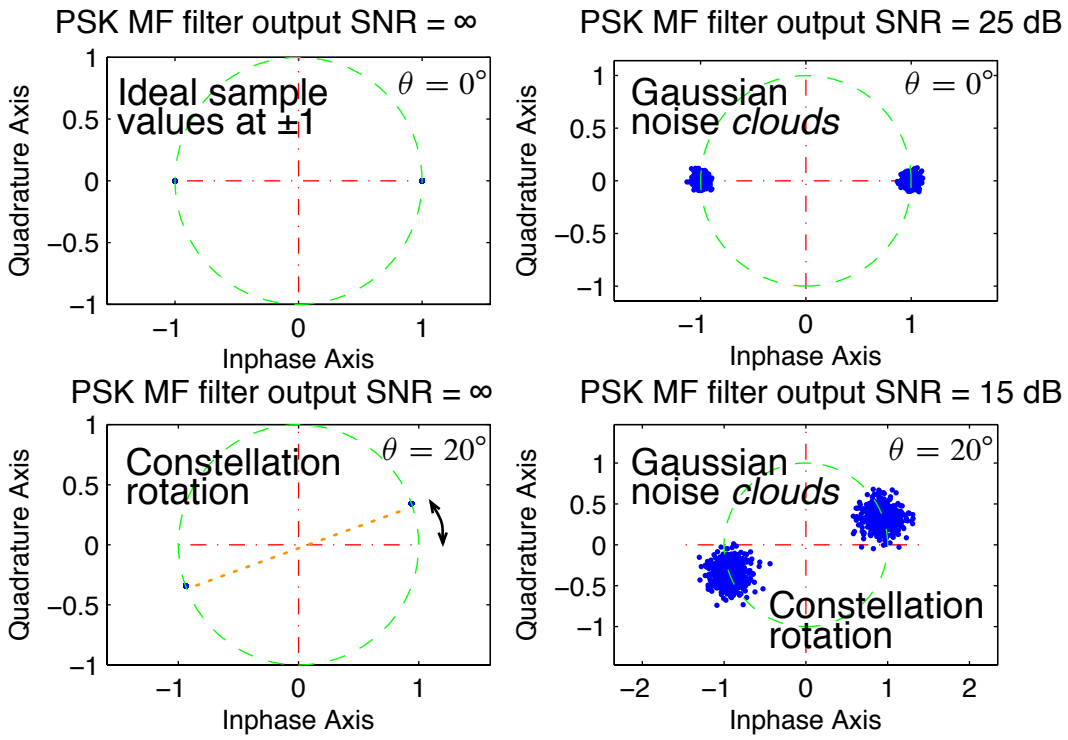
- If say, a phase error is present, the signal point set will be rotated about the origin by angle θ

- If AWGN is present there will be a *cloud* of points centered at the ideal location

- If the channel distorts the signal, then even under noise free conditions there will be a cloud of points, rather than a single point

- In general we desire the scatter plot to form a tight array of sample points, with the clusters easily discernible from each other, so that bit or symbol errors can be kept to a minimum

– With increasing AWGN the clusters eventually cross the decision boundary (the imaginary axis in the case shown below), resulting in bit/symbol errors



Scatter plot at bit rate sampled the matched filter output

Example 5.6: Complex Baseband Modeling of ASK, BPSK, and FSK in MATLAB

- A companion Jupyter notebook, `Data Transmission in Noise.ipynb`, provides similar examples in Python that make use of `scikit-dsp-comm`

- Since a phase error was considered for PSK we will incorporate that into the model as well

- **ASK**

$$\tilde{s}(t) = A \sum_k a_k \, p(t - kT) \cdot e^{j\theta(t)}$$

  with data bits $a_k \in \{0, 1\}$; Nominally $\theta(t) = 0$

- **PSK**

$$\tilde{s}(t) = A\left[\sqrt{1 - m^2} \sum_k a_k \, p(t - kT) + jm\right] \cdot e^{j\theta(t)}$$

  where m is the modulation index (0 for BPSK), $a_k \in \{-1, 1\}$

- **FSK**

$$\tilde{s}(t) = A \exp\left[j 2\pi \frac{\Delta f}{2} t \sum_k a_k \, p(t - kT)\right] \cdot e^{j\theta(t)}$$

  where Δf is the peak-to-peak frequency deviation of the carrier and $a_k \in \{-1, 1\}$

- For now in all three schemes, $p(t)$ is a REC pulse shape over $[0, T]$

- To model the AWGN channel in the bandpass domain requires the real RP $n(t)$, while in the complex baseband case we have a complex noise process

$$\tilde{n}(t) = n_I(t) + j n_Q(t)$$

- Formally $\tilde{n}(t)$ is bandlimited white noise, but for now we assume that the noise bandwidth is much wider than the signal bandwidth, so we assume a white PSD of height N_0 W/Hz

- **MATLAB Simulation Tools**: A function file collection was written to make the MATLAB experimentation an easy process
- The collection of functions used in this example and beyond is given below; the Signal Processing Toolbox™ is required

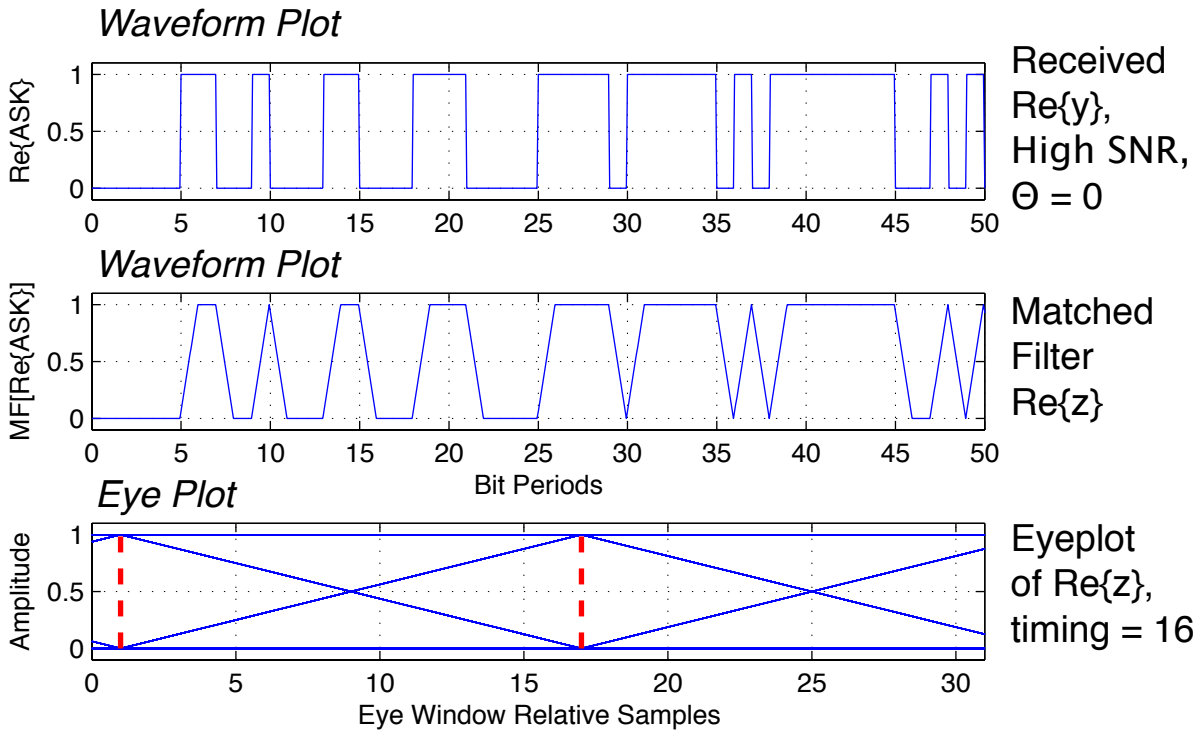
MATLAB complex baseband simulation code base

ask.m	Complex baseband binary ask modulator
ask_sim.m	Coherent ASK modem end-to-end simulation function
bit_errors.m	Count bit errors for a 1D binary signal constellation
cpx_AWGN.m	Complex baseband AWGN: y = cpx_AWGN(x,SNRdB,Ns)
diff_dec.m	Differential decoding 0/1 logic levels: d = diff_dec(c)
diff_enc.m	Differential encoding 0/1 logic levels: c = diff_enc(d,IC)
discrim.m	Complex baseband discriminator: y = discrim(x)
eyeplot.m	Basic eyeplot function: eyeplot(data, W, offset)
eyeplot_mark.m	Draw a red timing mark at Ns intervals of an eye plot
filter1.m	Custom filter for fsk_sim( ) designed using fdatool
fsk.m	Complex baseband binary FSK modulator
fsk_sim.m	Coherent and NC FSK modem end-to-end simulation function
psk.m	Complex baseband binary PSK modulator
psk_sim.m	Coherent Binary FSK modem end-to-end simulation function
scatter_plot.m	Plot the complex baseband data sequence as a constellation
spec_plot.m	Estimated PSD of real and complex baseband signals

- We will focus on the top level functions `ASK_sim`, `PSK_sim`, and `FSK_sim`
- `ASK_sim`: Set a break-point at line 39 so we can observe a variety of signals

```
>> [Pe,errors,N_RecBits,z] = ask_sim(100,0*pi/180,100,16);
39  z_det = real(z);
K>> plot([0:length(y)-1]/16,real(y))
K>> subplot(311)
K>> plot([0:length(y)-1]/16,real(y))
K>> subplot(312)
K>> plot([0:length(y)-1]/16,real(z))
```

```
K>> subplot(313)
>> [Pe,errors,N_RecBits,z] = ask_sim(100,0*pi/180,10000,16);
////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 10000
       BEP: 0.00e+00
////////////////////////////////////////////////////////
>> eyeplot_mark(real(z(1:5000)),2*16,46,16,16)
```

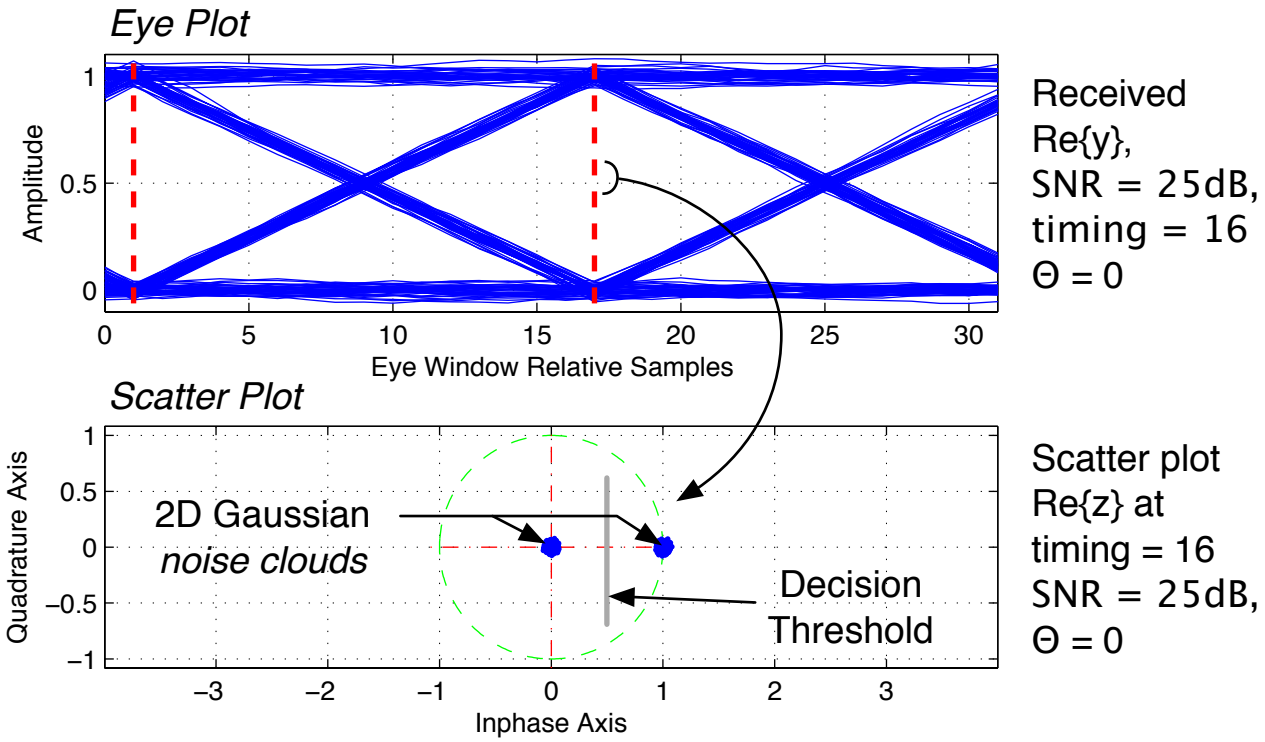


ASK received waveform (real part) before and after the matched filter

- From the above we see that a sampling instant of 16 places samples at the maximum eye opening
- Now we drop the SNR and observe the impact on both the eye plot and the scatter plot

```
>> [Pe,errors,N_RecBits,z] = ask_sim(25,0*pi/180,10000,16);
////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 10000
       BEP: 0.00e+00
```

```
//////////////////////////////////////////////////////
>> subplot(211)
>> eyeplot_mark(real(z(1:5000)),2*16,46,16,16)
>> subplot(212)
>> plot(cos(t),sin(t),'g--'); hold on; scatter_plot(z,16,16);
>> axis equal; hold off;
```



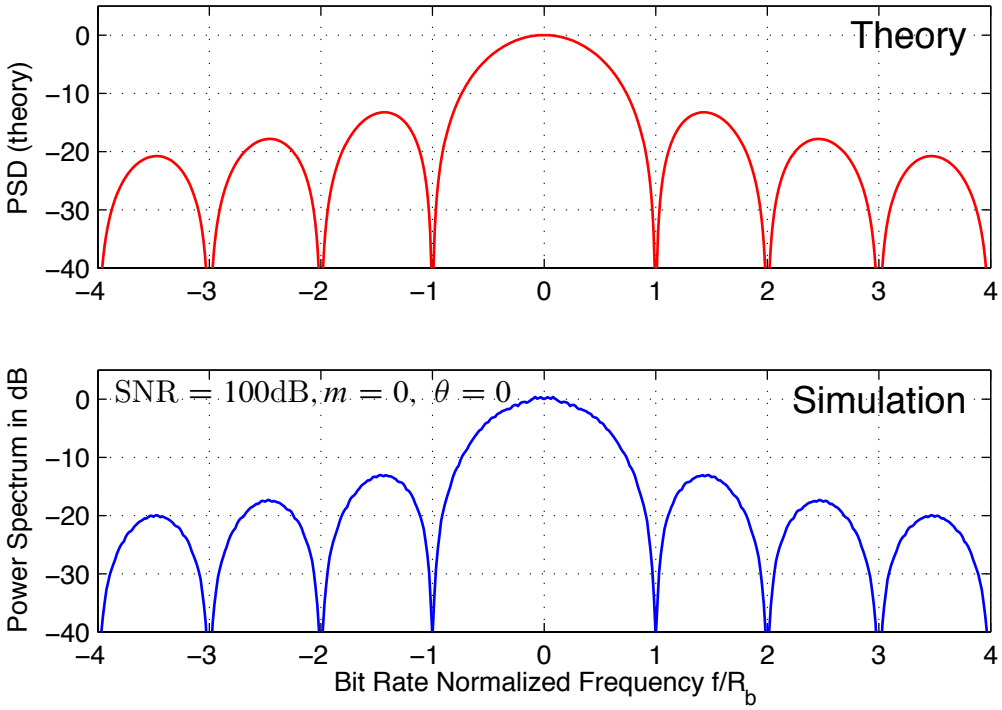
Eyeplot and scatter plot with SNR lowered to 25 dB; optimum timing

- PSK_sim: Set a break-point at line 43 so we can observe all signals in the system model
- First consider the PSD of the transmit signal $x[n]$
- From the random pulse train analysis of notes Chapter 6, we know that with $A = 1$

$$S_x(f) = T\,\mathrm{sinc}^2(fT)$$

- We compare measured and theory using the function `spec_plot`

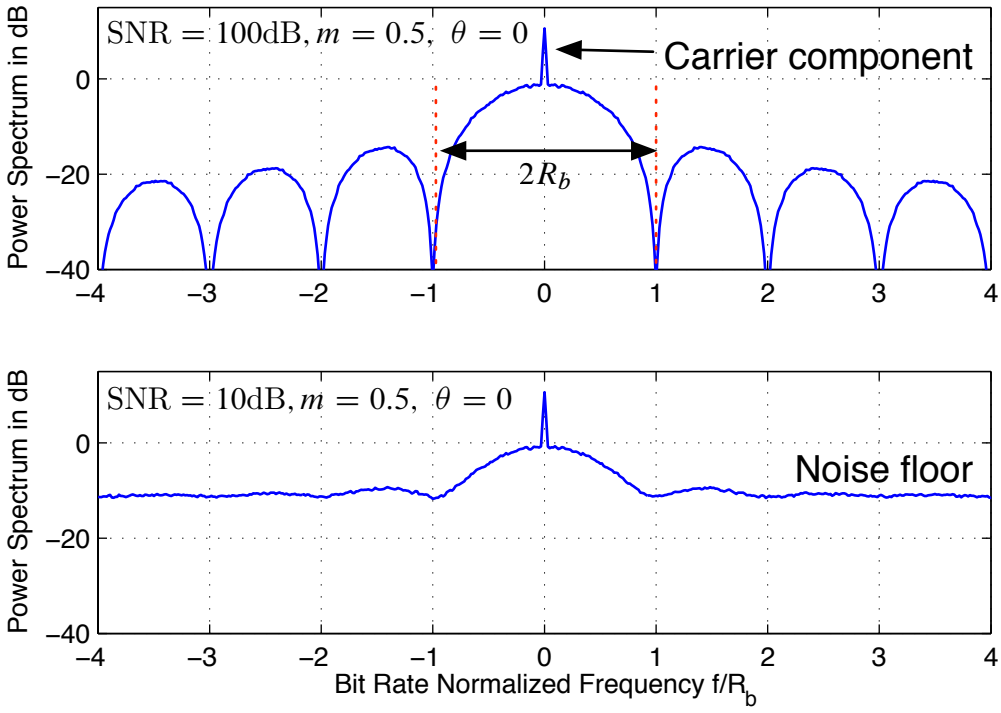
```
>> subplot(211)
>> f = -8:0.01:8;
>> Sxt = sinc(f).^2;
>> plot(f,10*log10(Sxt),'r')
>> subplot(212)
>> [Pe,errors,N_RecBits,z] = psk_sim(100,0,0*pi/180,50000,16);
K>> spec_plot(x,2^10,16);
```



Theoretical and experimental BPSK PSD at complex baseband

- Set $m > 0$ to include pure carrier (at complex baseband dc) and raise noise level

```
>> [Pe,errors,N_RecBits,z] = psk_sim(10,0.5,0*pi/180,50000,16);
43  z_det = real(z);
K>> subplot(211)
K>> spec_plot(x,2^10,16);
K>> subplot(212)
K>> spec_plot(y,2^10,16);
```



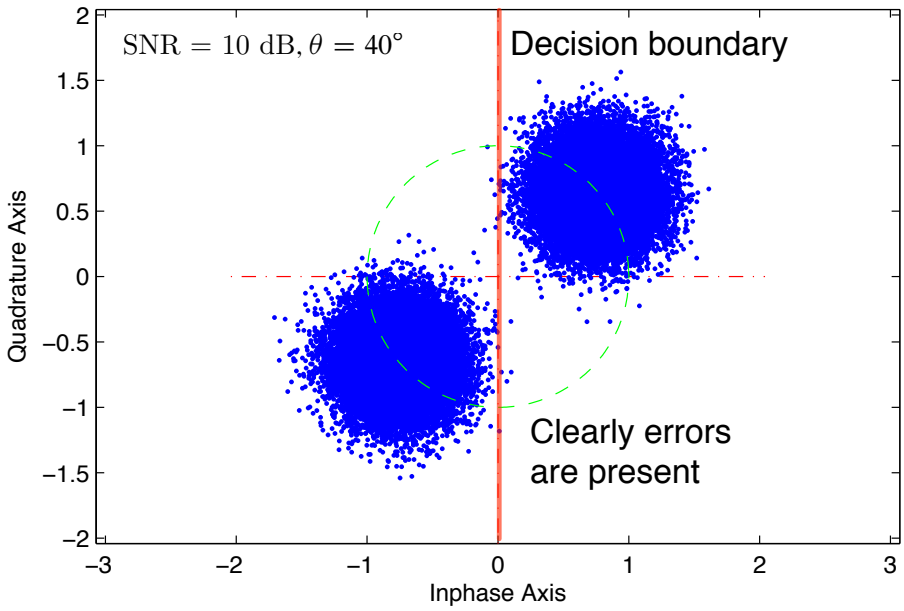
Impact of $m > 0$ and AWGN at 100 and 10 dB

- We now compare the experimental BEP with our theoretical models
- The function `psk_sim` estimates the bit error probability via *Monte-Carlo* simulation, which is just a fancy name for repeated trials
- For an AWGN channel we need to obtain at least 100 error events to have reasonable *confidence* in the estimate $\hat{P}_E$ of P_E
- At higher SNR values this requires longer simulation runs, and due to RAM limitations means that multiple calls to the function `psk_sim` may be required
  - Writing a top level function or script that calls `psk_sim` is a good approach, this way you can log the data to disk

as the simulation runs, and you can queue multiple SNR data points for running your PC unattended

- Here I simulated BPSK with $\theta = 0$ and then with a static phase error, $\theta = 40°$

```
>> [Pe,errors,N_RecBits,z] = ...
     psk_sim(5,0,0*pi/180,100000,16); % 5 dB, no phase error
>> [Pe,errors,N_RecBits,z] = ...
     psk_sim(5,0,40*pi/180,100000,16); % 5 dB, 40  deg phase error
```



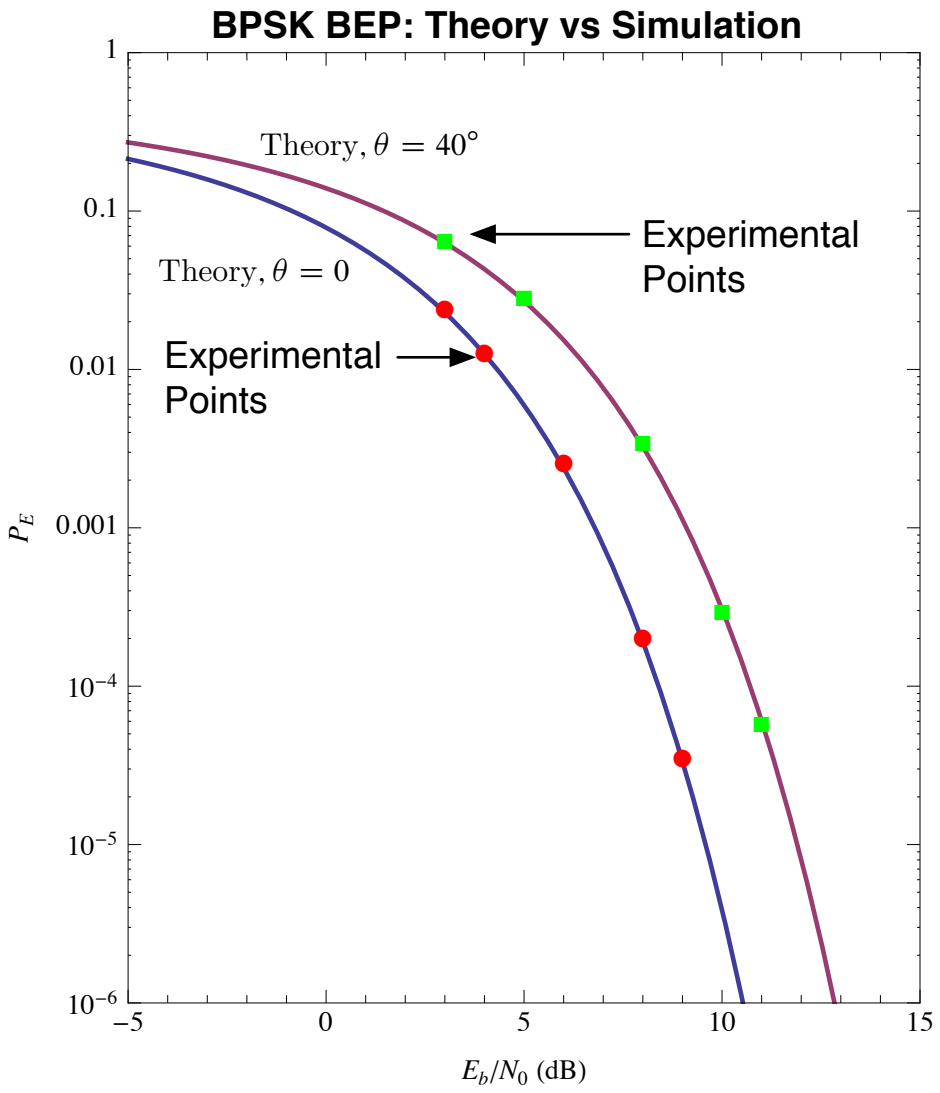
Scatter plot of z at optimum sampling point

- For theoretical comparison, recall that

$$P_E(\theta) = Q\left(\sqrt{2z\cos^2(\theta)}\right)$$

- The experimental results were obtained in MATLAB and then compiled in Mathematica for plotting

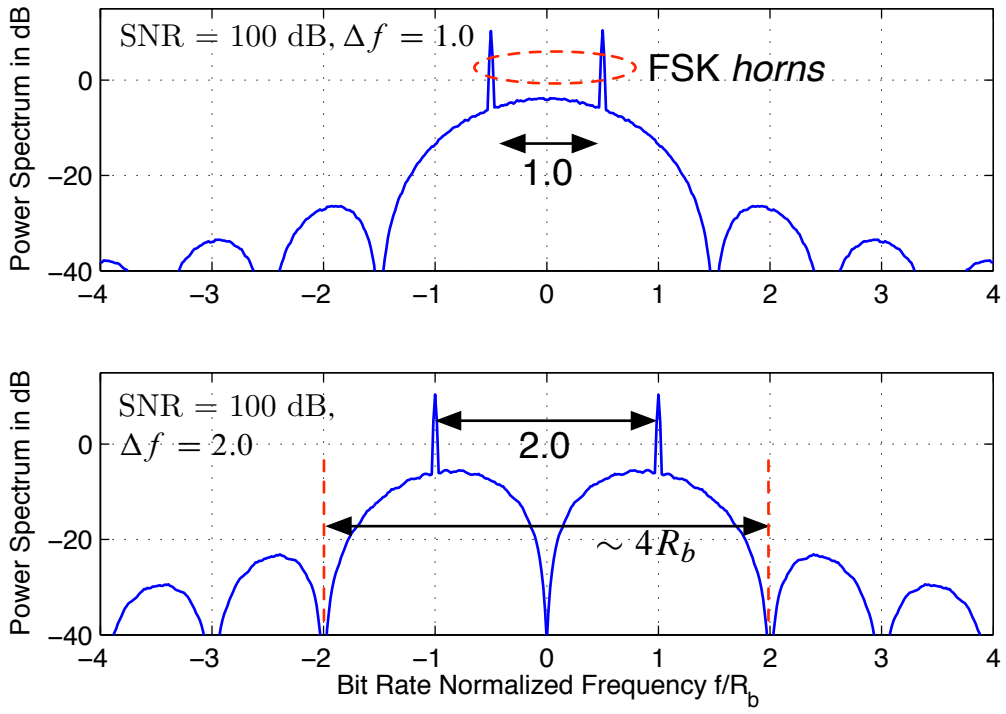
- The following plot shows that theory and experiment are in very close agreement!



BPSK BEP theoretical and experimental comparison

- FSK_sim: Set a break-point at line 44 so we can observe all signals in the system model
- First consider the PSD of the transmit signal $x[n]$

```
>> [Pe,errors,N_RecBits,z] = fsk_sim(100,1.0,50000,16,'coherent');
44          d_hat = (sign(z(timing:Ns:end))+1)/2; % 0,1 logic levels
K>> subplot(211)
K>> spec_plot(x,2^10,16);
K>> subplot(212)
>> [Pe,errors,N_RecBits,z] = fsk_sim(100,2.0,50000,16,'coherent');
44          d_hat = (sign(z(timing:Ns:end))+1)/2; % 0,1 logic levels
K>> spec_plot(x,2^10,16);
```



FSK PSD for $\Delta f = 1.0$ and 2.0

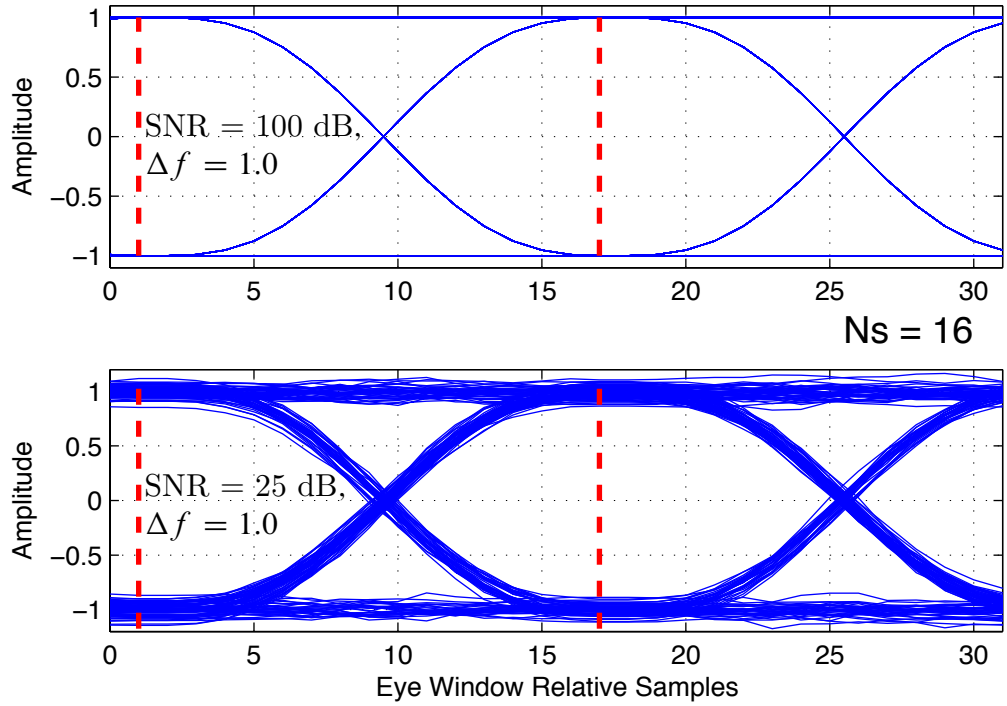
- When we select the `'coherent'` receiver option the decision variable `z` is formed as the difference between two correlator outputs

```
case 'coherent'
        % Correlator intgrator via rectangle pulse filter
        b_REC = ones(1,Ns)/Ns; % make filter unity gain at dc
        % correlation with s1(t)
        r1 = filter(b_REC,1,y.*x_ref);
        % correlation with s2(t)
        r2 = filter(b_REC,1,y.*conj(x_ref));
        % Form decision statistic
        z = real(r1 - r2);
        % Decision logic
        d_hat = (sign(z(timing:Ns:end))+1)/2; % 0,1 logic levels
        % Error detect
        [Pe,errors,N_RecBits] = bit_errors(data,d_hat);
```

- The eye plot at two SNR values is obtained as

```
>> subplot(211)
>> [Pe,errors,N_RecBits,z] = fsk_sim(100,1.0,5000,16,'coherent');
```

```
////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 5000
       BEP: 0.00e+00
////////////////////////////////////////////////////////
>> eyeplot_mark(real(z(1:5000)),2*16,46,16,16)
>> subplot(212)
>> [Pe,errors,N_RecBits,z] = fsk_sim(25,1.0,5000,16,'coherent');
////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 5000
       BEP: 0.00e+00
////////////////////////////////////////////////////////
>> eyeplot_mark(real(z(1:5000)),2*16,46,16,16)
```



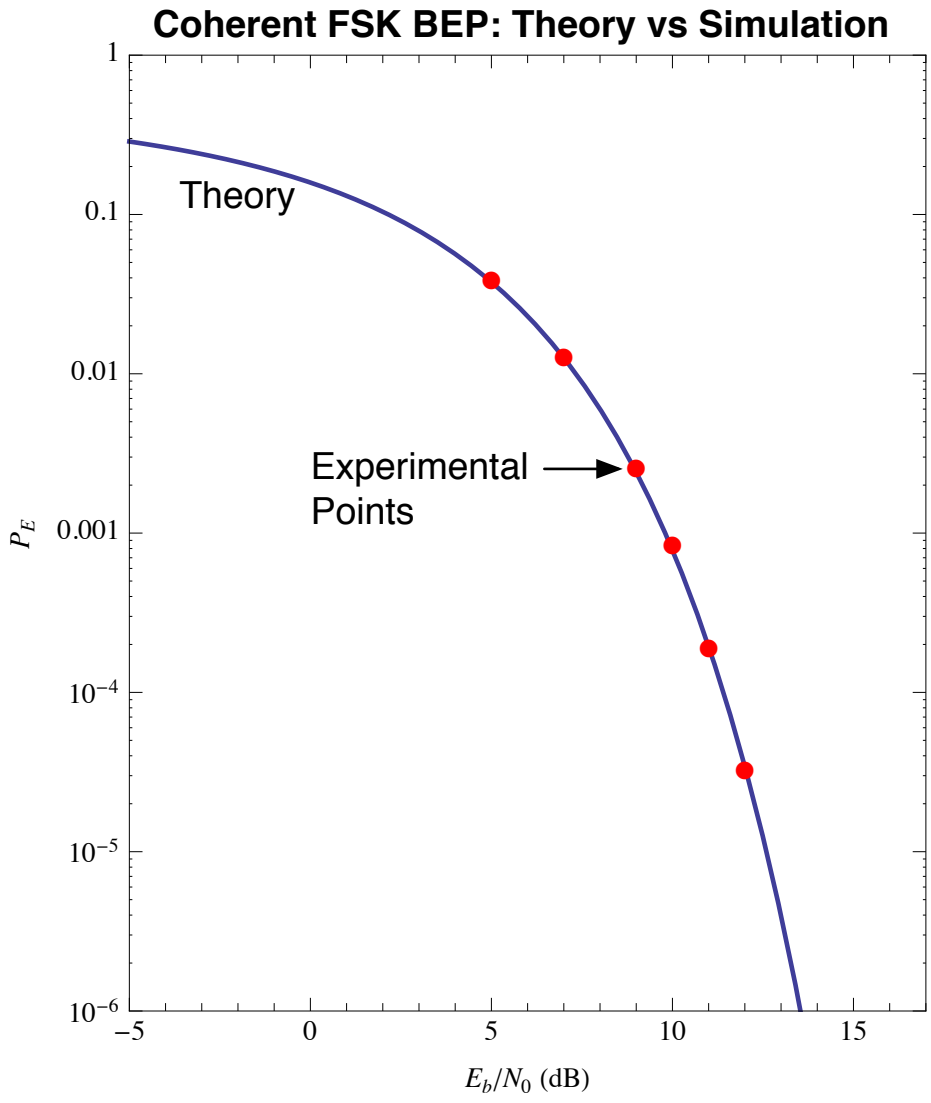
Eye plot of the coherent receiver decision waveform

- We now compare the experimental BEP with our theoretical models
- For theoretical comparison, recall that

$$P_E = Q\left(\sqrt{z}\right)$$

- The experimental results were obtained in MATLAB and then compiled in Mathematica for plotting

- The following plot shows that theory and experiment are in very close agreement!



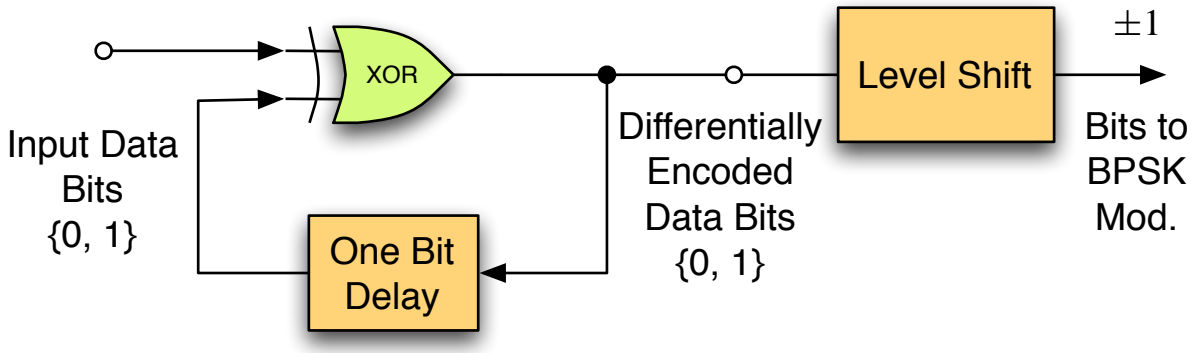
FSK BEP theoretical and experimental comparison

5.5 Modulation Schemes Not Requiring a Coherent Reference

- The need for a receiver-based coherent reference signal is a drawback to the ASK, PSK, and FSK we have just studied
- It is possible to design the modulation such that a coherent reference is not needed at the receiver
- Two examples we will consider here are *differentially coherence phase-shift keying* (DPSK) and *noncoherent* FSK (NCFSK)

5.5.1 Differential Phase-Shift Keying (DPSK)

- The idea with DPSK is encode the transmit data differentially, such the carrier phase of the previous bit can be used as a reference to demodulate the present bit
- The first step is to *differentially encode* the message sequence a_k using an XOR gate and a one-bit delay



Differential encoding

- MATLAB functions for performing encoding and decoding are given below

```
%-----------------------------------------------------------------------%
function c = diff_enc(d,IC)
% function c = diff_enc(d,IC)
% Inputs are assumed to be 0/1
% IC is the initial condition for the encoder, typically zero.
%
% Mark Wickert, December 2006

c_old = IC;
c = zeros(size(d));
for n=1:length(d)
    c(n) = xor(d(n),c_old);
    c_old = c(n);
end

%-----------------------------------------------------------------------%
function d = diff_dec(c)
% function d = diff_dec(c)
% Inputs are assumed to be 0/1
%
%
% Mark Wickert, December 2006

d = xor(c,filter([0 1],1,c));
```

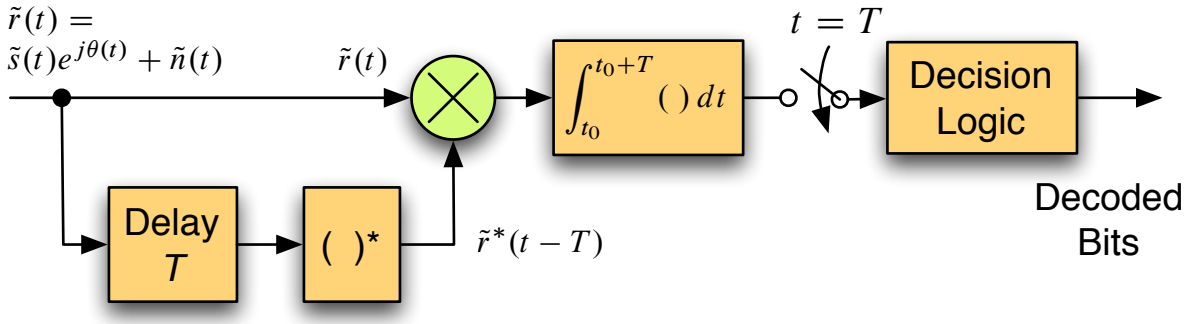
- The *differential decoding* operation can be done by XOR'ing the present bit with the previous bit

```
>> data = randi(0:1,1,15)
data =  0     1     1     1     1     1     0     0     1     0
        0     0     0     0     0

>> x = diff_enc(data,0)
x =     0     1     0     1     0     1     1     1     0     0
        0     0     0     0     0

>> y = diff_dec(x)
y =     0     1     1     1     1     1     0     0     1     0
        0     0     0     0     0
```

- With DSPK the decoding will actually be implemented by the receiver structure

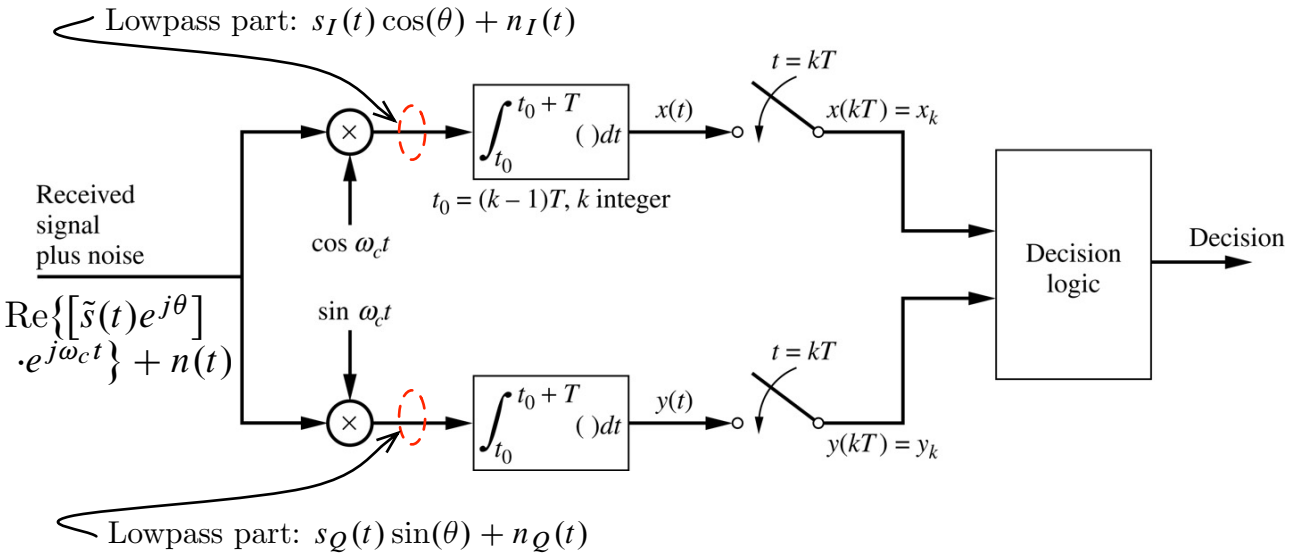
- A logical choice is some form of delay-and-multiply operation

- Since the carrier phase will be unknown (e.g. θ), a fully complex delay and multiply is needed (assuming complex baseband signals are at hand)
- It turns out that this receiver is still only sub-optimum; the text solution, but not in complex baseband form



Suboptimum complex baseband DPSK receiver

- The IQ structure of the text is used for analysis



Optimum IQ DPSK receiver for BEP analysis

- The test statistic in the receiver is the RV

$$l = x_k x_{k-1} + y_k y_{k-1}$$

- The sign of l dictates the signal sequence having been sent (diff. encoded): Two like bits in a row makes $l > 0$ while two different bits in a row makes $l < 0$
- Since it is assumed that $\theta(t) \simeq \theta = \text{const.}$ we can assume that $\theta = 0$ in further analysis
- To obtain the BEP for the case of like bits, we must find

$$P_E = P\left[x_k x_{k-1} + y_k y_{k-1} < 0 | (+1,+1)\text{sent}, \theta = 0\right]$$

- We assume that $\omega_c T$ is an integer multiple of 2π, which is approximated when $\omega_c T \gg 1$
- We have four random variables to analyze, x_0, x_1, y_0, y_1:

$$x_0 = \frac{AT}{2} + n_1 \quad \text{and} \quad y_0 = n_3$$

where

$$n_1 = \int_{-T}^{0} n(t)\cos(\omega_c t)\,dt \text{ and } n_3 = \int_{-T}^{0} n(t)\sin(\omega_c t)\,dt$$

- To obtain the BEP for the case of like bits, we must find

$$P_E = P\left[x_k x_{k-1} + y_k y_{k-1} < 0 | (+1,+1)\text{sent}, \theta = 0\right]$$

- Similarly,

$$x_1 = \frac{AT}{2} + n_2 \quad \text{and} \quad y_1 = n_4$$

where

$$n_2 = \int_{0}^{T} n(t)\cos(\omega_c t)\,dt \text{ and } n_4 = \int_{0}^{T} n(t)\sin(\omega_c t)\,dt$$

- By construction n_1, n_2, n_3, and n_4 each zero mean with variance $N_0T/4$, mutually uncorrelated, hence independent since Gaussian

- SEP calculation is now of the form

$$P_E = P\left[\left(\frac{AT}{2} + n_1\right)\left(\frac{AT}{2} + n_2\right) + n_3 n_4 < 0\right]$$

- Further manipulation yields

$$P_E = P\left[\left(\frac{AT}{2} + w_1\right)^2 + w_3^2 < w_2^2 + w_4^2\right]$$

  where

$$w_1 = \frac{n_1}{2} + \frac{n_2}{2},\ w_2 = \frac{n_1}{2} - \frac{n_2}{2},\ w_3 = \frac{n_3}{2} + \frac{n_4}{2},\ w_4 = \frac{n_3}{2} - \frac{n_4}{2}$$

- The w_i's are Gaussian, mutually uncorrelated with zero mean and variance $N_0T/8$

- Taking the square root on each side of the inequality yields two new random variables

$$R_1 = \sqrt{\left(\frac{AT}{2} + w_1\right)^2 + w_3^2} \quad \text{and} \quad R_2 = \sqrt{w_2^2 + w_4^2}$$

- The R_1 is Ricean and the RV R_2 is Rayleigh (notes Chapter 4, text Chapter 6)

- The desired probability is obtain by integrating the joint pdf $f_{R_1R_2}(r_1, r_2)$ such that $P[R_1 < R_2]$

- The region in the R_1R_2-plane is given below, where we also make use of the fact that R_1 and R_2 are independent

$$P_E = \int_0^\infty \left[\int_{r_1}^\infty f_{R_2}(r_2)\, dr_2 \right] f_{R_1}(r_1)\, dr_1$$

- The remaining work requires characterizing the Ricean and Rayleigh pdfs and then working from integral tables; see the text for details

- It can be shown that

$$P_E = \frac{1}{2}\exp\left(-\frac{A^2T}{2N_0}\right) = \frac{1}{2}\exp\left(-\frac{E_b}{N_0}\right)$$

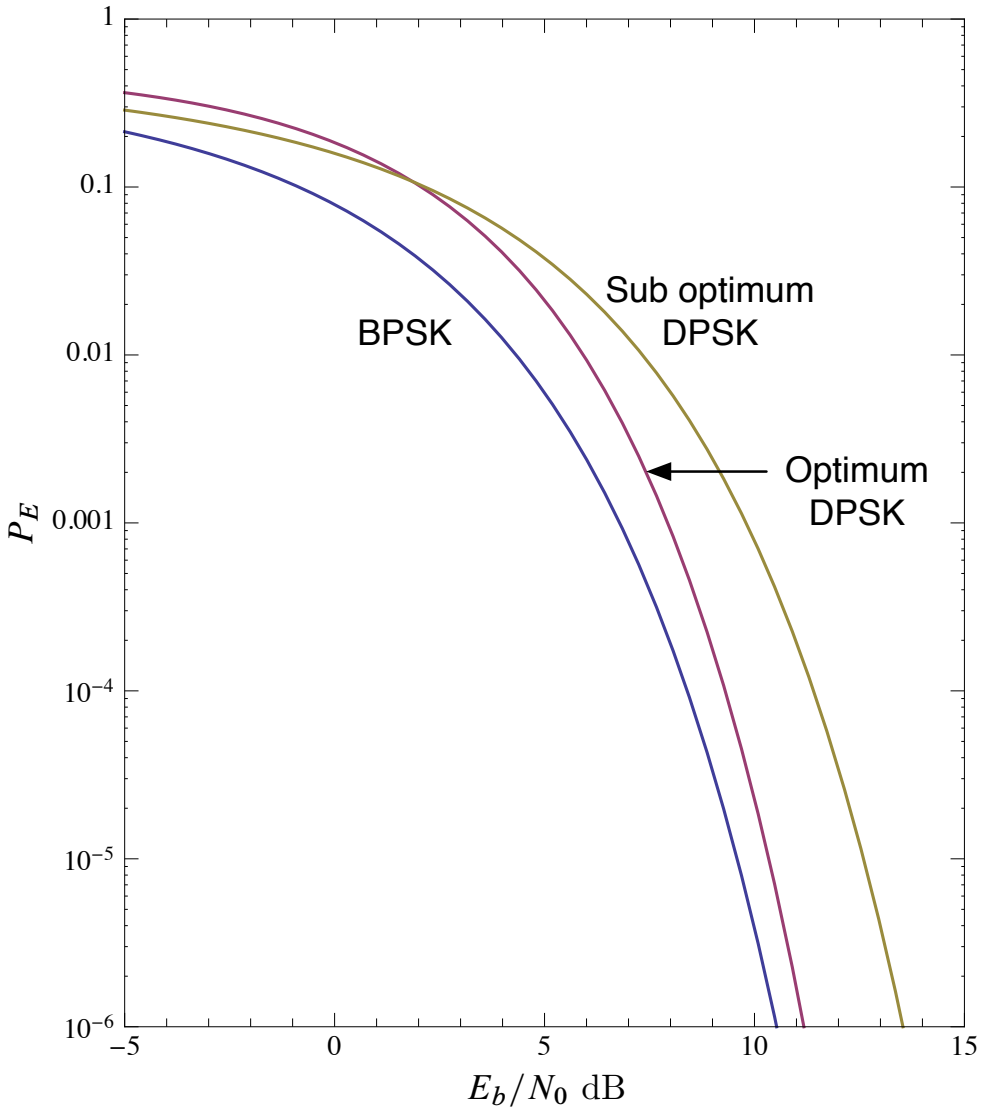
  where $E_b = A^2T/2$ ($\pm A$ bits are on a cos( ) carrier)

- The suboptimum detector, using a simple delay-and-multiply, achieves approximately

$$P_E \simeq Q\left(\sqrt{\frac{E_b}{N_0}}\right)$$

  when the input bandwidth is $B = 2/T$ and E_b/N_0 is large

- We now compare BPSK, simple DPSK, and the optimum DPSK BEP



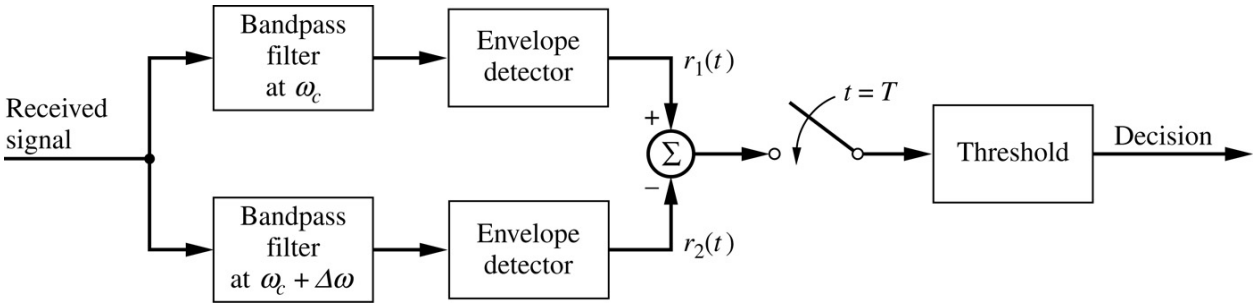
DPSK BEP comparisons

5.5.2 Noncoherent FSK

- Noncoherent FSK refers to the receiver implementation in this case
- The transmitter in no different from the earlier discussion, except now we consider the carrier phase to be arbitrary
- Using the earlier notation,

$$s_1(t) = A\cos(\omega_c t + \theta),\ 0 \le t \le T$$
$$s_2(t) = A\cos\left[(\omega_c + \Delta\omega)t + \theta)\right],\ 0 \le t \le T$$

- We also need to choose $\Delta\omega$ large enough so that the spectra of $s_1(t)$ and $s_2(t)$ do not overlap too much
- The noncoherent receiver employs bandpass filters centered at ω_c and $\omega_c + \Delta\omega$ followed by envelope detectors



Envelope detection of FSK

- The analysis here is not as complex as that of DPSK
- Each bandpass filter output is similar to that of ASK, that is a switched carrier when that data bit has the respective frequency active (we assume the adjacent FSK frequency has negligible output on the opposite filter

- The statistics of $R_1 = r_1(T)$ and $R_2 = r_2(T)$ are Ricean or Rayleigh, depending which frequency is active

- Assuming $s_1(t)$ is active

$$f_{R_1}(r_1) = \frac{r_1}{N} e^{-(r_1^2+A^2)/(2N)} I_0\left(\frac{Ar_1}{N}\right),\ r_1 \geq 0$$
$$f_{R_2}(r_2) = \frac{r_2}{N} e^{-r_2^2/(2N)},\ r_2 \geq 0$$

- As in the DPSK case we integrate over the R_1R_2-plane to find

$$\begin{aligned} P(E|s_1(t)) &= P[R_2 > R_1] \\ &= \int_0^\infty f_{R_1}(r_1) \left[\int_{r_1}^\infty f_{R_2}(r_2)\, dr_2\right] dr_1 \end{aligned}$$

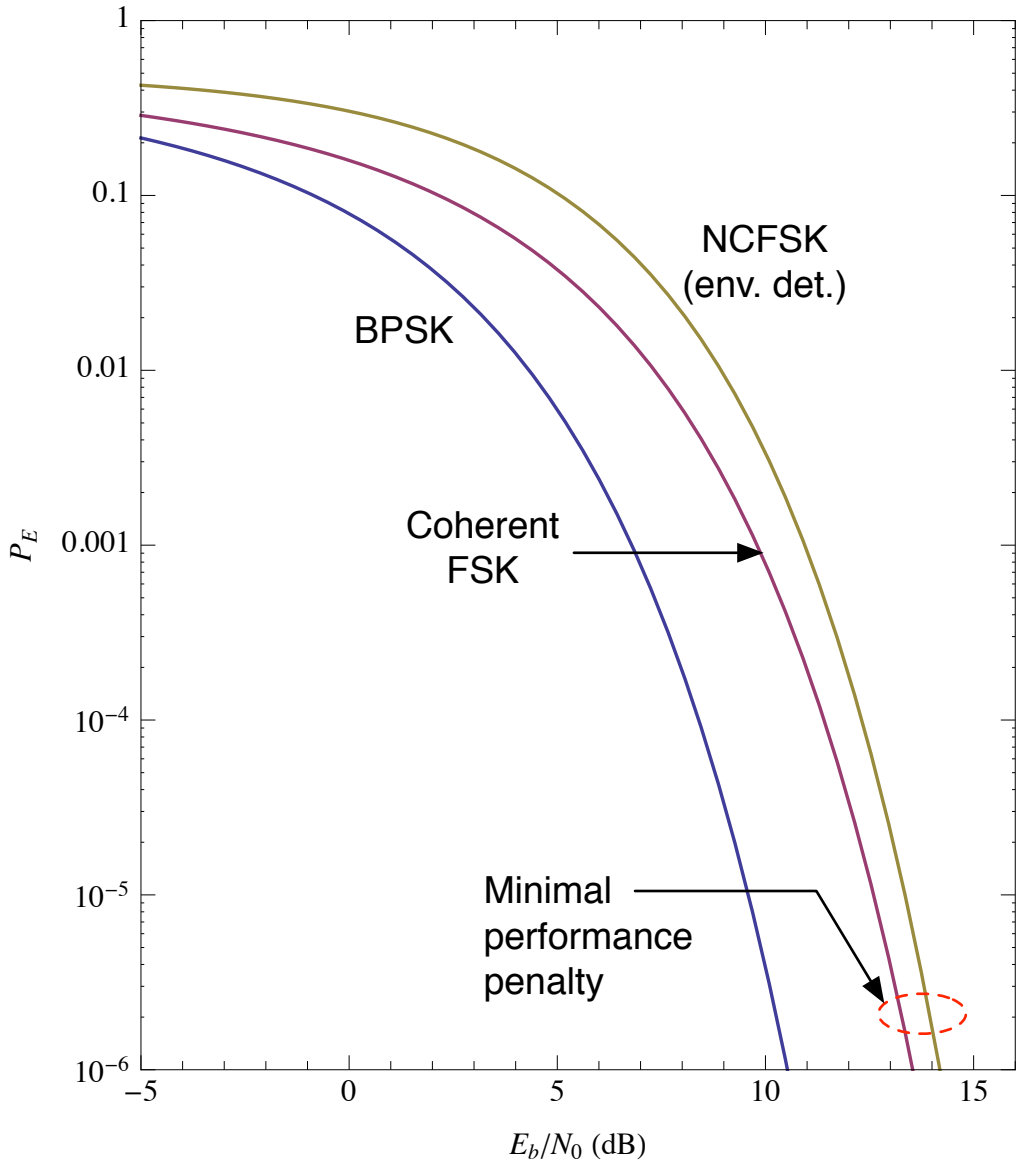
- The inner Rayleigh integral evaluates to $\exp(-r_1^2/(2N))$ and $s_1(t)$ and $s_2(t)$ are symmetrical, so

$$P_E = e^{-z} \int_0^\infty \frac{r_1}{N} I_0\left(\frac{Ar_1}{N}\right) e^{-r_1^2/N}\, dr_1 \overset{\text{tables}}{=} \frac{1}{2} \exp\left(-\frac{z}{2}\right)$$

- Since for large SNR coherent FSK has

$$P_E \simeq \frac{\exp(-z/2)}{\sqrt{2\pi z}},\ z \gg 1$$

the noncoherent detector using an envelope detector is degraded only by the factor $\sqrt{2/(\pi z)}$



NCFSK with envelope detector BEP compare

- The performance of the envelope detector is quite good, however, the bandwidth occupied by the FSK had to be increased, i.e., typically Δf needs to be $\sim 2/T$ Hz so the null-to-null bandwidth is now

$$B_{\text{NCFSK}} = \frac{4}{T} = 4R_b$$

- Another popular noncoherent receiver is the *limiter discriminator*
- From the study of analog frequency modulation (FM) we know that a discriminator is a device which produces output proportional to the instantaneous frequency deviation of the input
- At complex baseband a discriminator can be implemented using DSP

```
function disdata = discrim(x)
% function disdata = discrimf(x)
% x is the received signal in complex baseband form
%
% Mark Wickert

xI=real(x); % xI is the real part of the received signal
xQ=imag(x); % xQ is the imaginary part of the received signal
N=length(x); % N is the length of xI and xQ
b=[1 -1];   % filter coefficients
a=[1 0];    % for discrete derivative
der_xI=filter(b,a,xI);  % derivative of xI,
der_xQ=filter(b,a,xQ);  % derivative of xQ
% normalize by the squared envelope acts as a limiter
disdata=(xI.*der_xQ-xQ.*der_xI)./(xI.^2+xQ.^2);
```

- To understand the operation of `discrim()` start with a general angle modulated signal and obtain the complex envelope

$$\begin{aligned} x_c(t) &= A_c \cos(\omega_c t + \phi(t)) \\ &= \mathrm{Re}\{A_c e^{j\phi(t)} e^{j\omega_c t}\} \\ &= A_c \mathrm{Re}\{[\cos\phi(t) + j\,\sin\phi(t)] e^{j\omega_c t}\} \end{aligned}$$

- The complex envelope is

$$\tilde{x}_c(t) = \cos\phi(t) + j\,\sin\phi(t) = x_I(t) + jx_Q(t)$$

where x_I and x_Q are the in-phase and quadrature signals respectively

- A frequency discriminator obtains $d\phi(t)/dt$
- In terms of the I and Q signals,

$$\phi(t) = \tan^{-1}\left(\frac{x_Q(t)}{x_I(t)}\right)$$

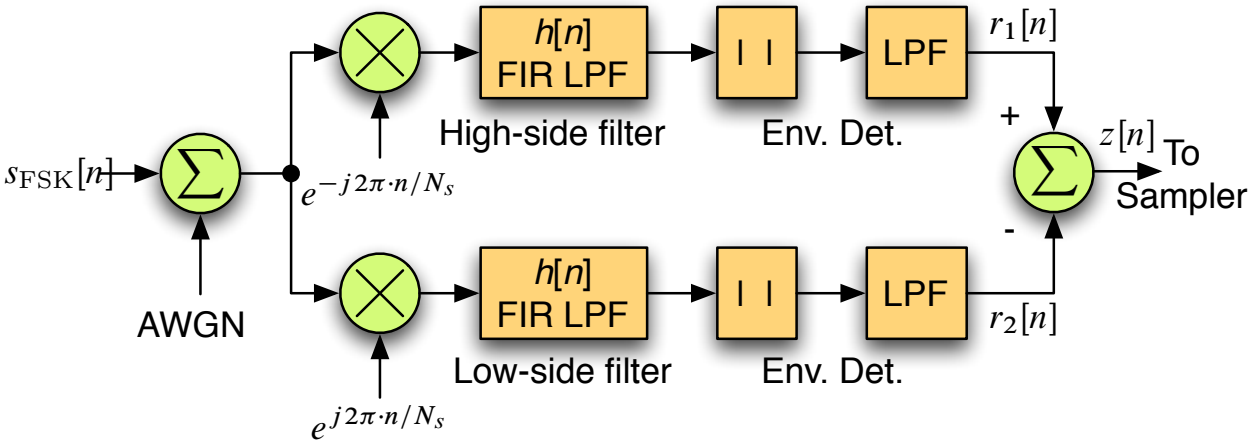
- The derivative of $\phi(t)$ is

$$\begin{aligned}\frac{d\phi(t)}{dt} &= \frac{1}{1+(x_Q(t)/x_I(t))^2}\frac{d}{dt}\left(\frac{x_Q(t)}{x_I(t)}\right)\\ &= \frac{x_I(t)x_Q'(t) - x_I'(t)x_Q(t)}{x_I^2(t)+x_Q^2(t)}\end{aligned}$$

- In the DSP implementation $x_I[n] = x_I(nT)$ and $x_Q[n] = x_Q(nT)$, where T is the sample period
- The derivatives, $x_I'(t)$ and $x_Q'(t)$ are approximated by the *backwards difference* $x_I[n] - x_I[n-1]$ and $x_Q[n] - x_Q[n-1]$ respectively
- Assuming an AWGN channel, the signal entering the discriminator needs to be bandpass filtered at the carrier or lowpass filtered at complex baseband
- The P_E analysis of the discriminator can be found in the literature

Example 5.7: NCFSK MATLAB Simulation

- In an earlier example the FSK simulation function `fsk_sim` was developed
- This function contains three receiver implementations:
  1. `'coherent'`
  2. `'bpf_env_det'`
  3. `'lim_discrim'`
- We now consider the non-optimized performance of the last two
- Since we have a complex baseband simulation framework for FSK already available, the bandpass filter envelope detector scheme is implemented using a lowpass filter with complex frequency translation



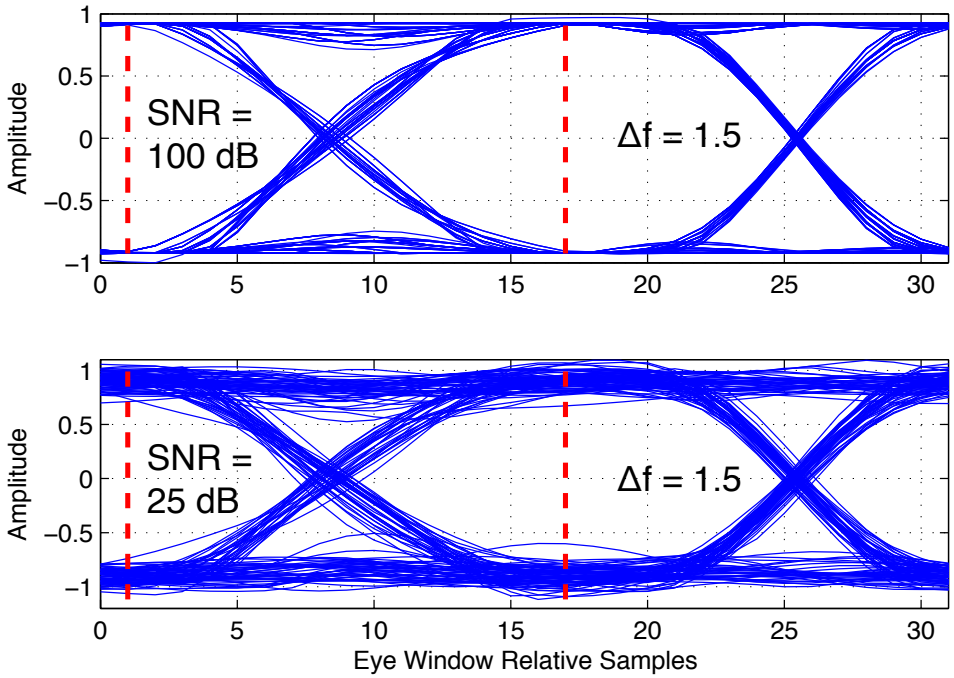
Non-coherent FSK receiver using a BPF and envelope detector

- The m-code

```
case 'bpf_env_det'
    % BPF Envelope detector receiver
    % Design the receive filter using fdatool written m-file function
    Hd1 = filter1;
    % Process complex BB signal through filters via complex frequency
    % translation to +/-
    n = 0:length(x)-1;
    r1 = filter(Hd1.numerator,1,y.*exp(-j*2*pi*n*(Df/2)/Ns));
    r2 = filter(Hd1.numerator,1,y.*exp(j*2*pi*n*(Df/2)/Ns));
    % Form decision statistic
    z = abs(r1)-abs(r2);
    % Decision logic
    d_hat = (sign(z(timing:Ns:end))+1)/2; % 0,1 logic levels
    % Error detect
    [Pe,errors,N_RecBits] = bit_errors(data,d_hat);
```

- We set $\Delta f = 1.5$ and consider the eye plot at $z[n]$ with and without noise

```
>> subplot(211)
>> [Pe,errors,N_RecBits,z] = fsk_sim(100,1.5,5000,16,'bpf_env_det');
/////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 4998
       BEP: 0.00e+00
/////////////////////////////////////////////////////////
>> eyeplot_mark(real(z(1:5000)),2*16,46,16,16)
>> subplot(212)
>> [Pe,errors,N_RecBits,z] = fsk_sim(25,1.5,5000,16,'bpf_env_det');
/////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 4998
       BEP: 0.00e+00
/////////////////////////////////////////////////////////
>> eyeplot_mark(real(z(1:5000)),2*16,46,16,16)
```

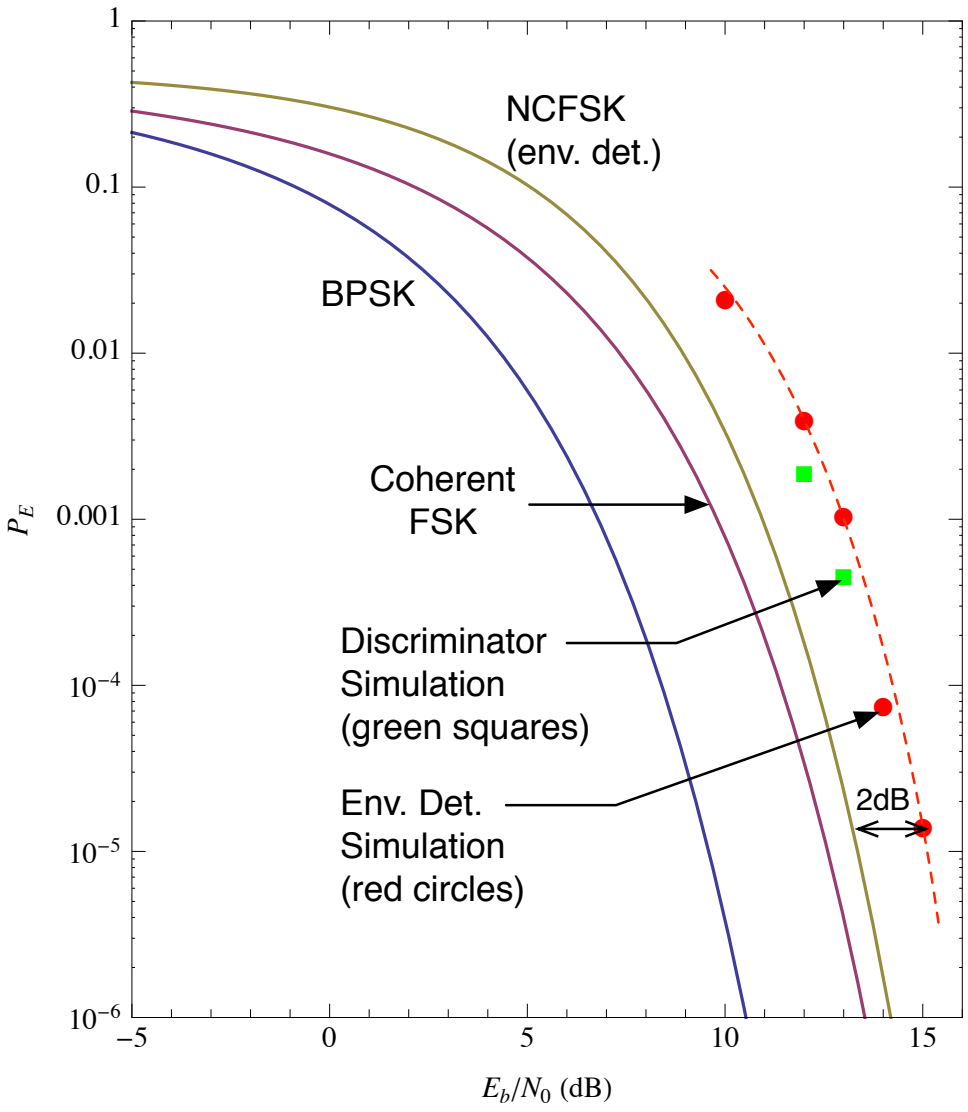


Non-coherent FSK receiver eye plots

- A similar eye plot is obtained for the discriminator receiver, except here the discriminator directly processes the complex baseband signal to a real signal
- A lowpass prefilter is required to reduce the impact of noise on the nonlinear discriminator processing, e.g.,

```
case 'lim_discrim'
        % Limiter discriminator receiver
        %//////////////////////////////////
        % Design the receive filter using fdatool written m-file function
        Hd2 = filter2;
        b2 = fir1(32,2*1.5/Ns);
        % Process complex BB signal through receive filter
        %r1 = filter(Hd2.numerator,1,y);
        r1 = filter(b2,1,y);
        % Form decision statistic via limiter discriminator
        z = discrim(r1);
        % Decision logic
        d_hat = (sign(z(timing:Ns:end))+1)/2; % 0,1 logic levels
        % Error detect
        [Pe,errors,N_RecBits] = bit_errors(data,d_hat);
```

- BEP simulation results are obtained for both non-coherent receivers and compared with the theoretical model presented earlier
- The system is not optimized
- The simulation points lie about 2 dB away from theory



Non-coherent FSK BEP simulation and theory comparison

5.6 M-ary PAM

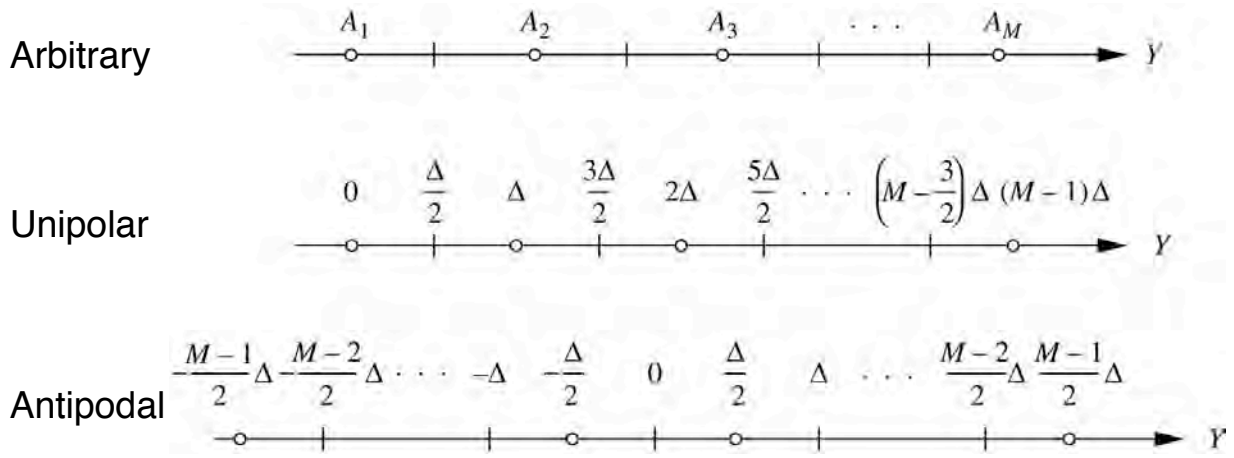
- M-ary modulation is taken up more fully in the next chapter, but extending ASK is a simple task that is considered here

- We extend the signal set from simple on-off keying to M possible amplitude levels, typically M is a power of two

$$s(t) = \sum_k a_k \, p(t - kT),$$

where now a_k can take on one of M different amplitude values, $A_1, A_2, \ldots, A_M$, and $p(t)$ is a unit energy pulse shape, i.e.,

$$E_p = \int_0^T p^2(t)\, dt = 1$$



M-ary ASK amplitude considerations

- The received signal in AWGN is

$$y(t) = s(t) + n(t) = \sum_k a_k \, p(t - kT) + n(t)$$

- A correlator receiver using pulse shape $p(t)$ has output following the sampler of the form

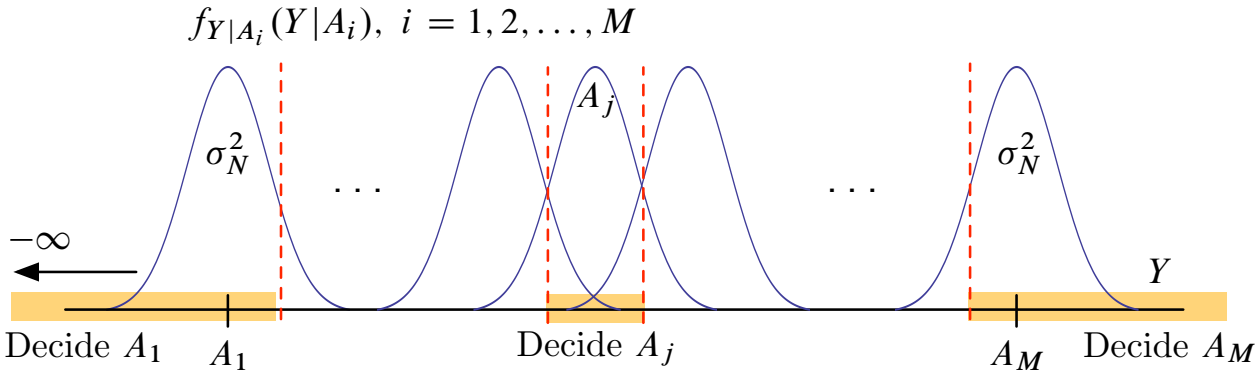
$$Y = \int_0^T \big[s(t) + n(t)\big]p(t)\,dt = A_i + N,\ i = 1, 2, \ldots, M$$

on the interval $[t_0, t_0 + T]$, and

$$N = \int_0^T n(t)\,p(t)\,dt$$

which is a Gaussian RV having zero mean and variance $\sigma_N^2 = N_0/2$

- The decision statistic is a conditional RV taking one of M pdf's



Decision statistic conditional pdf's

- To calculate the error probability we need to make a specific assumption about the A_i values and then determine the optimum threshold values

- We assume the A_i's are equally spaced case and occur with equally probability, thus the thresholds for the inner values lie halfway between A_1 and A_2, etc.

- Let $\Delta = A_i - A_{i-1}$ as shown in an earlier figure
- Under these assumptions, the probability of error for the end symbols (1 & M) can be written as

$$\begin{aligned} P(E|A_1 \text{ sent}) &= P(E|A_M \text{ sent}) \\ &= 1 - \int_{-\infty}^{\Delta/2} \frac{\exp(-\eta^2/N_0)}{\sqrt{\pi N_0}}\, d\eta \\ &= Q\left(\frac{\Delta}{\sqrt{2N_0}}\right) \end{aligned}$$

- For the interior symbols we have

$$\begin{aligned} P(E|A_j \text{ sent}) &= 1 - \int_{-\Delta/2}^{\Delta/2} \frac{\exp(-\eta^2/N_0)}{\sqrt{\pi N_0}}\, d\eta \\ &= 2Q\left(\frac{\Delta}{\sqrt{2N_0}}\right), \; j = 2, \ldots, M-1 \end{aligned}$$

- The probability of a symbol error is thus

$$\begin{aligned} P_E &= \frac{1}{M}\sum_{j=1}^{M} P(E|A_j \text{ sent}) \\ &= \frac{2(M-1)}{M} Q\left(\frac{\Delta}{\sqrt{2N_0}}\right) \end{aligned}$$

- The remaining manipulation of P_E formulas involves working with the average signal energy

$$E_{\text{ave}} = \frac{1}{M}\sum_{j=1}^{M} E_j = \frac{1}{M}\sum_{j=1}^{M} A_j^2$$

- We consider two cases: (1) unipolar or just PAM and (2) bipolar or antipodal PAM
- For unipolar PAM the amplitude values are $A_j = (j-1)\Delta$ and for antipodal PAM the amplitude values are $A_j = (j-1)\Delta - (M-1)/2 \cdot \Delta$
- It can be shown that

$$E_{\text{ave,PAM}} = \frac{(M-1)(2M-1)\Delta^2}{6}$$
$$E_{\text{ave,Anti. PAM}} = \frac{(M-1)\Delta^2}{12}$$

- The corresponding error probability expressions reduce to

$$P_{E,\text{PAM}} = \frac{2(M-1)}{M} Q\left(\sqrt{\frac{3E_{\text{ave}}}{(M-1)(2M-1)N_0}}\right)$$
$$P_{E,\text{Anti. PAM}} = \frac{2(M-1)}{M} Q\left(\sqrt{\frac{6E_{\text{ave}}}{(M^2-1)N_0}}\right)$$

  - Note that when $M = 2$ in the antipodal case, the P_E expression reduces to binary antipodal, as expected

- We would like to relate the symbol error probability (SEP) to the bit error probability
- We assume that M is a power of two, so $M = 2^m$ for some integer m and $m = \log_2 M$ The energy per bit is thus

$$E_b = \frac{E_{\text{ave}}}{M} = \frac{E_{\text{ave}}}{\log_2 M} \quad \text{or} \quad E_{\text{ave}} = E_b \log_2 M$$

- **Gray Coding**: To minimize the number of bits that error when a symbol errors, we chose the $\log_2 M$ bits representing each A_j such that adjacent values differ by only one bit, e.g., `000`, `001`, ..., `110`, `111`, etc.

- When a symbol error occurs at moderate SNR values it is more likely that an adjacent symbol is decoded, so only one bit error occurs

- Under the Gray coding assumption

$$P_b \simeq \frac{P_{\text{symbol}}}{\log_2 M} = \frac{P_s}{\log_2 M}$$

- Finally then we can approximate the BEP for PAM as

$$P_{b,\text{PAM}} = \frac{2(M-1)}{M \log_2 M} Q\left(\sqrt{\frac{3\log_2(M) E_b}{(M-1)(2M-1)N_0}}\right)$$
$$P_{b,\text{Anti. PAM}} = \frac{2(M-1)}{M \log_2 M} Q\left(\sqrt{\frac{6\log_2(M) E_b}{(M^2-1)N_0}}\right)$$

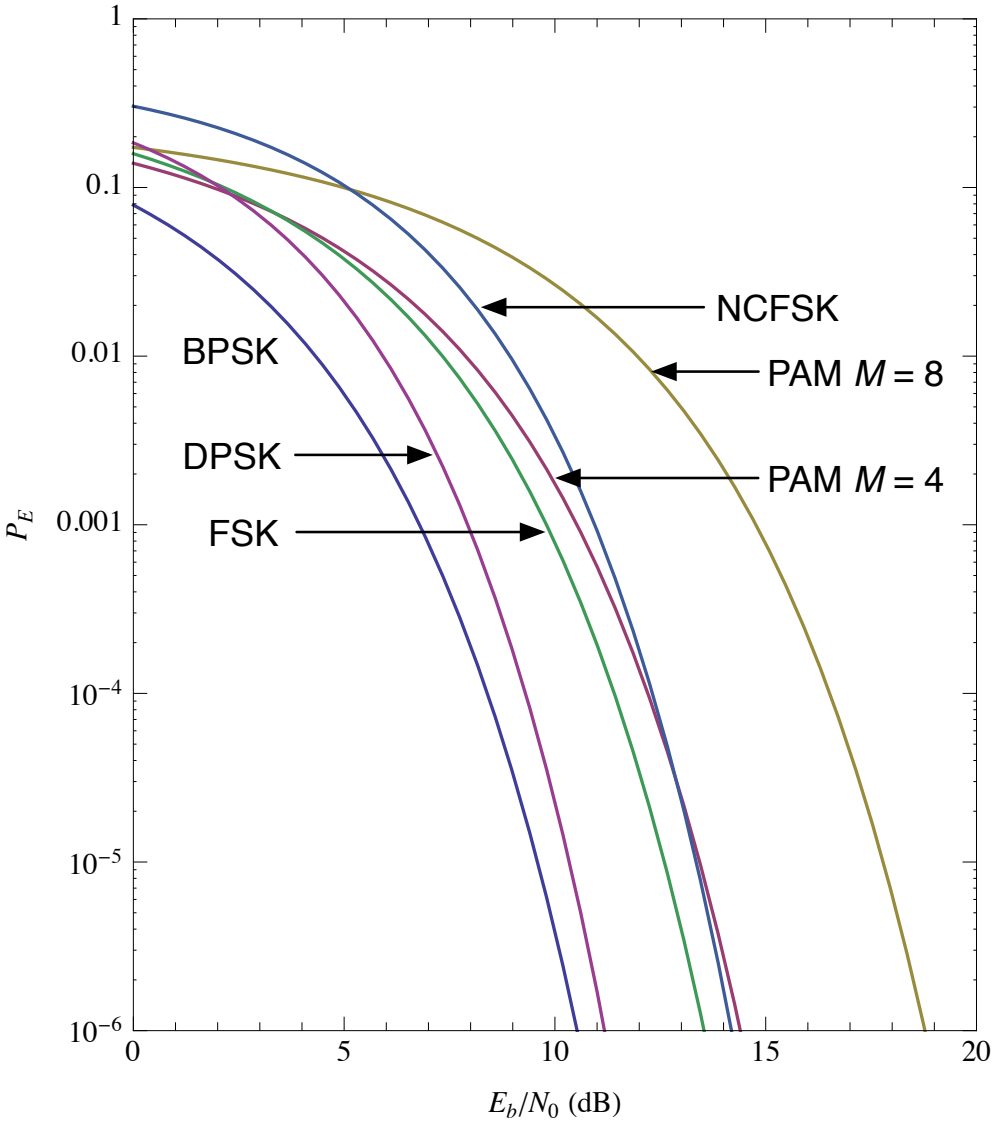
- The bandwidth occupied by PAM is identical to BPSK, that is, with regard to the RF spectrum the null-to-null spacing will be $2R_s$, where R_s is the symbol rate, thus by way of comparison

$$B_{\text{RF, BPSK}} = \frac{2}{T_b} = 2R_b$$
$$B_{\text{RF, PAM}} = \frac{2}{T_s} = \frac{2}{T_b \log_2 M} = \frac{2R_b}{\log_2 M},$$

which is a bandwidth efficiency of $0.5 \log_2 M$ bps/Hz

5.7 Comparison of Modulation Schemes Thus Far

- Plot P_b versus E_b/N_0 for all schemes discussed thus far



BEP comparison over all schemes thus far

- We see that the BEP of PAM degrades rapidly as M increases, but in return we get an increasing bandwidth efficiency

- The non-coherent schemes may not in general perform as well as coherent schemes, but the implementation is easier, and often is the only option if the carrier phase cannot be reliably tracked.

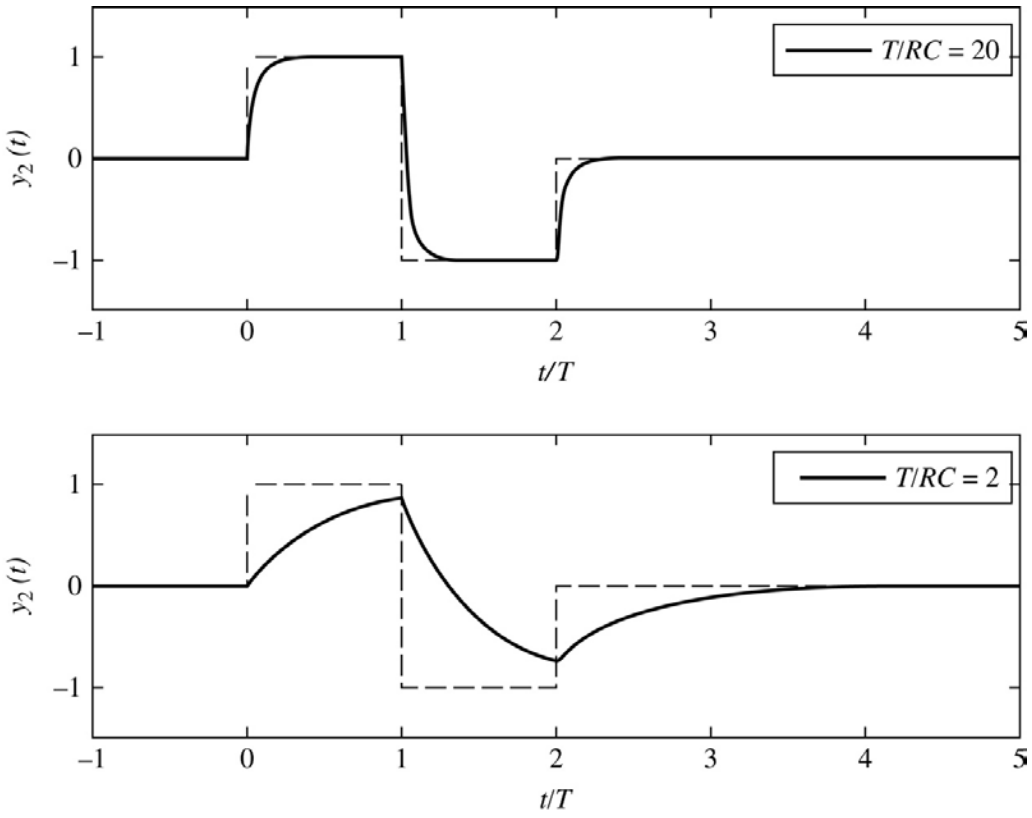
5.8 Performance of Zero-ISI Digital Data Systems

This section of Z&T picks up from Section 5.3 and 5.4 of Chapter 5. Here the term *intersymbol interference* (ISI) is defined as the signal distortion that results when a digital modulation signal is passed through a channel with limited bandwidth, relative to that needed to support the pulse shape being used. This material is not covered in Chapter 9 so we jump back to Chapter 5 to pick it up.

5.8.1 Effects of Filtering of Digital Data Waveforms

- Thus far we have made heavy use of the REC pulse shape
- From the study of the random binary pulse train in notes Chapter 4, we know that smoother pulse shapes offer a more compact PSD than the REC pulse (i.e., $T\,\mathrm{sinc}^2(fT)$)
- There is a class of pulse shapes which offer a very compact spectra, and hence are less sensitive to ISI resulting from a bandwidth limiting channel
- At baseband if a waveform employing REC pulses encounters say an RC lowpass filter, ISI results

- Consider the following:



ISI resulting when RC lowpass filtering REC pulses

- *Equalization*: At the receiver we attempt to undo the ISI caused by the channel, is one corrective measure to consider

5.8.2 Nyquist Pulse Shaping for Zero ISI

- We desire a pulse shape that has a zero ISI property and has a compact spectra
- Suppose we were sending impulses, $a_n\delta(t-nT)$, once per bit time $T = 1/R_b$, through a channel having an ideal lowpass response of bandwidth $W = R_b/2$, where a_n is the nth data bit being sent
- The sampling theorem says that the channel output can be perfectly reconstructed from samples taken at rate $R_b = 1/T$
- Recall that the impulse response of this channel is

$$h(t) = \text{sinc}(2Wt)$$

- The information bearing signal arriving at the receiver, less noise, is

$$y(t) = \sum_n a_n \text{sinc}\left[2W\left(t - \frac{n}{2W}\right)\right]$$

- Observe that the channel output when sampled at $t_m = m/(2W)$ has value a_m, because

$$\text{sinc}(m-n) = \begin{cases} 1, & m = n \\ 0, & m \neq n \end{cases}$$

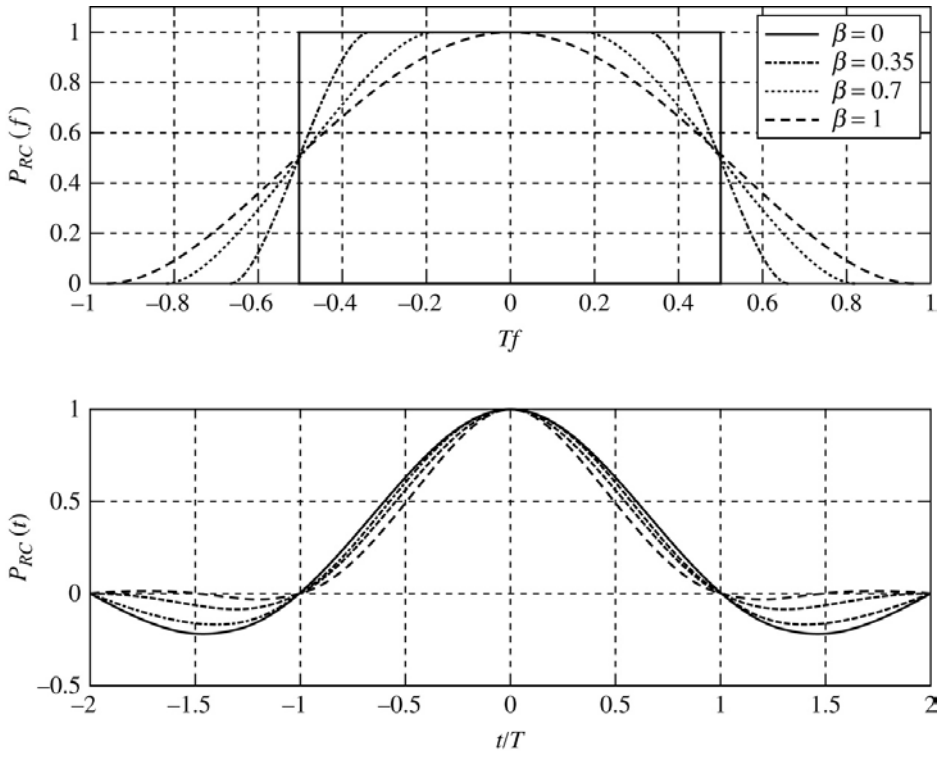
- Since the sinc( ) function has infinite extent it is not very practical, but the family of pulses having a *raised cosine* spectra also has zeros spaced every Ts

$$p_{\text{RC}}(t) = \frac{\cos(\pi\beta t/T)}{1-(2\beta t/T)^2}\text{sinc}\left(\frac{t}{T}\right)$$

- The corresponding spectra is

$$P_{\mathrm{RC}}(f) = \begin{cases} T, & |f| \le \frac{1-\beta}{2T} \\ \frac{T}{2}\left\{1 + \cos\left[\frac{\pi T}{\beta}\left(|f| - \frac{1-\beta}{2T}\right)\right]\right\}, & \frac{1-\beta}{2T} < |f| \le \frac{1+\beta}{2T} \\ 0, & |f| > \frac{1+\beta}{2T} \end{cases}$$

where β (frequently α) is known as the *roll-off factor* or *excess bandwidth factor*



RC spectra pulse and spectra

5.8.3 Nyquist's Pulse Shaping Criterion

- Formally, Nyquist's pulse shaping criterion states that

$$\sum_{k=-\infty}^{\infty} P\left(f + \frac{k}{T}\right) = T, \; |f| \le \frac{1}{2T}$$

gives rise to a pulse shape $p(t)$ such that

$$p(nT) = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases}$$

- The zero ISI condition thus results in the signal

$$y(t) = \sum_{n=-\infty}^{\infty} a_n p(t - nT)$$

- For proof see Z&T page 225

Example 5.8: RC Spectra Pulses in MATLAB

- A truncated version of the raised-cosine spectra filter can be utilized to create a realizable digital filter, along with impulse train modulation discussed earlier
- The MATLAB signal processing toolbox has both FIR and IIR filter types predefined, but a custom function is available for both MATLAB and Python
- A custom function for doing this is listed below:

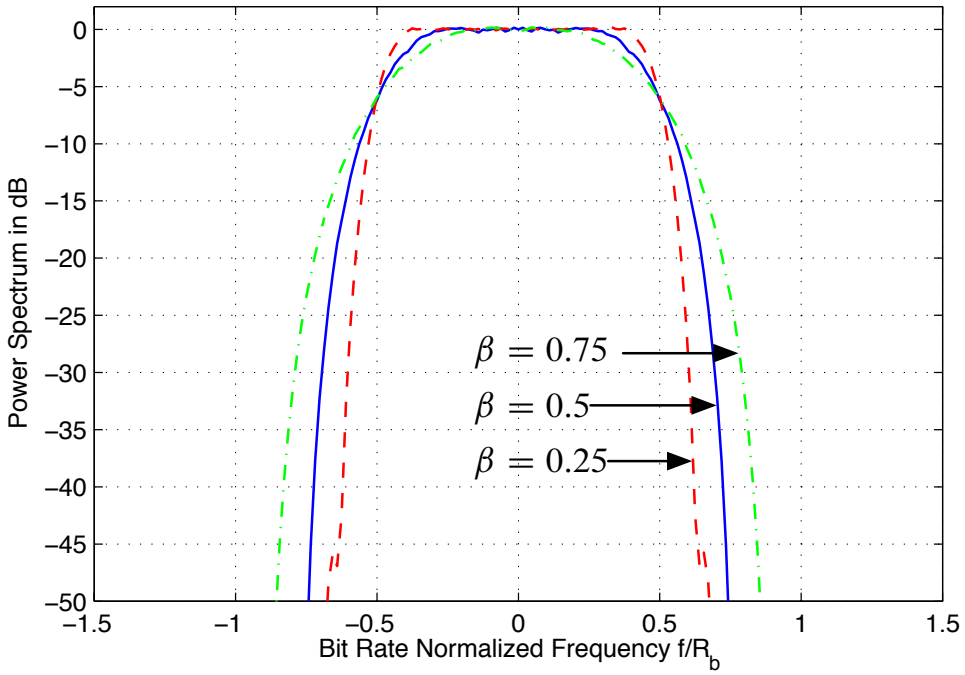
```
%//////////////////////////////////////////////////////////////////////
function [x,data,b] = shaped_psk(Nbit, Ns, beta, shape)
%        [x,data,b] = shaped_psk(Nbit, Ns, beta, shape)
%
% Complex baseband binary PSK modulator
%//////////////////////////////////////////////////////////////////////
%   Nbit = number of symbols processed
%   Ns = the number of samples per bit
%   beta = RC shape parameter; 0 --> sinc pulses
%  shape = 'REC', 'RC', SRC'
```

```
%
%      x = Complex baseband transmit signal vector
%   data = Binary 0,1 data being sent; used for error detecting
%  x_ref = Complex baseband reference signal used for coherent
%          demodulation of FSK (does not contain data)
%//////////////////////////////////////////////////////////////////////
% Mark Wickert October 2010

A = 1;
data = randi(0:1,1,Nbit); % 0,1 random integers
y = [2*data-1; zeros(Ns-1,Nbit)]; %impulse train modulate +/-1
y = reshape(y,1,Nbit*Ns);

switch lower(shape)
    case 'rec'
        b = ones(1,Ns);
    case 'rc'
        b = rc_imp(Ns,beta,1,1,6);
    case 'src'
        b = sqrt_rc_imp(Ns,beta,6);
otherwise
        error('Unknown shape type')
end
y = filter(b,1,y); % filter impulse train data with selected pulse shape

x = A*y;

%//////////////////////////////////////////////////////////////////////
function b = rc_imp(Ns,alpha,T,Tm,M)
%    b = rc_imp(Ns,alpha,T,Tm,M)
%
%Raised-Cosine Impulse Response Filter
%//////////////////////////////////////////////////////////////////////
%   Ns = number of samples per symbol
%alpha = excess bandwidth factor = 0.25 in EDGE  recvr meas. filter
%    T = symbol period
%   Tm = measurement equivalent symbol period, note Ns <-> T
%    M = equals RC one-sided symbol truncation factor
%
% Tm = 1/180 kHz in EDGE receiver measuring filter
%  T = 1/270.83333 kHz in GSM and EDGE 8-psk
%//////////////////////////////////////////////////////////////////////
% Mark Wickert

% Design the filter
n = -M*Ns:M*Ns;
```

```
b = zeros(size(n));
a = alpha;
Ns = Ns*Tm/T; % rescale Ns to reflect Tm possibly different from T
for i=1:length(n),
   if (1 - 4*(a*n(i)/Ns)^2) == 0
      b(i) = pi/4*sinc(1/(2*a));
   else
      b(i) = sinc(n(i)/Ns)*cos(pi*a*n(i)/Ns)/(1 - 4*(a*n(i)/Ns)^2);
   end
end
```

- First consider the PSD in dB for $\beta = 0.75$, 0.5, and 0.25

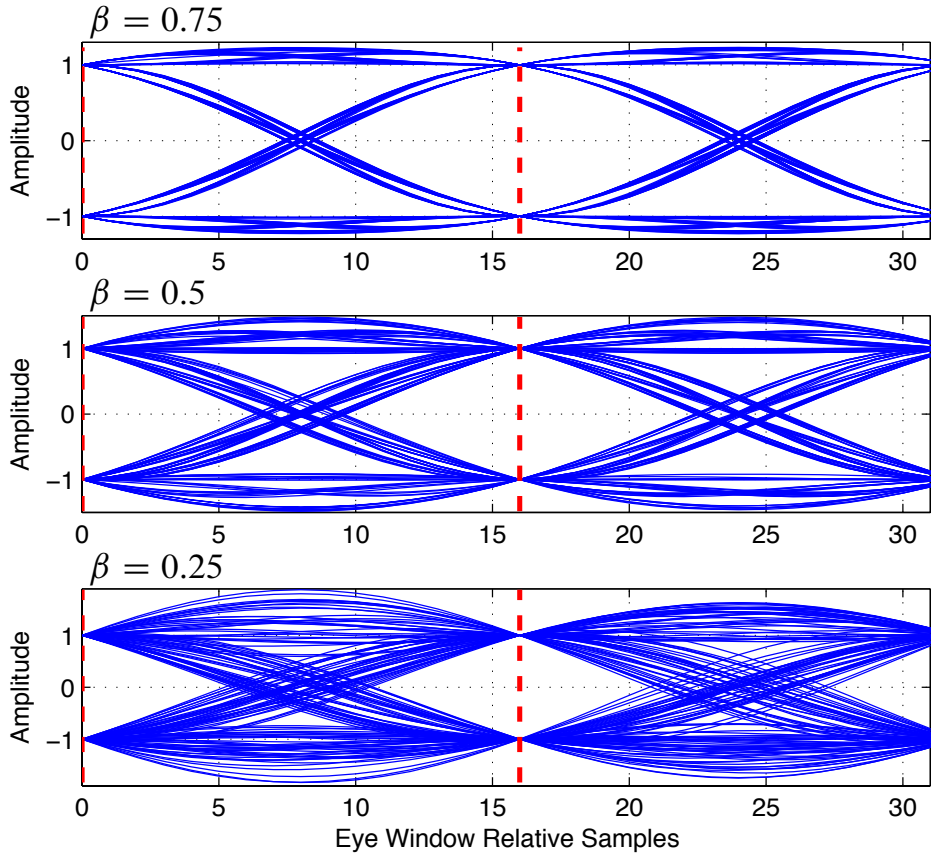
```
>> [x,data,b] = shaped_psk(100000, 16, 0.5, 'RC');
>> spec_plot(x,2^10,16);
>> [x,data,b] = shaped_psk(100000, 16, 0.25, 'RC');
>> spec_plot(x,2^10,16,'r');
>> [x,data,b] = shaped_psk(100000, 16, 0.75, 'RC');
>> spec_plot(x,2^10,16,'g');
>> grid
>> axis([-1.5 1.5 -50 2])
```



Raised cosine spectra shaped baseband binary modulation PSD

- Next we consider the eye plots corresponding binary baseband modulation (this could be BPSK)

```
>> [x,data,b] = shaped_psk(10000, 16, 0.75, 'RC');
>> subplot(311);  eyeplot_mark(x(1:5000),2*16,96,16,1)
>> [x,data,b] = shaped_psk(10000, 16, 0.5, 'RC');
>> subplot(312);  eyeplot_mark(x(1:5000),2*16,96,16,1)
>> [x,data,b] = shaped_psk(10000, 16, 0.25, 'RC');
>> subplot(313);  eyeplot_mark(x(1:5000),2*16,96,16,1)
```

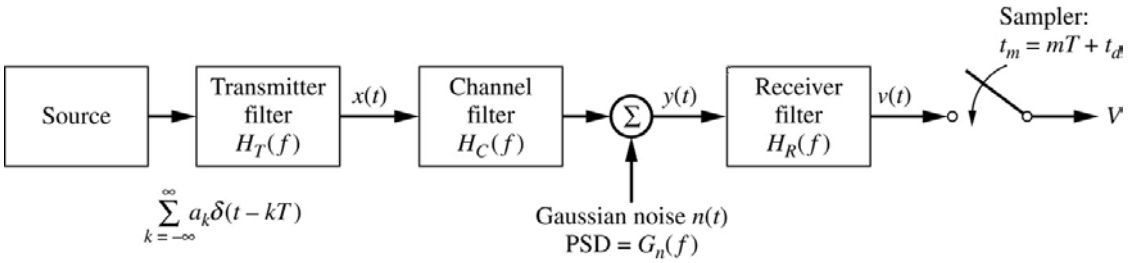


Raised cosine spectra shaped baseband binary modulation eye plots

- From the PSD smaller β produces a decreased signal bandwidth
- Notice also that as β gets smaller the eye width is reduced, thus increasing the timing sensitivity

5.8.4 Zero-ISI System Modeling with Noise

- We now consider the following system model:



Signal with additive colored noise through a bandlimited channel

- The transmitted signal is of the form

$$x(t) = \sum_{k=-\infty}^{\infty} a_k \delta(t - kT) * h_T(t) = \sum_{k=-\infty}^{\infty} a_k h_T(t - kT)$$

- The received signal, prior to the detection filter $h_R(t)$ (like a matched filter), is

$$y(t) = x(t) * h_C(t) + n(t)$$

- We require that the sample of $v(t)$ taken at time $t = t_d$ (t_d due to delays imparted by the filters) be of the form

$$V = Aa_0 p(0) + N = Aa_0 + N$$

where

$$Ap(t - t_d) = h_T(t) * h_C(t) * h_R(t)$$

- In the frequency domain this is equivalent to

$$AP(f)e^{-j2\pi f t_d} = H_T(f)H_C(f)H_R(f)$$

where $P(f) = \mathcal{F}\{p(t)\}$

- The RV N is Gaussian and is the result of filtering $n(t)$ with $H_R(f)$

$$N = n(t) * h_R(t)|_{t=t_d}$$

- For now we assume that the a_n are binary ± 1 data bits equally likely

- This results in P_E reducing to

$$P_E = P[N \geq A] = \int_A^\infty \frac{\exp(-u^2/(2\sigma N^2)}{\sqrt{2\pi\sigma_N^2}}\, du = Q\left(\frac{A}{\sigma_N}\right)$$

where

$$\sigma_N^2 = \int_{-\infty}^{\infty} G_n(f)|H_R(f)|^2\, df$$

- The end game here is to minimize P_E by choosing $H_T(f)$ and $H_R(f)$ such that A/σ_N is maximized and the zero ISI constraint is maintained through the channel

- As in the matched filter receiver analysis, we again make use of the Schwarz inequality to obtain

$$|H_R(f)|_{\text{opt}} = \frac{KP^{1/2}(f)}{G_n^{1/4}(f)|H_C(f)|^{1/2}}$$
$$|H_T(f)|_{\text{opt}} = \frac{AP^{1/2}(f)G_n^{1/4}(f)}{K|H_C(f)|^{1/2}},$$

where K is an arbitrary constant and by construction, $P(f)$ (the cascade of all three filters) has the zero ISI property

- The minimum value for P_E becomes

$$P_{b,\min} = Q\left[\sqrt{E_T}\left(\int_{-\infty}^{\infty} \frac{G_n^{1/2}(f)P(f)}{|H_C(f)|}\,df\right)^{-1}\right]$$

where

$$E_T = E\{a_m^2\}\int_{-\infty}^{\infty}|h_T(t)|^2\,dt = \int_{-\infty}^{\infty}|H_T(f)|^2\,df$$

since $E\{a_m^2\} = 1$

- proof: See Z&T p. 440

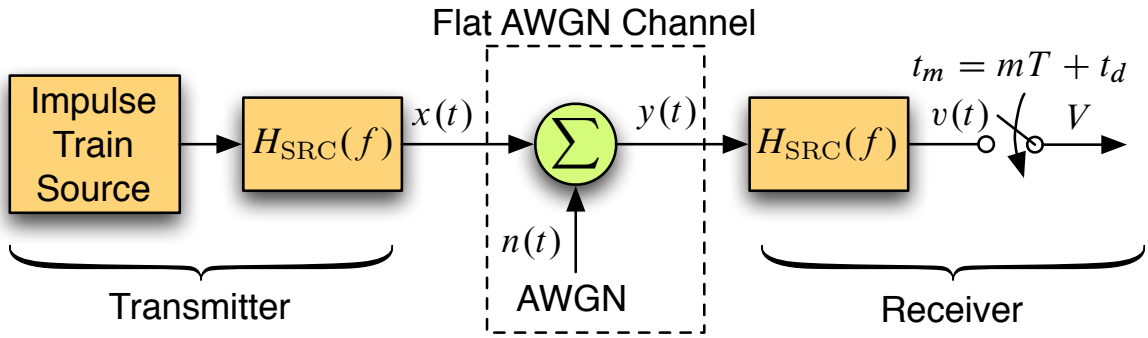
Special Case

- An important special case occurs when the noise is white, i.e.,

$$G_n(f) = \frac{N_0}{2}, \forall f,$$

and the channel is flat or ideal over the signal spectrum, that is

$$H_C(f) = 1,\ |f| \le \frac{1}{T}$$



Baseband SRC transceiver

- We now plug into the $|H_R(f)|_{\text{opt}}$ and $|H_T(f)|_{\text{opt}}$ formulas to obtain

$$|H_R(f)|_{\text{opt}} = |H_T(f)|_{\text{opt}} = K' P^{1/2}(f),$$

  where K' is an arbitrary constant

- Under these conditions, and with $P(f)$ a raised cosine spectrum, the transmit and receive filters become what is known as *square-root raised-cosine filters* (SRC) or (RRC)[4]

$$p_{\text{SRC}}(t) = \begin{cases} 1 - \beta + 4\frac{\beta}{\pi}, & t = 0 \\ \frac{\beta}{\sqrt{2}}\left[\left(1 + \frac{2}{\pi}\right)\sin\left(\frac{\pi}{4\beta}\right) + \left(1 - \frac{2}{\pi}\right)\cos\left(\frac{\pi}{4\beta}\right)\right], & t = \pm\frac{T}{4\beta} \\ \left\{\sin\left[\pi\frac{t}{T}(1-\beta)\right] + 4\beta\frac{t}{T}\cos\left[\pi\frac{t}{T}(1+\beta)\right]\right\} \cdot \left\{\pi\frac{t}{T}\left[1 - \left(4\beta\frac{t}{T}\right)^2\right]\right\}^{-1}, & \text{otherwise} \end{cases}$$

- MATLAB has filter design functions to support this filter type in the discrete-time domain

- The P_E expression now reduces to

$$P_E = Q\left\{\sqrt{E_T}\left[\frac{N_0}{2}\int_{-1/T}^{1/T} P(f)\,df\right]^{-1}\right\} = Q\left(\sqrt{\frac{2E_T}{N_0}}\right)$$

[4] http://en.wikipedia.org/wiki/Root-raised-cosine_filter

where

$$p(0) = \int_{-1/T}^{1/T} P(f)\, df = 1$$

- This is the familiar binary antipodal result assuming an infinite bandwidth baseband channel

Example 5.9: Baseband Binary Tx/Rx with SRC in MATLAB

- Using the result just developed, we consider the simulation of a SRC pulse-shaped antipodal baseband transceiver
  - Note: This could just as well be a complex baseband simulation of SRC shaped BPSK
- With the function `shaped_psk()`, `shape` may take the value `'src'`
- The function returns the transmit pulse shape filter in the vector `b`
- An SRC pulse shape can be obtained using the custom function `sqrt_rc_imp()` (also available in Python `digitalcom.py`)

```
function b = sqrt_rc_imp(Ns,beta,M)
%        b = sqrt_rc_imp(Ns,beta,M)
%
%   Sqrt-Raised-Cosine Impulse Response Filter
%////////////////////////////////////////////////////////////////////
%  Ns = number of samples per symbol
% beta = excess bandwidth factor ( = 0.35 for IS 136)
%     M = equals sqrt(RC) one-sided symbol truncation factor
%////////////////////////////////////////////////////////////////////
% Mark Wickert
```

```
% Design the filter
n = -M*Ns:M*Ns;
b = zeros(size(n));
a = beta;
for i=1:length(n),
   if (1 - 16*a^2*(n(i)/Ns)^2) == 0
      b(i) = 1/2*((1+a)*sin((1+a)*pi/(4*a))-(1-a)*...
             cos((1-a)*pi/(4*a))+ (4*a)/pi*sin((1-a)*pi/(4*a)));
   else
      b(i) = 4*a./(pi*(1 - 16*a^2*(n(i)/Ns).^2));
      b(i) = b(i).*(cos((1+a)*pi*n(i)/Ns) + ...
             sinc((1-a)*n(i)/Ns)*(1-a)*pi/(4*a));
   end
end
```

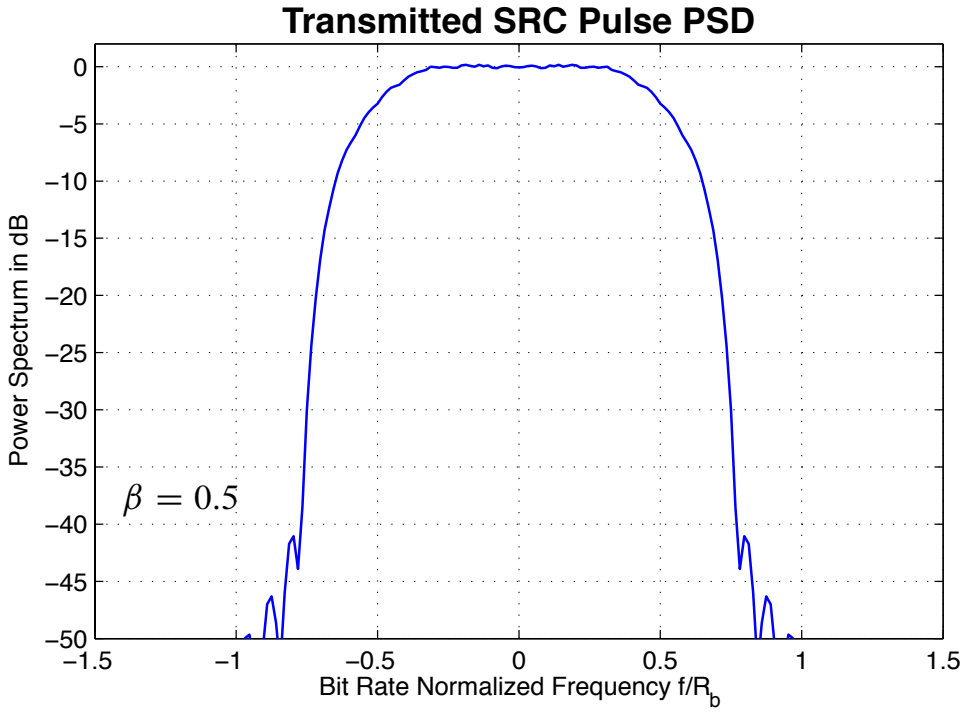
- To generate shaped psk consider `shaped_psk_sim()`

```
function [Pe,errors,N_RecBits,z] = shaped_psk_sim(SNR,Nbits,timing,beta,shape)
% [Pe,errors,N_RecBits,z] = shaped_psk_sim(SNR,m,theta,Nbits,timing)
%
% Coherent Binary BPSK modem end-to-end simulation function
%//////////////////////////////////////////////////////////////////////
%      SNR = Set the received signal SNR in dB; for on-off keying the SNR
%            is defined interms of the average signal power, e.g., 1/2 the
%            peak symbol energy
%    Nbits = The number of bits simulated
%   timing = Sets the sampler timing to give the minimum BEP; find value
%            using eyeplot_mark function
% rec_type = Receiver type: 'coherent', bpf_env_det', or 'lim_discrim'
%
%       Pe = Estimated bit error probability (BEP)
%   errors = Number of bit errors found (good to obtain at least 100
%            errors over an AWGN channel
%  RecBits = The number of bits processed y bit_errors function; If you
%            create a 'top-level' function 'errors' and 'RecBits' can be
%            used to combine results from multiple runs
%        z = Receiver decision statistic; use as input to eyeplot_mark for
%            further analysis, e.g., find a good value for 'timing'
%//////////////////////////////////////////////////////////////////////

% Mark Wickert October 2010

Ns = 16; % Fix the number of samples per bit to 16
```

```
% Generate Shaped PSK tx signal
[x,data,b] = shaped_psk(Nbits, 16, beta, shape);
y = cpx_AWGN(x,SNR,16);% AWGN channel

% Enter receiver

% Coherent receiver (at complex baseband)
%//////////////////////////////
% Matched filter is designed by the Tx function as vector b
% Process complex BB signal through filter to form decision statistic
z = filter(b,1,y)/Ns;
z_det = real(z);
% Decision logic
d_hat = (sign(z_det(timing:16:end))+1)/2; % 0,1 logic levels
% Error detect
[Pe,errors,N_RecBits] = bit_errors(data,d_hat);
```

- We now set a breakpoint inside the function so we can observe the transmit SRC PSD

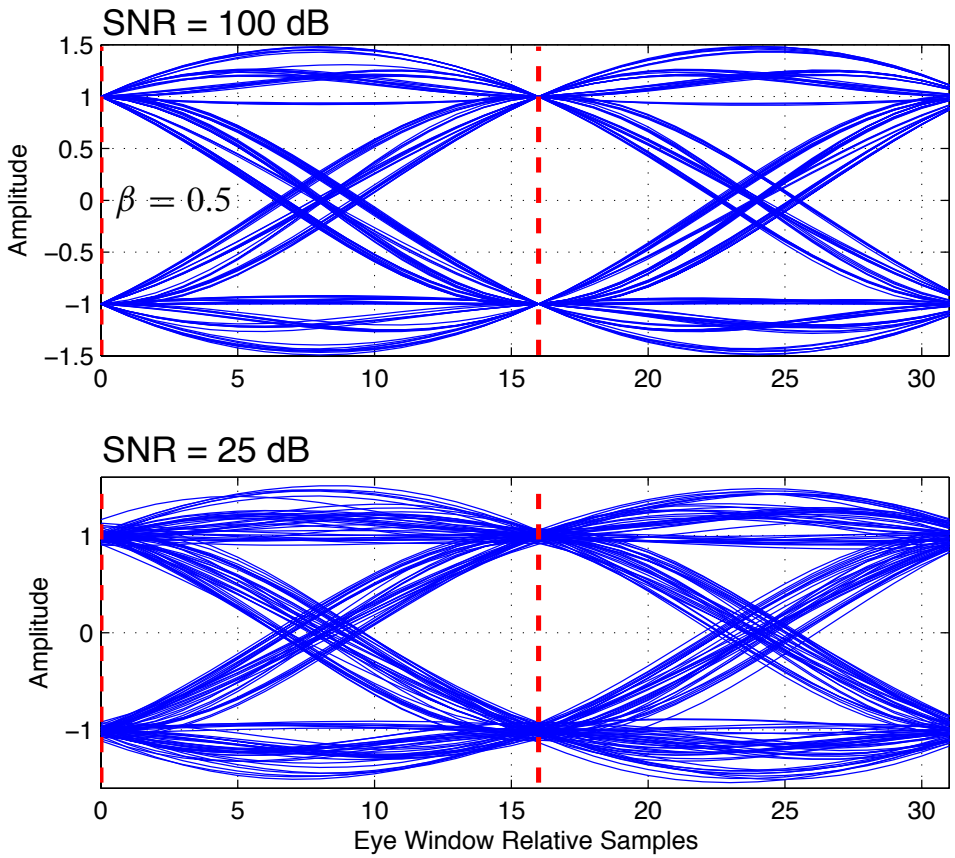
```
>> [Pe,errors,N_RecBits,z] = shaped_psk_sim(100,100000,1,0.5,'src');
39  z_det = real(z);
K>> spec_plot(x,2^10,16);
```



SRC shaped baseband binary modulation PSD

- Next we observe the eye plot at high (100 dB) SNR and at 25 dB

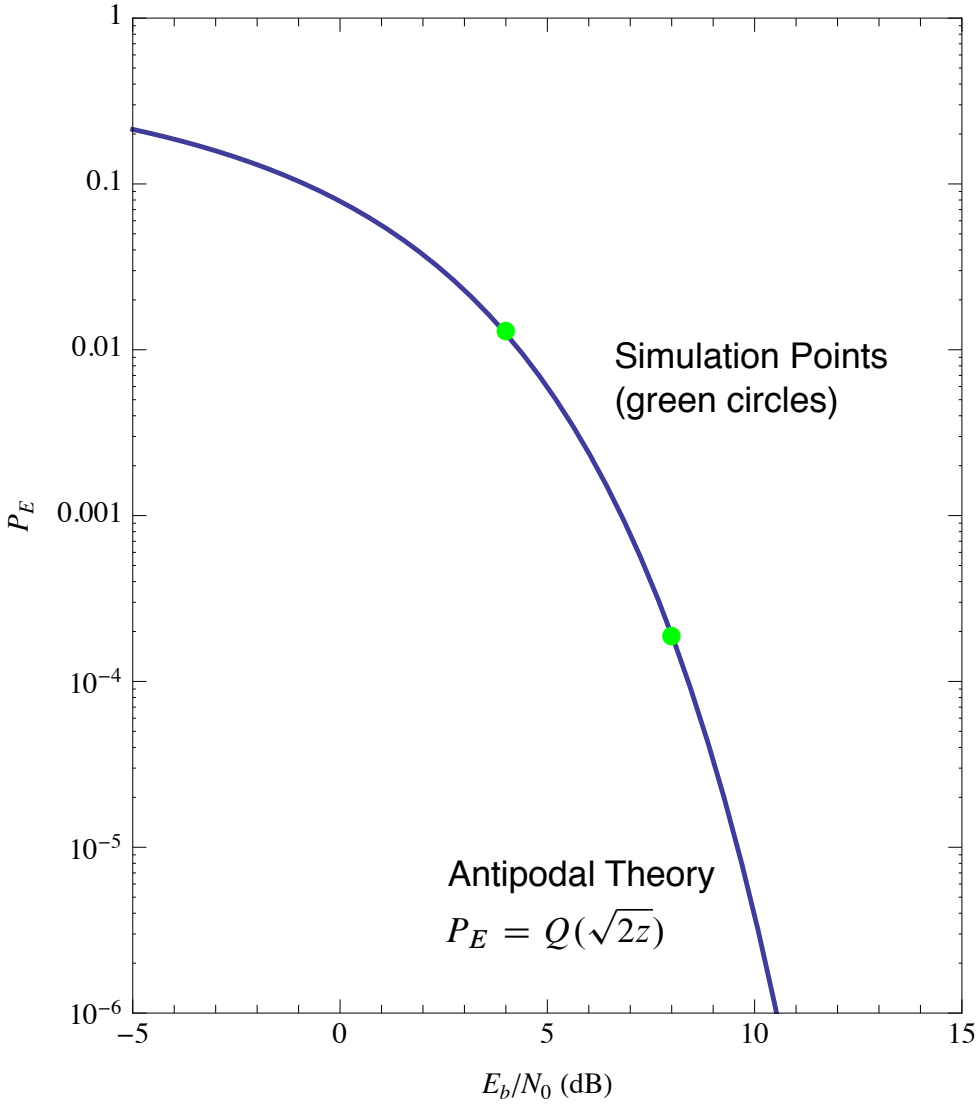
```
>> [Pe,errors,N_RecBits,z] = shaped_psk_sim(100,10000,1,0.5,'src');
////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 9988
       BEP: 0.00e+00
////////////////////////////////////////////////////////
>> subplot(211)
>> eyeplot_mark(real(z(1:5000)),2*16,192,16,1);
>> [Pe,errors,N_RecBits,z] = shaped_psk_sim(25,10000,1,0.5,'src');
////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 9988
       BEP: 0.00e+00
////////////////////////////////////////////////////////
>> subplot(212)
>> eyeplot_mark(real(z(1:5000)),2*16,192,16,1);
```



Receiver eye plots at SNR = 100 dB and 25 dB

- Finally, we check a BEP points and see if they agree with the antipodal theory

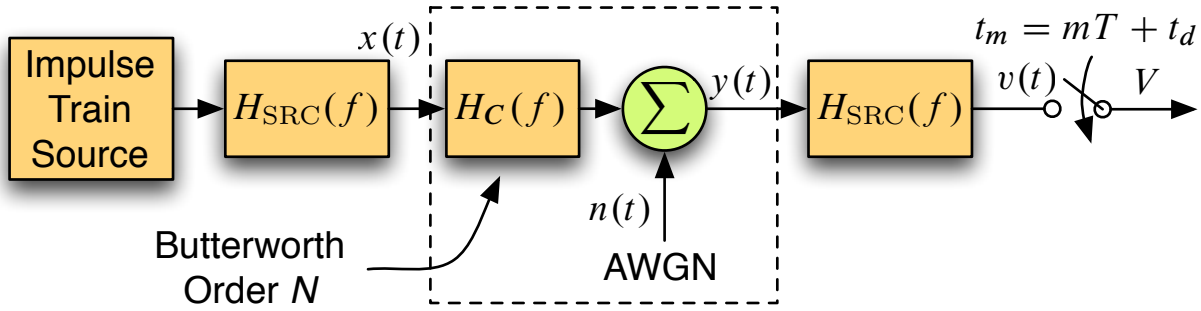
```
>> [Pe,errors,N_RecBits,z] = shaped_psk_sim(8,1000000,1,0.5,'src');
//////////////////////////////////////////////////////
Bit Errors: 182
Bits Total: 999988
       BEP: 1.82e-04
//////////////////////////////////////////////////////
>> [Pe,errors,N_RecBits,z] = shaped_psk_sim(4,100000,1,0.5,'src');
...
```



Measured and theoretical BEP

Example 5.10: Degradation Due to Channel Bandlimiting

- Suppose that we have AWGN, but the channel has an unspecified frequency response
  - This could be due to *mild* bandlimiting due to a first-order lowpass or similar



System block diagram

- We are interested in the resulting degradation factor in E_T/N_0 relative to the infinite-bandwidth AWGN channel, with the pulse shaping unchanged

- We have for P_E

$$
\begin{aligned}
&Q\left(\sqrt{E_T}\left(\int_{-\infty}^{\infty}\frac{G_n^{1/2}(f)P(f)}{|H_C(f)|}\,df\right)^{-1}\right)\\
&\qquad=Q\left(\sqrt{E_T}\left(\int_{-\infty}^{\infty}\frac{\sqrt{N_0/2}P(f)}{|H_C(f)|}\,df\right)^{-1}\right)\\
&\qquad=Q\left(\sqrt{\frac{2E_T}{N_0}}\left(\int_{-\infty}^{\infty}\frac{P(f)}{|H_C(f)|}\,df\right)^{-1}\right)\\
&\qquad=Q\left(2\left(\int_{-\infty}^{\infty}\frac{P(f)}{|H_C(f)|}\,df\right)^{-2}\sqrt{\frac{E_T}{N_0}}\right)\\
&\qquad=\sqrt{\frac{2}{F}\cdot\frac{E_T}{N_0}},
\end{aligned}
$$

  where F is the integral

$$
F=\left(\int_{-\infty}^{\infty}\frac{P(f)}{|H_C(f)|}\,df\right)^2=\left(2\int_{0}^{\infty}\frac{P(f)}{|H_C(f)|}\,df\right)^2
$$

- The degradation in dB is

$$
F_{\text{dB}}=20\log_{10}\left(2\int_{0}^{\infty}\frac{P(f)}{|H_C(f)|}\,df\right)
$$

- As a case in point consider Nth-order Butterworth lowpass having magnitude response

$$
|H_C(f)|=\sqrt{\frac{1}{1+(f/f_3)^{2N}}}
$$

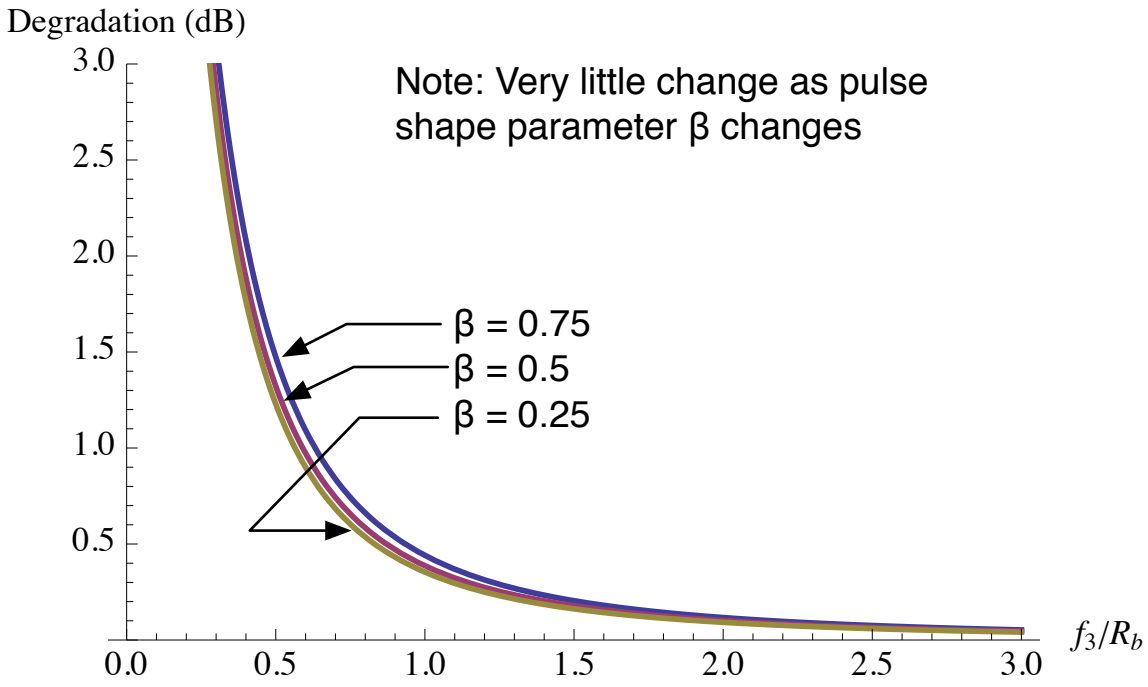
- The integration is performed in Mathematica

```
P[f_, β_] := Piecewise[
  {{1, Abs[f] ≤ (1 - β)/2}, {1/2 (1 + Cos[π/β (Abs[f] - (1 - β)/2)]), Abs[f] > (1 - β)/2 && Abs[f] ≤ (1 + β)/2}}];

H_RC[f_, f3_, N_] := Sqrt[1/(1 + (f / f3)^(2 N))];

FdB[f3_, β_, N_] := 20 Log[10, 2 NIntegrate[P[f, β]/H_RC[f, f3, N], {f, 0, Infinity}]];
```

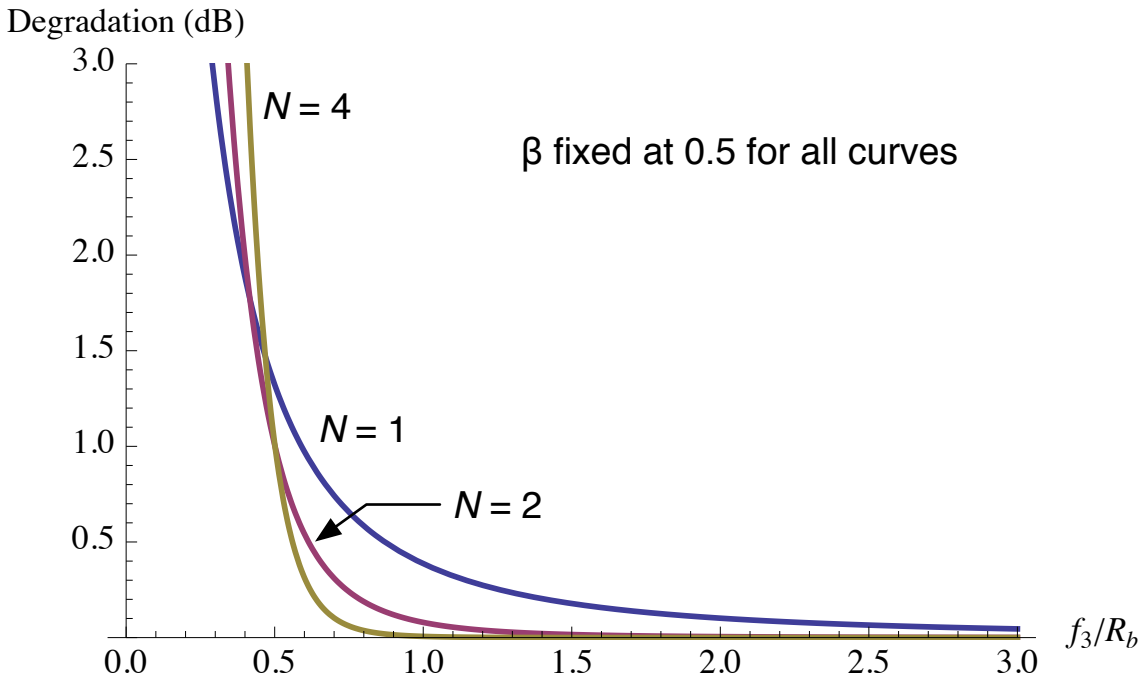
Calculation set-up in Mathematica

- First we consider degradation with $N = 1$ (RC lowpass) and $\beta = 0.75, 0.5, 0.25$, versus f_3/R_b

- Seeing that the sensitivity to β is rather small, we next consider filter orders $N = 1, 2$, and 4 and $\beta = 0.5$, versus f_3/R_b



Degradation in dB with fixed pulse shaping and changing $H_C(f)$



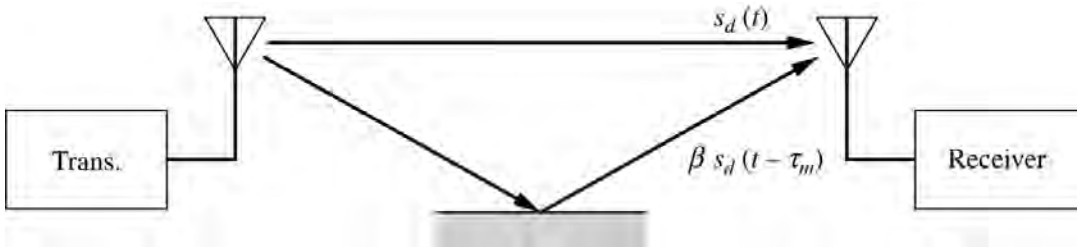
Degradation in dB with fixed pulse shaping and changing $H_C(f)$

- As interesting exercise would be to verify the above results via Monte-Carlo simulation

---

5.9 Multipath Interference

- The AWGN channel is convenient and simple, but it does not always fit the real-world situation
- A simple *two-ray* multipath model is the following



Two-ray multipath model

- If we assume that the received signal is perturbed by AWGN of PSD $N_0/2$ we have a received signal of the form

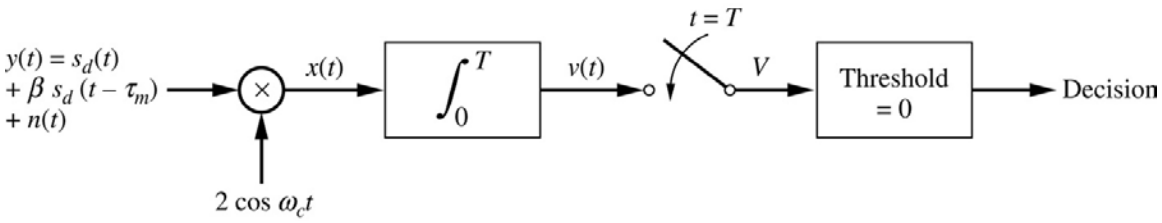
$$y(t) = s_d(t) + \beta s_d(t - \tau_m) + n(t)$$

- Consider the impact on BPSK which has

$$s_d(t) = Ad(t)\cos(\omega_c t),$$

where $d(t)$ is the data stream, say ± 1 rectangular pulses of duration T

- A coherent receiver will first multiply the received signal by $\cos(\omega_c t)$, lowpass filter to remove the double frequency terms, and then matched filter:



Hybrid bandpass/baseband receiver model

- In terms of complex baseband modeling, the received signal is of the form

$$\begin{aligned}\tilde{x}(t) &= \tilde{s}_d(t) + \tilde{s}_d(t - \tau_m) + \tilde{n}(t) \\ &= Ad(t) + \beta Ad(t - \tau_m)e^{-j\omega_c \tau_m} + \left[n_I(t) + jn_Q(t)\right]\end{aligned}$$

- In the coherent receiver the matched filter only operates on the in-phase component

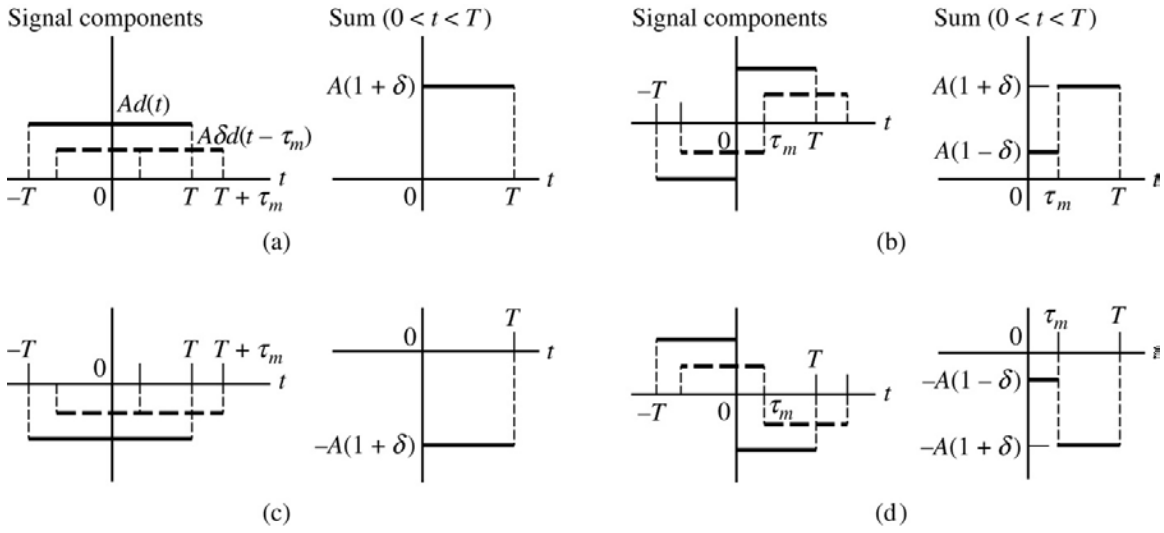
$$x_I(t) \overset{\text{text}}{=} Ad(t) + \beta Ad(t - \tau_m)\cos(\omega_c \tau_m) + \underbrace{n_I(t)}_{n_c(t)}$$

- Analysis of the most general case is difficult, but there are two special cases worth considering

  1. **Slow & fast fading small delay**: Assume $\tau_m/T \simeq 0$ and hence that $d(t-\tau_m) \simeq d(t)$; We also assume that the product $\omega_c\tau_m$ is treated as an RV uniform on $[-\pi,\pi)$; Furthermore we assume that there are actually many multipath components of random amplitudes and phases such that the I and Q components are Gaussian, the envelope is Ricean or Rayleigh and the phase is uniform
  2. **Static fading**: Assume that $0 < \tau_m/T \leq 1$ which results in successive bits of $d(t)$ and $d(t-\tau_m)$ overlapping, that is there is ISI present; We assume that $\delta = \beta\cos(\omega_c\tau_m)$ is a constant parameter

- **Case 2 Analysis**: The analysis here reduces down to considering four two-bit ISI patterns

$$x_I(t) = x(t) = Ad(t) + A\delta d(t-\tau_m) + n_I(t)$$



Two-bit ISI patterns in BPSK static fading analysis

- The average BEP is

$$P_E = \frac{1}{4}\big[P(E|++) + P(E|-+) + P(E|+-) + P(E|--)\big]$$

- Noise component at the integrator (correlator matched filter) output is just

$$N = \int_0^T 2n(t)\cos(\omega_c t)\,dt$$

  Note: The 2 has been added in the block diagram for convenience

- The noise is Gaussian with variance

$$\begin{aligned}\sigma_n^2 &= E\left[4\int_0^T\int_0^T n(t)n(\lambda)\cos(\omega_c t)\cos(\omega_c\lambda)\,dt\,d\lambda\right]\\ &= 4\int_0^T\int_0^T \frac{N_0}{2}\delta(t-\lambda)\cos(\omega_c t)\cos(\omega_c\lambda)\,dt\,d\lambda\\ &= N_0\int_0^T \big[1+\cos(2\omega_c t)\big]\,dt\\ &= N_0 T, \quad \text{assuming } \omega_c T \text{ is a multiple of } 2\pi\end{aligned}$$

- The two-bit patterns are symmetrical so we only need to consider two conditional error probabilities

- Under a $(1,1)$ transmission the integrator output signal component is

$$V_{++} = AT(1+\delta)$$

and the conditional error probability is

$$P(E|++) = P[AT(1+\delta) + N < 0]$$
$$= Q\left(\sqrt{\frac{2E}{N_0}}(1+\delta)\right)$$

- Under a $(-1, 1)$ transmission the integrator output signal component is

$$V_{-11} = A(T-\tau_m)(1+\delta) + A(\tau_m)(1-\delta)$$
$$= AT\left[(1+\delta) - \frac{2\delta\tau_m}{T}\right]$$

and the conditional error probability is

$$P(E|-+) = P\left[AT\left\{(1+\delta) - \frac{2\delta\tau_m}{T}\right\} + N < 0\right]$$
$$= Q\left(\sqrt{\frac{2E}{N_0}}\left[(1+\delta) - \frac{2\delta\tau_m}{T}\right]\right)$$

- Finally, the average BEP is

$$P_E = \frac{1}{2}Q\left(\sqrt{2z_0}(1+\delta)\right) + \frac{1}{2}Q\left(\sqrt{2z_0}\left[(1+\delta) - \frac{2\delta\tau_m}{T}\right]\right)$$

where $z_0 \triangleq E_b/N_0 = A^2T/(2N_0)$ as in the past

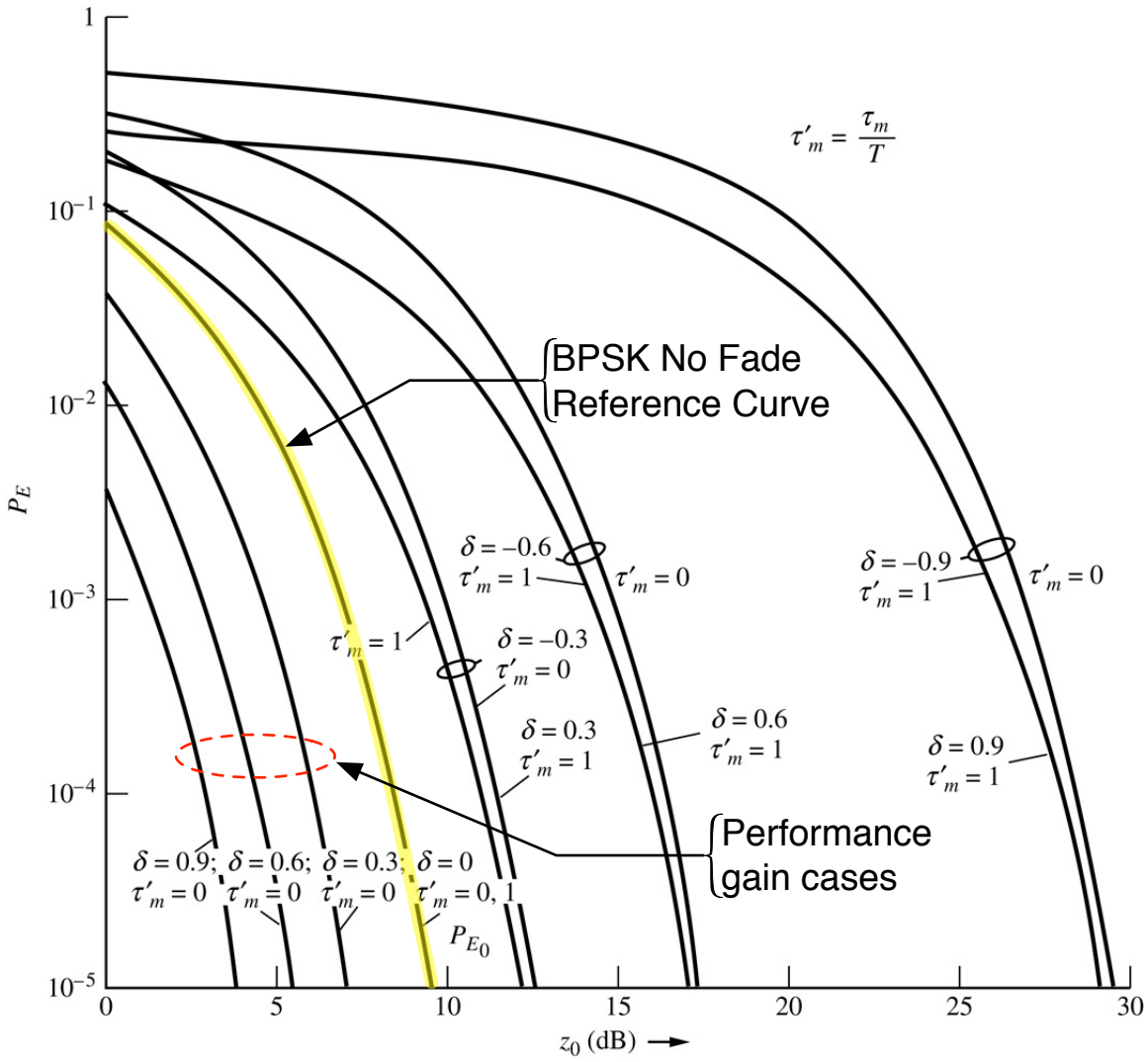
- An issue to consider is that the total *effective* received energy is actually

$$z_m = z_0(1+\delta)^2$$

meaning that with multipath it is possible under some circumstances to have slightly improved performance

  - Consult Z&T pages 435–436 for more dicussion

- In the plots below we use z_0 as the reference



Static fading BEP curves

- The use of equalization can be used to counter a static multipath channel

5.10 Flat Fading Channels

- The fast & slow fading scenario of the previous section was explained under limiting conditions as consisting of many delayed multipath components with random amplitude and phases
- Two types of components exist in this situation
  1. A direct *line-of-sight* (LOS) component from the transmitter to the receiver, known as the *specular component* (not always present)
  2. A collection of non-LOS components, such that by application of the central limit theorem, makes their sum have Gaussian in-phase and quadrature components; this is the *diffuse component*
- The envelope of the composite specular plus diffuse components, signal thus has a Ricean pdf

$$f_R(r) = \frac{r}{\sigma^2} \exp\left(-\frac{(r^2 + A^2)}{2\sigma^2}\right) I_0\left(\frac{rA}{\sigma^2}\right), \quad r \geq 0$$

- Now we need to consider the rate at which the fade parameters vary
  1. If the envelope varies slowly relative to the bit duration, we have *slow fading*
  2. If the envelope and/or phase varies significantly over the bit duration, we have *fast fading*
- Fast fading is more difficult to analyze and will not be considered at present

- Under slow fading we would like to consider both a Ricean and Rayleigh envelope, but for now will just consider the Rayleigh, which means non-LOS communications

- Considering again the case of BPSK, we can write the received signal as

$$x(t) = Rd(t) + n_c(t)$$

where R is a Rayleigh RV (Ricean with $A = 0$)

- The conditional P_E is

$$P_E(R) = Q\left(\sqrt{2Z}\right)$$

where now Z is an RV by virtue of the Rayleigh fading, that is

$$Z = \frac{R^2 T}{2N_0}$$

- The average P_E can be obtained by weighting the conditional P_E with the Rayleigh pdf and integrating

$$\overline{P}_E = \int_0^\infty Q\left(\sqrt{2z}\right) \frac{1}{\overline{Z}} e^{-z/\overline{Z}}\, dz$$

where $\overline{Z}$ is the average SNR

- Integration by parts, $\int u\, dv = uv - \int v\, du$, (see Z& T p. 451) yields

$$\overline{P}_E = \frac{1}{2} - \frac{1}{2\sqrt{\pi}} \int_0^\infty \frac{\exp[-z(1 + 1/\overline{Z})]}{\sqrt{z}}\, dz$$

- The integral above can be reduced via a very convenient substitution, namely $w = \sqrt{z}$

$$\overline{P}_E = \frac{1}{2} - \frac{1}{\sqrt{\pi}} \int_0^{\infty} \exp\left[-w^2\left(1 + \frac{1}{\overline{Z}}\right)\right] dz$$

- Since $\overline{Z}$ is just a constant and the area of a zero mean Gaussian pdf on $[0, \infty]$ is 1/2, it follows with a variable substitution that

$$\overline{P}_E = \frac{1}{2}\left(1 - \sqrt{\frac{\overline{Z}}{1 + \overline{Z}}}\right), \quad \text{BPSK}$$

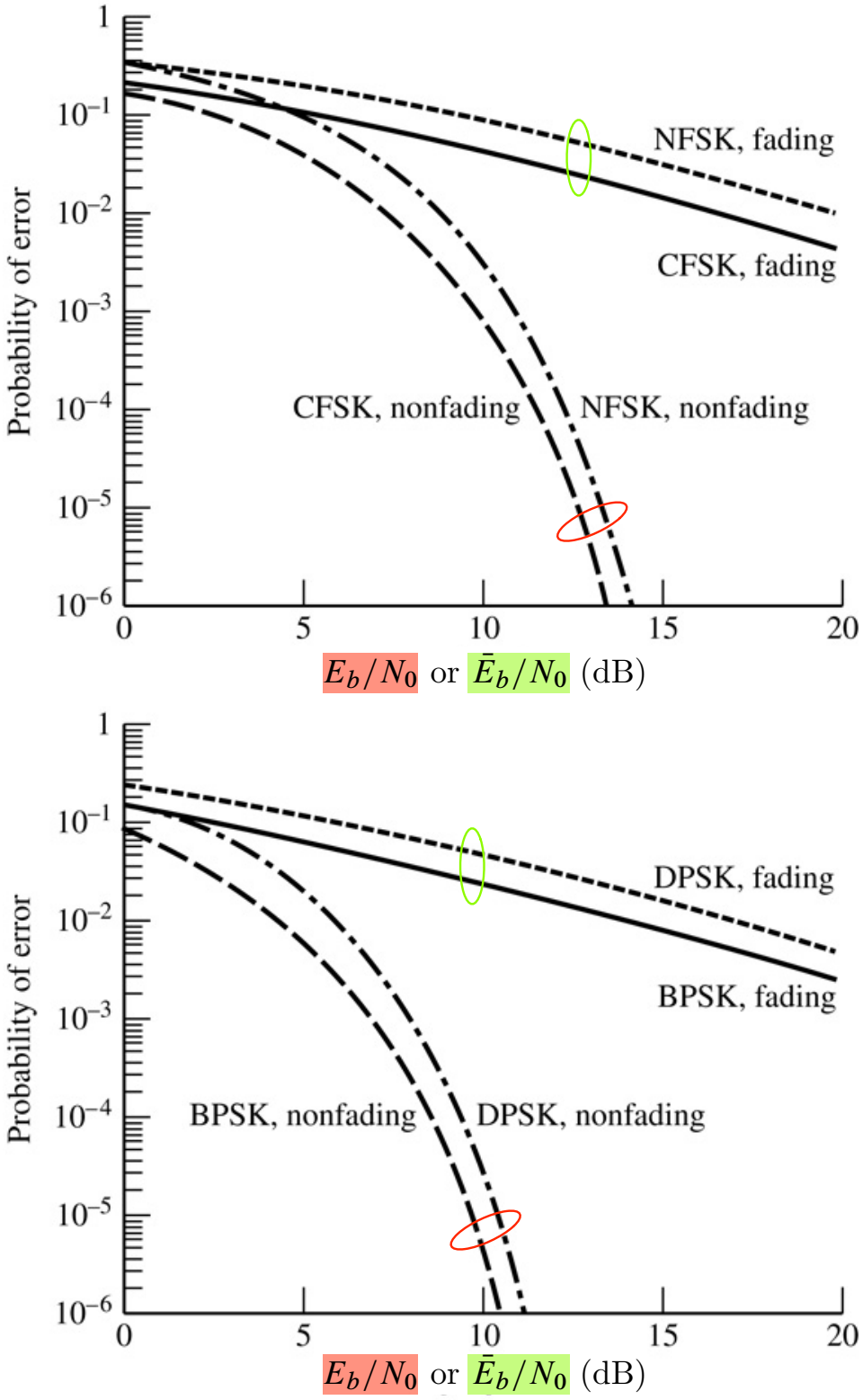
- Additional analysis can be performed for coherent FSK, DPSK, and noncoherent FSK, yielding

$$\overline{P}_E = \frac{1}{2}\left(1 - \sqrt{\frac{\overline{Z}}{2 + \overline{Z}}}\right), \quad \text{coherent FSK}$$

$$\overline{P}_E = \frac{1}{2(1 + \overline{Z})}, \quad \text{DPSK}$$

$$\overline{P}_E = \frac{1}{2 + \overline{Z}}, \quad \text{noncoherent FSK}$$

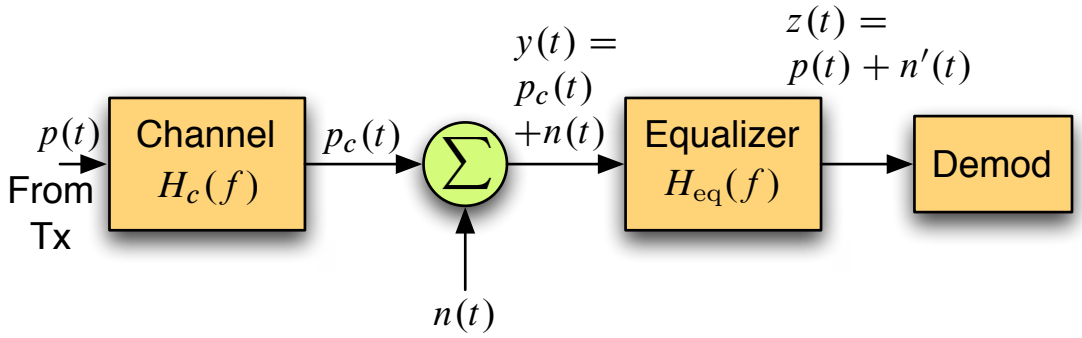
- What is $\overline{Z}$ again?
  - It is the average value of E_b/N_0, sometimes written as $\overline{E}_b/N_0$



BEP for BPSK, FSK, DPSK, and NCFSK with slow fading

5.11 Equalization

- An equalizer can be used to correct for distortions caused by multipath fading and/or bandlimiting due to filters, e.g., $H_C(f)$
- The idea is we wish to form an inverse filter to undo channel induced distortions



The equalizer concept

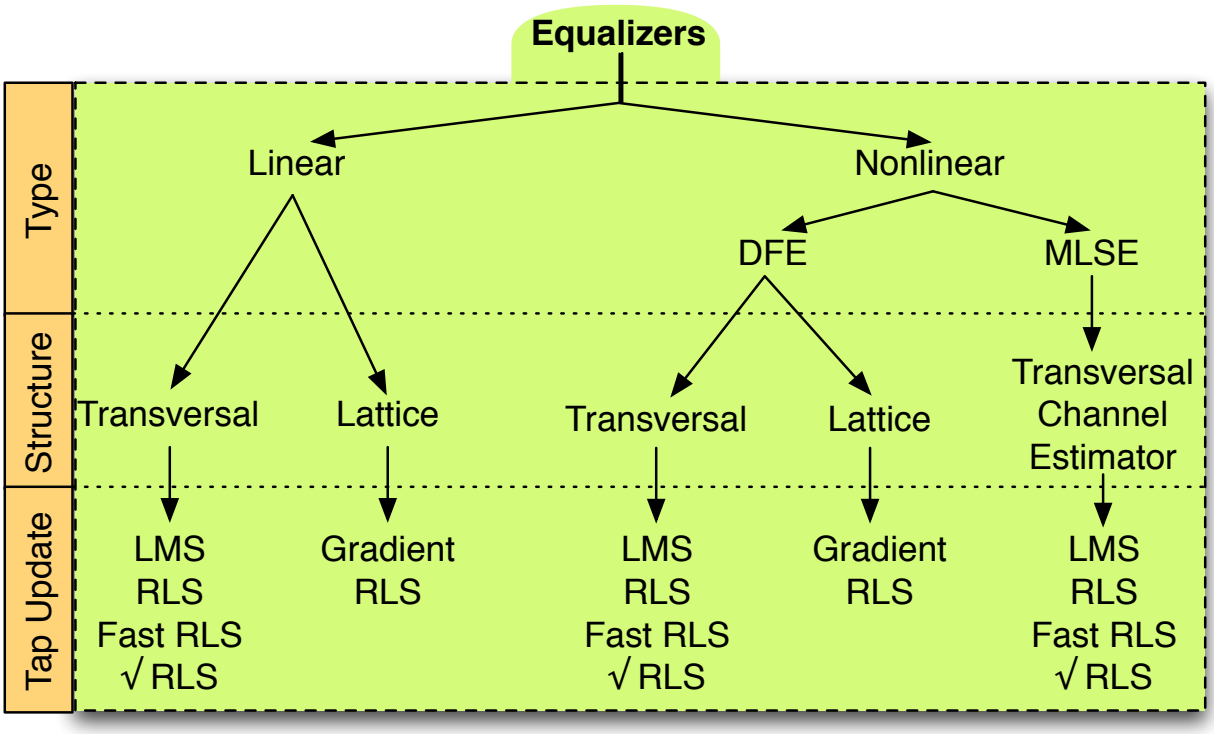
- To remove the ISI introduced by the channel we let

$$H_{eq}(f) = 1/H_c(f)$$

- The noise in $y(t)$ is now colored, and in particular if $H_c(f)$ contains a spectral null $S_{n'}(f)$ will be infinite at some frequency
- Even without spectral nulls the channel may greatly attenuate the signal over some frequency bands, in which case the noise will be greatly enhanced as the equalizer tries to bring the signal back
- In spite of reducing the ISI to at or near zero, the performance will still be poor because of SNR at the post equalization point
- Thus, equalization must balance ISI mitigation along with maximization the post-detection SNR

5.11.1 Equalizer Types

- Linear and nonlinear equalizers exist that can balance minimizing ISI while also keeping the SNR high
- The linear techniques are easiest to implement and understand, but often suffer from noise enhancement
- Nonlinear equalizers, such as *decision feedback equalization* (DFE) are often preferred, but readily implemented
- A summary of equalizer types is given below[5]



Equalizer types

- *Maximum likelihood sequence estimation* (MLSE) is optimal, but it is very complex, especially as the duration of the chan-

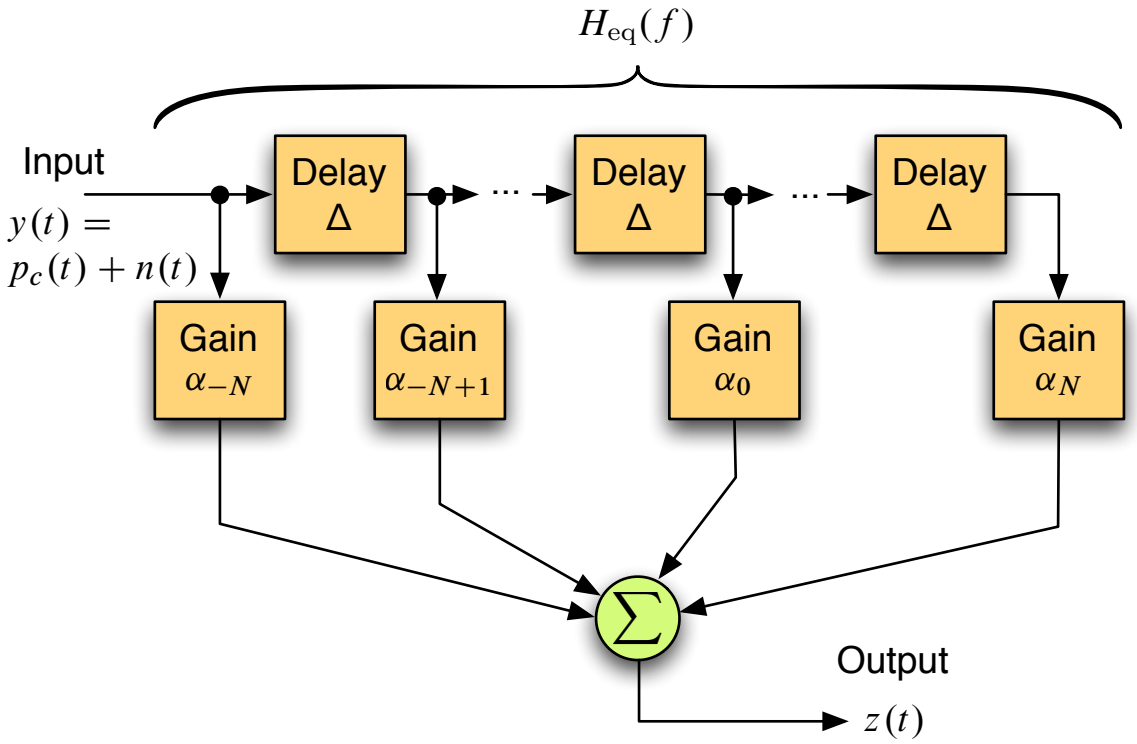
[5] Andrea Goldsmith, *Wireless Commuications*, Cambridge University Press, New York, 2005

nel impulse response grows large relative to the bit (symbol) period

- Linear techniques and the DFE, operate in a symbol-by-symbol (SBS) fashion, where ISI is removed from each symbol and then the symbol is detected

- With MLSE entire sequences are estimated using a trellis structure[6]

- The *tapped delay line* or *transversal filter* is the most common structure; in DSP this is just an FIR filter



Transversal (or in DSP FIR) filter for ISI mitigation

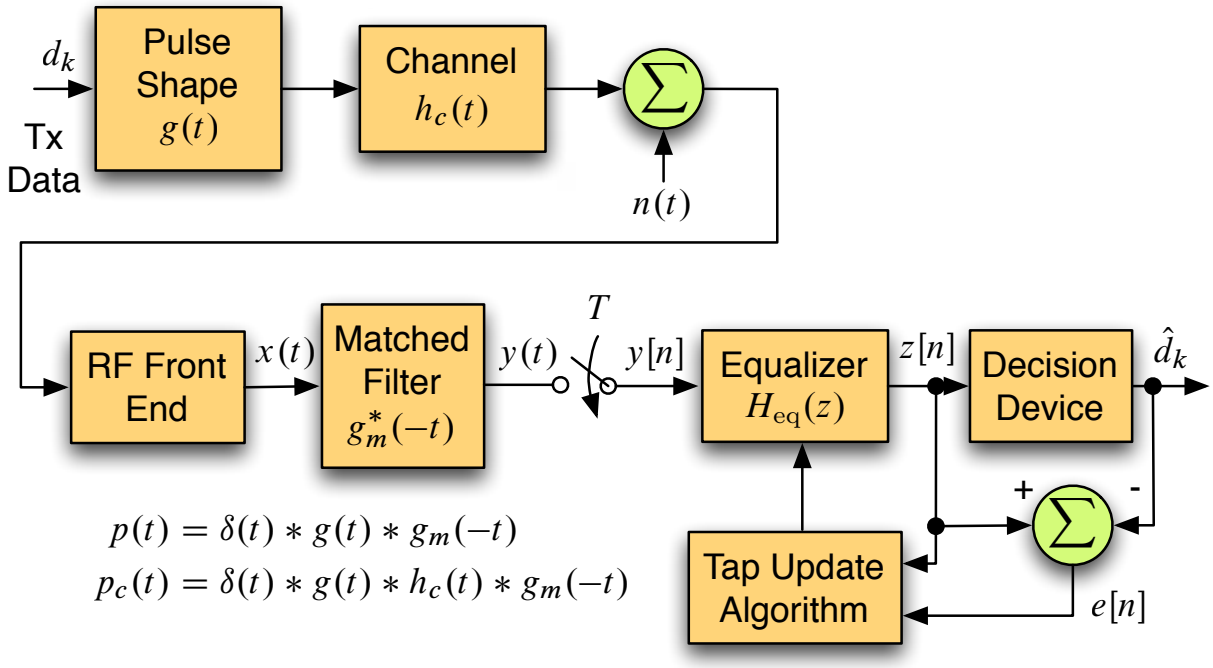
- The filter coefficients, denoted here as $\alpha_{-N}, \ldots, \alpha_0 \ldots, \alpha_N$, or

[6]B. Sklar"How I Learned to Love the Trellis," *IEEE Signal Processing Magazine*, May 2003, pp. 88–102.

perhaps $\alpha_0, \ldots, \alpha_{2N}$ for a $2N + 1$ tap FIR filter, can be determined for linear equalizers using two basic criterion:

1. *Zero-forcing* (ZF)
2. *Minimization of the mean-squared error* (MMSE)

- The DSP implementation of an equalizer at complex baseband takes the following form:



Complex baseband equalizer general form

- Suppose that $g(t)$ is chosen to give a particular transmit pulse spectrum in an ideal channel, then the matched filter is chosen to give zero ISI

- The channel impulse response ($h_c(t) \neq \delta(t)$) is likely time varying, so it is not practical to have $g_m(t)$ matched to $g(t) * h_c(t)$

5.11.2 Equalization by Zero Forcing

- With zero forcing we make the pulse response of the channel have value 1 at the desired sampling time and zero at N samples either side of the unity sample value
- We have $2N + 1$ unknowns so we need to set up a system of $2N + 1$ equations

$$p_{\text{eq}}(mT) = \sum_{n=-N}^{N} \alpha_n p_c\big[(m-n)T\big] = \begin{cases} 1, & m = 0 \\ 0, & m \neq 0 \end{cases} \quad m = 0, \pm 1, \ldots, \pm N$$

- In matrix form the system of equations is

$$\mathbf{P}_{\text{eq}} = \big[P_{\text{eq}}\big] = \big[P_c\big][A] = \mathbf{P}_c\mathbf{A}$$

where

$$\mathbf{P}_{\text{eq}} = \big[\underbrace{0\,0\,\cdots\,0}_{N \text{ zeros}}\ 1\ \underbrace{0\,0\,\cdots\,0}_{N \text{ zeros}}\big]^T$$

$$\mathbf{A} = \begin{bmatrix} \alpha_{-N} & \alpha_{-N+1} & \cdots & \alpha_{-1} & \alpha_0 & \alpha_1 & \cdots & \alpha_N \end{bmatrix}^T$$

$$\mathbf{P}_c = \begin{bmatrix} p_c(0) & p_c(-T) & \cdots & p_c(-2NT) \\ p_c(T) & p_c(0) & \cdots & p_c((-2N+1)T) \\ \vdots & & & \vdots \\ p_c(2NT) & & \cdots & p_c(0) \end{bmatrix}$$

- Because of the way $p_{\text{eq}}(mT)$ is defined, the optimum solution for $\mathbf{A}$ turns out to be

$$\mathbf{A}_{\text{opt}} = \mathbf{P}_c^{-1}\mathbf{P}_{\text{eq}} = \text{middle column of } \mathbf{P}_c^{-1}$$

- Forcing zeros in the equalizer response at $t = \pm T, \ldots, \pm NT$ does not say anything about the response outside the interval and since signal only was used in the zero forcing, *noise enhancement* may occur (lots of gain over a band of frequencies where the channel gain is weak)

- The frequency response of the equalizer, for symbol spaced taps, i.e., $\Delta = T$, is the DTFT of the filter coefficients

$$H_{\text{eq}}(e^{j\Omega}) = \sum_{n=-N}^{N} \alpha_n e^{-j\Omega n},$$

  or in terms of the analog frequency variable f

$$H_{\text{eq}}(f) = \sum_{n=-N}^{N} \alpha_n e^{-j2\pi nfT}$$

- To assess the noise impact we can calculate the noise power at the output of the equalizer assuming a bandlimited Gaussian noise input having PSD of $N_0/2$ for $|f| < B$, where B is the assumed bandwidth that the signal occupies, e.g., for RC spectra pulses about R_b (two-sided)

- Composite signal plus noise output, when symbol spaced, is

$$\begin{aligned} z(mT) &= \sum_{l=-N}^{N} \alpha_l \big\{ p_c[(m-l)T] + n[(m-l)T] \big\} \\ &= p_{\text{eq}}(mT) + \underbrace{\sum_{l=-N}^{N} \alpha_l n[(m-l)T]}_{N_m} \end{aligned}$$

Note: The RV $\{N_m\}$ are Gaussian having zero mean and variance

$$\begin{aligned}
\sigma_N^2 &= E\{N_k^2\} \\
&= E\left\{\sum_{j=-N}^{N} \alpha_j n[(k-j)T] \sum_{l=-N}^{N} \alpha_l n[(k-l)T]\right\} \\
&= E\left\{\sum_{j=-N}^{N}\sum_{l=-N}^{N} \alpha_j \alpha_l n[(k-j)T] n[(k-l)T]\right\} \\
&= \sum_{j=-N}^{N}\sum_{l=-N}^{N} \alpha_j \alpha_l E\{n[(k-j)T] n[(k-l)T]\} \\
&= \sum_{j=-N}^{N}\sum_{l=-N}^{N} \alpha_j \alpha_l R_n[(j-l)T]
\end{aligned}$$

where for the bandlimited noise considered here,

$$R_n(\tau) = N_0 B \operatorname{sinc}(2B\tau)$$

- Assuming that $B = R_b/2 = 1/(2T)$, then $2BT = 1$ and

$$R_n[(j-l)T] = \frac{N_0}{2T}\operatorname{sinc}(j-l) = \begin{cases} \frac{N_0}{2T}, & j = l \\ 0, & j \neq l \end{cases}$$

then

$$\sigma_N^2 = \frac{N_0}{2T}\sum_{j=-N}^{N} \alpha_j^2$$

- To complete the analysis of noise enhancement, we can assume a ± 1 equally likely binary transmission with the ISI com-

pletely removed

$$P_E = \frac{1}{2}P[N_m > 1|\text{-1 sent}] + \frac{1}{2}P[N_m < 1|\text{+1 sent}]$$
$$= \int_1^{\infty} \frac{\exp\left(-\eta^2/(2\sigma_N^2)\right)}{\sqrt{2\pi\sigma_N T2}}\, d\eta = Q\left(\frac{1}{\sigma_N}\right)$$
$$= Q\left(\frac{1}{\sqrt{\frac{N_0}{2T}\sum_j \alpha_j^2}}\right) = Q\left(\sqrt{\frac{1}{\sum_j \alpha_j^2}\cdot\frac{2E_b}{N_0}}\right)$$

- For an ideal channel, i.e., $h_c(t) = \delta(t)$, $\alpha_0 = 1$ and the others coefficients are zero, and there is no degradation
- In general we expect $\sum_j \alpha_j^2 \geq 1$

Example 5.11: MATLAB Zero Forcing Example

- In this example we will re-work the `shaped_psk_sim` code presented earlier; the new function will be `shaped_psk_ZFEQ_sim`
- The channel filter will be a 4th-order Butterworth filter with 3dB cuttoff frequency set to $R_b/2$ in complex baseband (equivalent to R_b in bandpass domain)
- A local function `zero_force()` is created inside the main simulation routine to design the zero forcing equalization filter

```
function [Pe,errors,N_RecBits,z,b_eq] = shaped_psk_ZFEQ_sim(SNR,Nbits,timing,beta,shape)
% [Pe,errors,N_RecBits,z,b_eq] = shaped_psk_ZFEQ_sim(SNR,m,theta,Nbits,timing)
%
% Coherent Binary BPSK modem ZF-EQ end-to-end simulation function
%////////////////////////////////////////////////////////////////////
%      SNR = Set the received signal SNR in dB; for on-off keying the SNR
%            is defined interms of the average signal power, e.g., 1/2 the
%            peak symbol energy
```

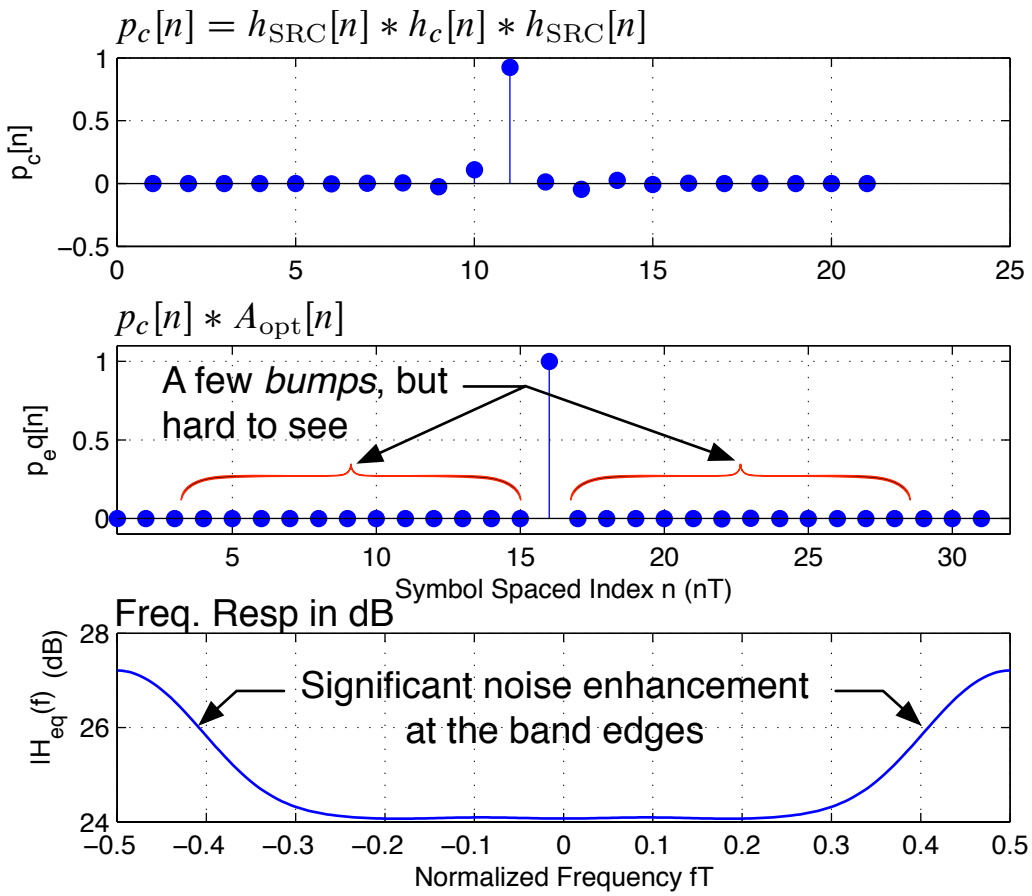
```
%    Nbits = The number of bits simulated
%   timing = Sets the sampler timing to give the minimum BEP; find value
%            using eyeplot_mark function
% rec_type = Receiver type: 'coherent', bpf_env_det', or 'lim_discrim'
%
%       Pe = Estimated bit error probability (BEP)
%   errors = Number of bit errors found (good to obtain at least 100
%            errors over an AWGN channel
%  RecBits = The number of bits processed y bit_errors function; If you
%            create a 'top-level' function 'errors' and 'RecBits' can be
%            used to combine results from multiple runs
%        z = Receiver decision statistic; use as input to eyeplot_mark for
%            further analysis, e.g., find a good value for 'timing'
%////////////////////////////////////////////////////////////////////

% Mark Wickert November 2010

Ns = 16; % Fix the number of samples per bit to 16

% Generate PSK tx signal
[x,data,b] = shaped_psk(Nbits, 16, beta, shape);

% Introduce channel distortion via a 4th-order Butterworth lowpass filter.
% Set fc = 0.5 times the bit rate.
[b_chan,a_chan] = butter(4,2* 0.5/Ns);
x = filter(b_chan,a_chan,x);

% AWGN channel
y = cpx_AWGN(x,SNR,16);

% Enter receiver

% Coherent receiver (at complex baseband)
%//////////////////////////////////
% Matched filter is designed by the Tx function as vector b
% Process complex BB signal through filter to form decision statistic
z = filter(b,1,y)/Ns;

%//////////////////////////////////
% Design a zero forcing equalizer using side information
b_eq = zero_force(b,b_chan,a_chan,Ns,5); % Ntaps = 2N+1 = 5*2+1 = 11
% Apply as an upsampled filter to better see timing needs
z = filter(upsample(b_eq,Ns),1,z);

z_det = real(z);
% Decision logic
```

```
d_hat = (sign(z_det(timing:16:end))+1)/2; % 0,1 logic levels
% Error detect
[Pe,errors,N_RecBits] = bit_errors(data,d_hat);

%//////////////////////////////////////////////////////////////////////
function Aopt = zero_force(tx_pulse,b_chan,a_chan,Ns,N)
% Aopt = zero_force(tx_pulse,b_chan,a_chan,Ns,N)
%% Zero Force Function
%
% Mark Wickert November 2011
%//////////////////////////////////////////////////////////////////////

Npre = N*Ns; Npost = N*Ns;
% Zero pad either side of pulse
p = [zeros(1,Npre*Ns) tx_pulse zeros(1,Npost*Ns)];
p = filter(b_chan,a_chan,p); % Filter twice to get end-to-end
p = filter(tx_pulse,1,p)/Ns; % (Tx/Rx pulse together) response
% Find the max
[p_max,I_max] = max(p);

p_vec = p(I_max-Ns*2*N:Ns:I_max+Ns*2*N); % Symbol spaced channel samples

%p_vec = fliplr(p_vec);
N_tap = 2*N+1;
Pc = zeros(N_tap,N_tap);
for k=1:2*N+1
    % Fill up the Pc matrix with samples channel pulse response samples
    Pc(:,k) = p_vec(2*N+1-(k-1):2*N+1-(k-1)+(N_tap-1)).';
end

Peq = [zeros(1,N) 1 zeros(1,N)]';

Aopt = Pc[backslash]Peq; % direct solve without using inv( )
%//////////////////////////////////////////////////////////////////////
```

- At present a 10-tap filter is set to be designed
- We run the simulation function and set breakpoints inside to test the equalizer design by stopping at line 51 to compare the eye plots before and after equalization

```
>> [Pe,errors,N_RecBits,z,b_eq] = shaped_psk_ZFEQ_sim(100,10000,1,...
                                  0.5,'src');
End of function shaped_psk_ZFEQ_sim>zero_force.
```

```
K>> subplot(311)
K>> stem(p_vec,'filled')
K>> subplot(312)
K>> stem(conv(p_vec,Aopt),'filled')
K>> subplot(313)
K>> f = -.5:1/200:.5;
K>> H = freqz(Aopt,1,2*pi*f); % Gain scale by Ns
K>> plot(f,abs(H))
K>> plot(f,20*log10(abs(H)))
```

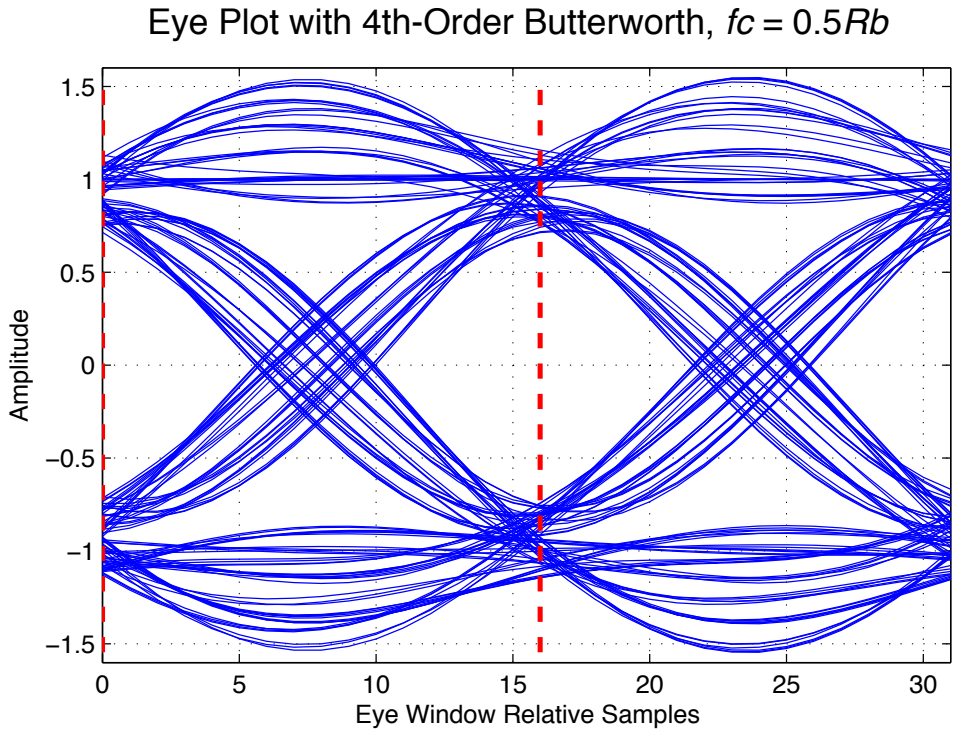


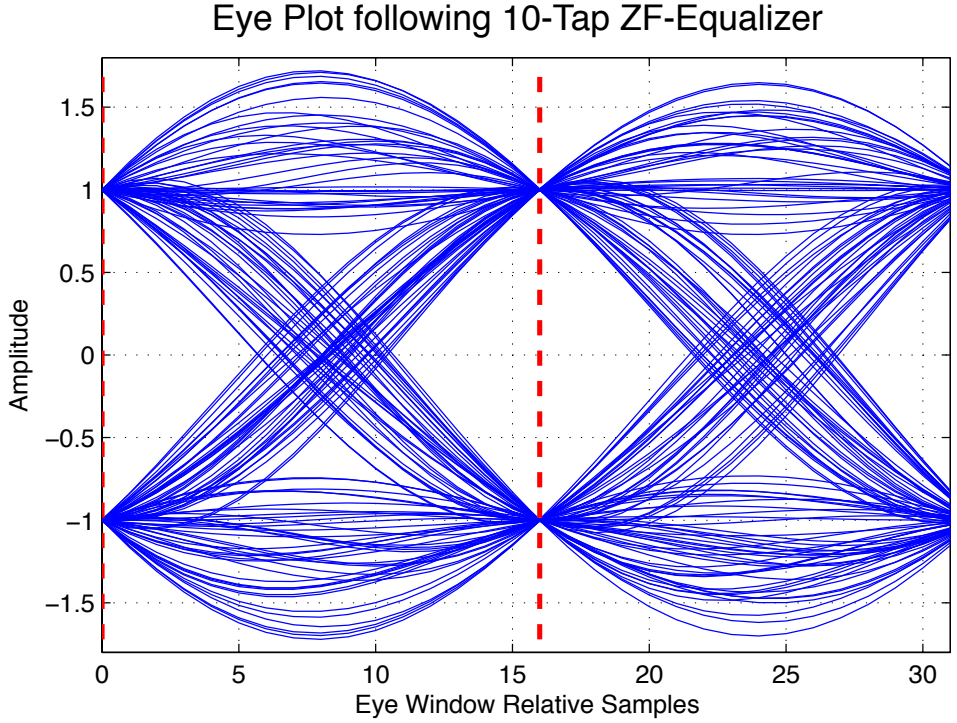
10-tap ZF-EQ: $p_c[n]$, $p_C[n] * A_{\mathrm{opt}}[n]$, $A_{\mathrm{opt}}(e^{j2\pi fT})$

- Next we consider the eye plot with and without the equalizer

```
51  z = filter(upsample(b_eq,Ns),1,z);
K>> eyeplot_mark(real(z(10*16+1:5000)),2*16,192,16,16);
K>> print -tiff -depsc ZF_EQ2.eps
////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 9982
```

```
       BEP: 0.00e+00
////////////////////////////////////////////////////////
>> eyeplot_mark(real(z(10*16+1:5000)),2*16,192,16,16);
```





Eye plot without (top) and with (bottom) 10-tap equalizer

- The noise enhancement factor, using the expression developed in the text/notes is

```
>> 10*log10(sum((b_eq).^2))

ans =   8.4461e-01 % in dB
```

- BEP simulation can be used to compare the performance with and without the equalizer (reader assignment!)

5.11.3 Equalization by Minimum MSE

- A linear equalizer type that mitigates the issues of noise enhancement can be formulated using a *minimum mean-squared error* (MMSE) criterion

- Suppose that the desired signal at the output of the equalizer is given by $d(t)$ (don't worry, we can assume a training signal is available)

- We seek the taps weights such that

$$\epsilon = E\left[e^2(t)\right] = E\left[\left(z(t) - d(t)\right)^2\right] = \text{minimum}$$

  where $z(t)$ is the equalizer output

$$z(t) = \sum_{n=-N}^{N} \alpha_n y(t - n\Delta)$$

- The error surface formed by ϵ as a function of the tap weights is *convex cup* or bowl shaped

- The mean-square error (MSE) is a function of both ISI and AWGN present at the input
- The optimal in the MMSE sense tap weights can be found by setting

$$\frac{\partial \epsilon}{\partial \alpha_m} = 0 = 2E\left[\big(z(t) - d(t)\big)\frac{\partial z(t)}{\partial \alpha_m}\right],\ m = 0, \pm 1, \ldots, \pm N$$

- Making substitutions and algebraic manipulations, results in

$$R_{yz}(m\Delta) = R_{yd}(m\Delta) = 0,\ \ m = 0, \pm 1, \ldots, \pm N$$

where

$$R_{yz}(\tau) = E[y(t+\tau)z(t)]$$
$$R_{yd}(\tau) = E[y(t+\tau)d(t)]$$

- Since $R_{yz}(\tau)$ is just $R_{\text{yy}}(\tau) * a_{\text{opt}}(\tau)$ ($a_{\text{opt}}(t) = h_{\text{eq}}(t)$), we can write using compact matrix notation that

$$\mathbf{R}_{yy}\mathbf{A}_{\text{opt}} = \big[R_{yy}\big]\big[A\big]_{\text{opt}} = \big[R_{yd}\big] = \mathbf{R}_{yd}$$

where

$$\mathbf{R}_{yy} = \begin{bmatrix} R_{yy}(0) & R_{yy}(\Delta) & \cdots & R_{yy}(2N\Delta) \\ R_{yy}(-\Delta) & R_{yy}(0) & \cdots & R_{yy}((2N-1)\Delta) \\ \vdots & & & \vdots \\ R_{yy}(-2N\Delta) & & \cdots & R_{yy}(0) \end{bmatrix}$$

$$\mathbf{R}_{yd} = \big[R_{yd}(-N\Delta)\ \ R_{yd}(-(N-1)\Delta)\ \ \cdots\ \ R_{yd}(N\Delta)\big]^T$$

- The solution for the tap weight vector (equalizer filter) is

$$\mathbf{A}_{\text{opt}} = \mathbf{R}_{yy}^{-1}\mathbf{R}_{yd}$$

- The formulation under the MMSE criterion is similar to zero forcing, except now we use a correlation matrix as opposed to the pulse response
- Having perfect knowledge of $\mathbf{R}_{yy}$ and $\mathbf{R}_{yd}$ is desired, but not essential
- As a performance measure we need to calculate the MMSE, ϵ
- It can be shown that in general

$$\epsilon = \sigma_d^2 - 2\mathbf{A}^T\mathbf{R}_{yd} + \mathbf{A}^T\mathbf{R}_{yy}\mathbf{A}$$

- When the optimum solution is employed we have

$$\epsilon_{\text{min}} = \sigma_d^2 - \mathbf{R}_{yd}^T\mathbf{A}_{\text{opt}}$$

- We have yet to specify Δ, although, we could again choose symbol spacing, that is let $\Delta = T$
- It is common to consider a *fractionally spaced equalizer* (FSE), that is $\Delta = T/K$, K an integer, when exact symbol rate samples are not available
  - The FSE increases the complexity, but allowing multiple samples per bit (symbol) it can also take on the role of the matched filter

Tap Weight Adjustment

- How do we get the tap weights in practice?
- At present the calculation requires knowledge of $d(t)$
    1. A periodic training sequence can be sent which can be used to adjust the tap weights
    2. If the operating SNR is relatively high, the detected data can be used by what is known as a *decision directed* (DD) implementation; before DD is engaged a training sequence may also be used
- The topic of *blind equalization* focuses on how to obtain the tap weights without explicit knowledge of a training vector
- Another issue is how to obtain the correlation function $R_{yy}(\tau)$ (we just need sample values)?
- This opens the door to the topic of *adaptive equalization*
- Since we know the error surface is bowl shaped, we just need a starting point (initial guess for the tap weight vector) and then move along on the error surface using the *method of steepest descent*

$$\mathbf{A}^{(k+1)} = \mathbf{A}^{(k)} + \frac{1}{2}\mu\big(-\nabla\epsilon^{(k)}\big), \quad k = 0, 1, 2, \ldots$$

where ∇ is the gradient operator and μ is a step size parameter to insure a stable convergence

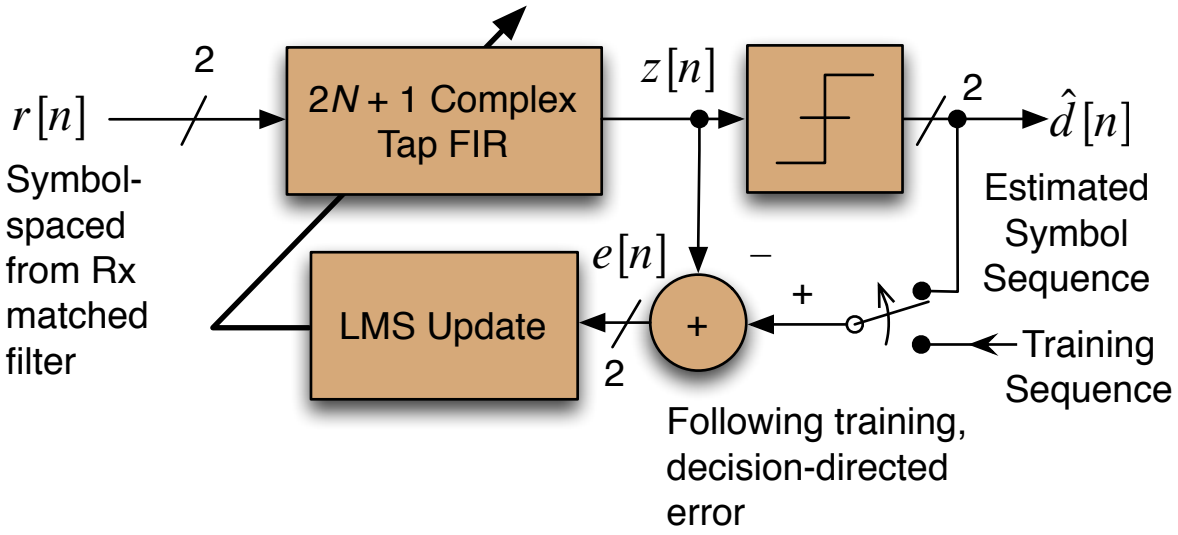
- In practice the true statistical auto- and cross-correlation functions are not available, so as an approximation, it is common for use the *stochastic gradient* which is an instantaneous time estimate
- The result is the *least mean-square* (LMS) algorithm, which states that at each time step we update the tap weights according to

  $$a_m^{(k+1)} = a_m^{(k)} - \mu y\big[(k-m)\Delta\big]\epsilon(k\Delta), \quad m = 0, \pm 1, \pm 2, \ldots, \pm N$$

  where $\epsilon(k\Delta) = z(k\Delta) - d(k\Delta)$
- MATLAB has a large set of functions for various equalization techniques; in Python you have to write your own

Example 5.12: MATLAB Decision Directed MMSE Example

- In this example we will re-work the `shaped_psk_sim` code presented earlier; the new function will be `shaped_psk_DDEQ_sim`; Note: Python version soon.



Complex-Baseband Decision Feedback Equalizer

- The channel filter will be a 4th-order Butterworth filter with 3dB cuttoff frequency set to $R_b/2$ in complex baseband (equivalent to R_b in bandpass domain)
- A local function `DD_LMS()` is created inside the main simulation routine to design the zero forcing equalization filter

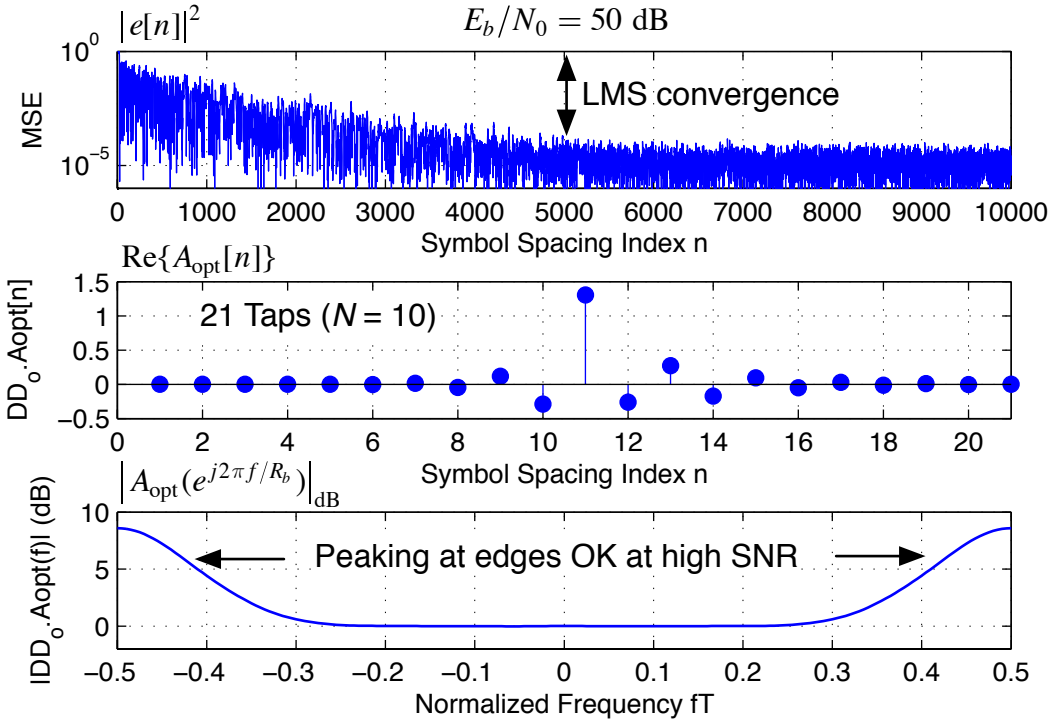
```
function [Pe,errors,N_RecBits,z,DD_o] = ...
              shaped_psk_DDEEQ_sim(SNR,Nbits,timing,beta,shape,DD_i)
% [Pe,errors,N_RecBits,z,DD_o] =
              shaped_psk_DDEQ_sim(SNR,Nbits,timing,beta,shape,DD_i)
%
% Coherent Binary BPSK modem ZF-EQ end-to-end simulation function
%////////////////////////////////////////////////////////////////////
%      SNR = Set the received signal SNR in dB; for on-off keying the SNR
%            is defined interms of the average signal power, e.g., 1/2 the
%            peak symbol energy
%    Nbits = The number of bits simulated
%   timing = Sets the sampler timing to give the minimum BEP; find value
%            using eyeplot_mark function
% rec_type = Receiver type: 'coherent', bpf_env_det', or 'lim_discrim'
%     DD_i = structure variable to hold DD-LMS parameters:
%            DD_i.N  = number of single-sided taps; total taps = 2*N+1
%            DD_i.mu = LMS algorithm convergence parameter, around 1e-3
%            DD_i.Nstart = Start BEP measuring with a delay due to LMS
%                          not convergenced yet, 500 to 1000 good
%                          but check MSE plot
%
%       Pe = Estimated bit error probability (BEP)
%   errors = Number of bit errors found (good to obtain at least 100
%            errors over an AWGN channel
%  RecBits = The number of bits processed y bit_errors function; If you
%            create a 'top-level' function 'errors' and 'RecBits' can be
%            used to combine results from multiple runs
%        z = Receiver decision statistic; use as input to eyeplot_mark for
%            further analysis, e.g., find a good value for 'timing'
%     DD_o = DD-LMS output variables:
%            DD_o.z_eq = Equalizer output at symbol spacing
%            DD_o.d_hat = Hard decision data
%            DD_o.e = Error sequence (complex)
%            DD_o.Aopt = Tap weigh vector at end of simulation (complex)
%////////////////////////////////////////////////////////////////////

% Mark Wickert November 2010
```

```
Ns = 16; % Fix the number of samples per bit to 16

% Generate PSK tx signal
[x,data,b] = shaped_psk(Nbits, 16, beta, shape);

% Introduce channel distortion via a 4th-order Butterworth lowpass filter.
% Set fc = 0.5 times the bit rate.
[b_chan,a_chan] = butter(4,2* 0.5/Ns);
x = filter(b_chan,a_chan,x);

% AWGN channel
y = cpx_AWGN(x,SNR,16);

% Enter receiver

% Coherent receiver (at complex baseband)
%//////////////////////////////
% Matched filter is designed by the Tx function as vector b
% Process complex BB signal through filter to form decision statistic
z = filter(b,1,y)/Ns;

%//////////////////////////////
% Design a MMSE equalizer using a DD-LMS
[DD_o.z_eq,DD_o.d_hat,DD_o.e,DD_o.Aopt] = DD_LMS(z(timing:16:end),...
                                          Ns,DD_i.N,DD_i.mu);
% Apply as an upsampled filter to better see timing needs
z = filter(upsample(DD_o.Aopt,Ns),1,z);

z_det = real(z);
% Decision logic
d_hat = (sign(z_det(timing:16:end))+1)/2; % 0,1 logic levels
% Error detect
[Pe,errors,N_RecBits] = bit_errors(data(1+DD_i.Nstart:end-(DD_i.N)+1),...
                                   d_hat(DD_i.Nstart+DD_i.N:end));

%//////////////////////////////////////////////////////////////////////
function [z_eq,d_hat,e,Aopt] = DD_LMS(y,Ns,N,mu)
% [z_eq,e,Aopt] = DD_LMS(y,Ns,N,mu)
%% DD LMS Function
%
% Mark Wickert November 2011
%//////////////////////////////////////////////////////////////////////

Nsim = length(y);
```

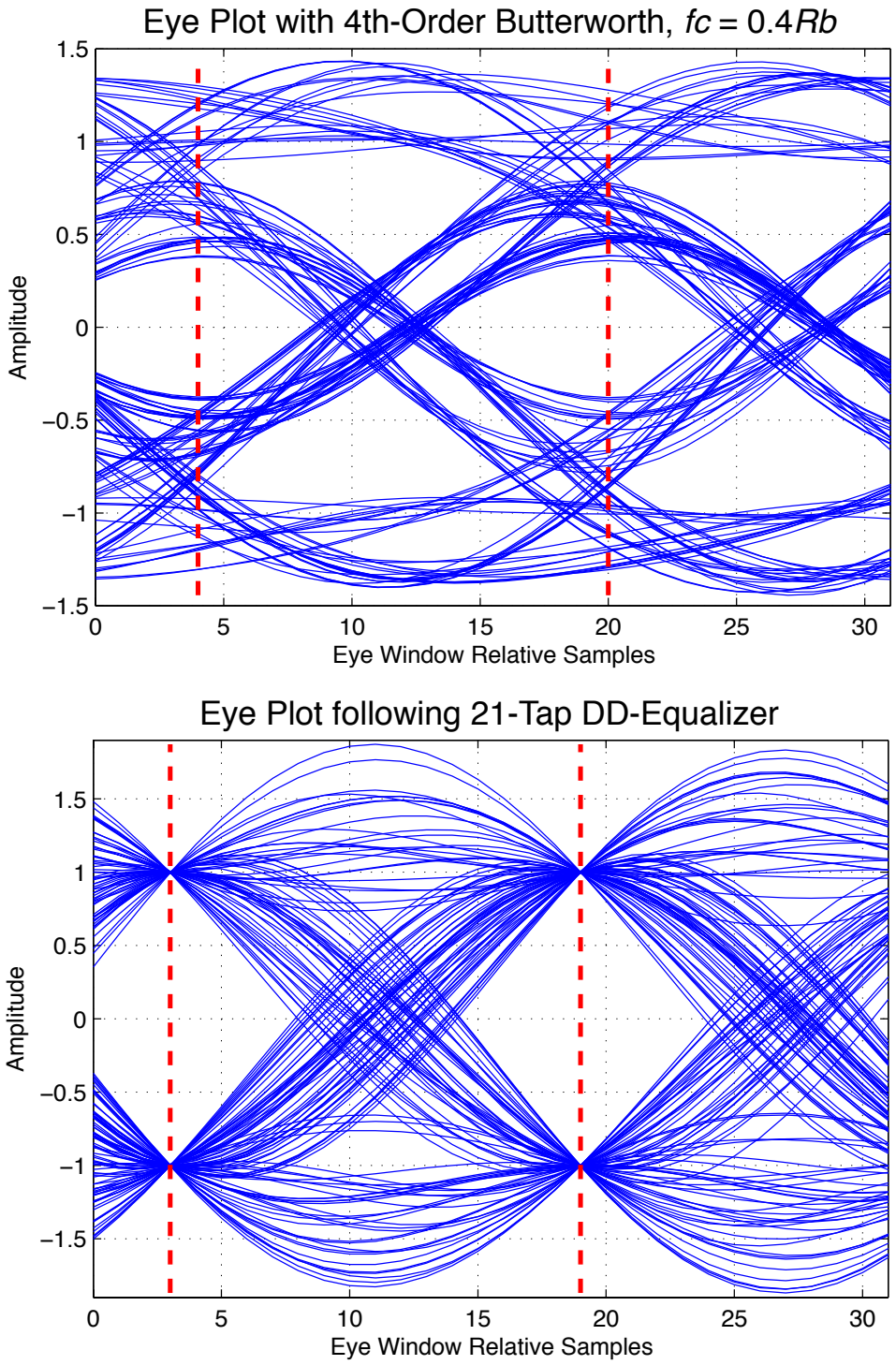
```
DD_A = 1.0;
constellation = [1 -1]*DD_A; % BPSK complex baseband constellation
a = 1; % 1/sqrt(2) for QPSK;
constellation = constellation*a;

z_eq = zeros(size(y));
d_hat = zeros(size(y));
e = zeros(size(y));
Aopt = [zeros(1,N) 1 zeros(1,N)];
y_states = zeros(1,2*N+1);

% Begin the LMS simulation sample-by-sample outer loop
% This is a symbol spaced equalizer, so there is only one sample per symbol
for k=1:Nsim
    % Update input state vector
    y_states = [y(k) y_states(1:end-1)];

    % Apply FIR Filter as a dot product, row times column
    z_eq(k) = z_eq(k) + Aopt*y_states.';

    % Form the hard decisions using the filter outputs
   DD_error = z_eq(k) - constellation;
   [min_error,index] = min(abs(DD_error));
   d_hat(k) = constellation(index); % recovered data in -1,1 format
   %d_hat.I(k) = fix(real(z_hat(k))/a);
   %d_hat.Q(k) = fix(imag(z_hat(k))/a);

   e(k) = d_hat(k) - z_eq(k);

%  Update the filter coefficients (LMS update eqn)
   Aopt = Aopt + mu*conj(y_states)*e(k);
end

test = 0;
%////////////////////////////////////////////////////////////////////
```

- At present an $N = 10$ (21-taps) filter is implemented
- We run the simulation function and set breakpoints inside to test the equalizer design by stopping at line 63 to compare the eye plots, examine `DD_o.Aopt` before and after equalization

```
>> [Pe,errors,N_RecBits,z,DD_o] =
                              shaped_psk_DDEQ_sim(50,10000,4,0.5,'src',DD_i);
63  z = filter(upsample(DD_o.Aopt,Ns),1,z);
K>> subplot(311)
K>> semilogy(abs(DD_o.e).^2)
K>> ylabel('MSE')
K>> xlabel('Symbol Spacing Index n')
K>> subplot(312)
K>> stem(real(DD_o.Aopt),'filled')
K>> subplot(313)
K>> f = -.5:1/200:.5;
K>> H = freqz(DD_o.Aopt,1,2*pi*f);
K>> plot(f,20*log10(abs(H)))
K>> print -tiff -depsc DDE_50dB.eps
K>> eyeplot_mark(real(z(12*16+1:5000)),2*16,192,16,5);
K>> print -tiff -depsc DDE_50dB_eye.eps
65  z_det = real(z);
K>> eyeplot_mark(real(z(12*16+1:5000)),2*16,192,16,4);
K>> print -tiff -depsc DDE_50dB_eye_after.eps
////////////////////////////////////////////////////////
Bit Errors: 0
Bits Total: 9477
       BEP: 0.00e+00
////////////////////////////////////////////////////////
```



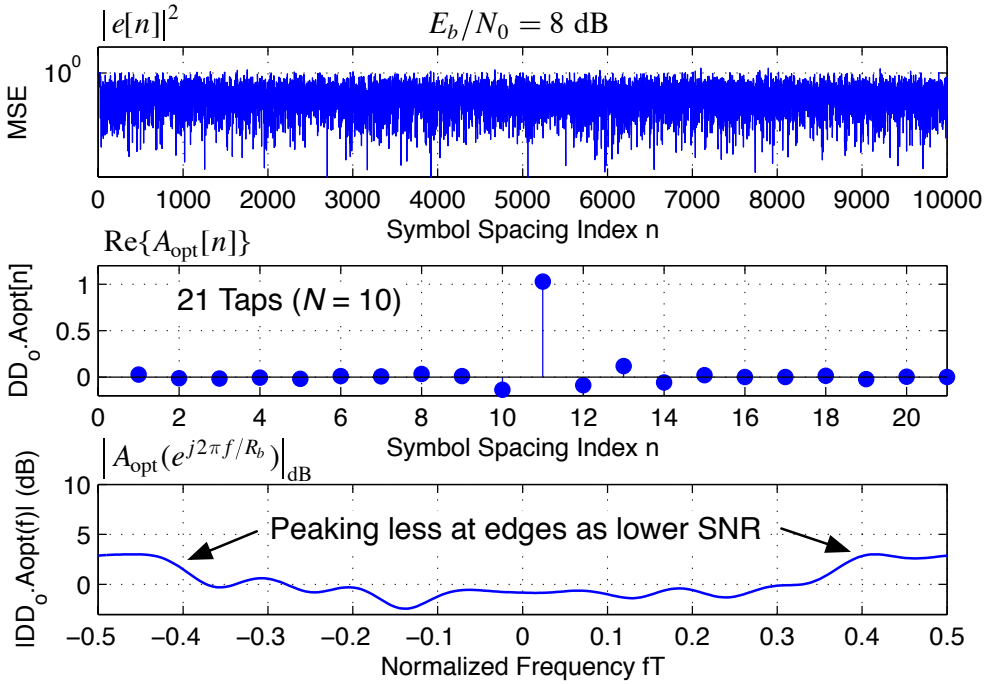
21-tap DD-MMSE-EQ, SNR=50dB: MSE, $\mathrm{Re}\{A_{\mathrm{opt}}[n]\}$, $A_{\mathrm{opt}}(e^{j2\pi fT})$



Eye plot without (top) and with (bottom) 21-tap equalizer

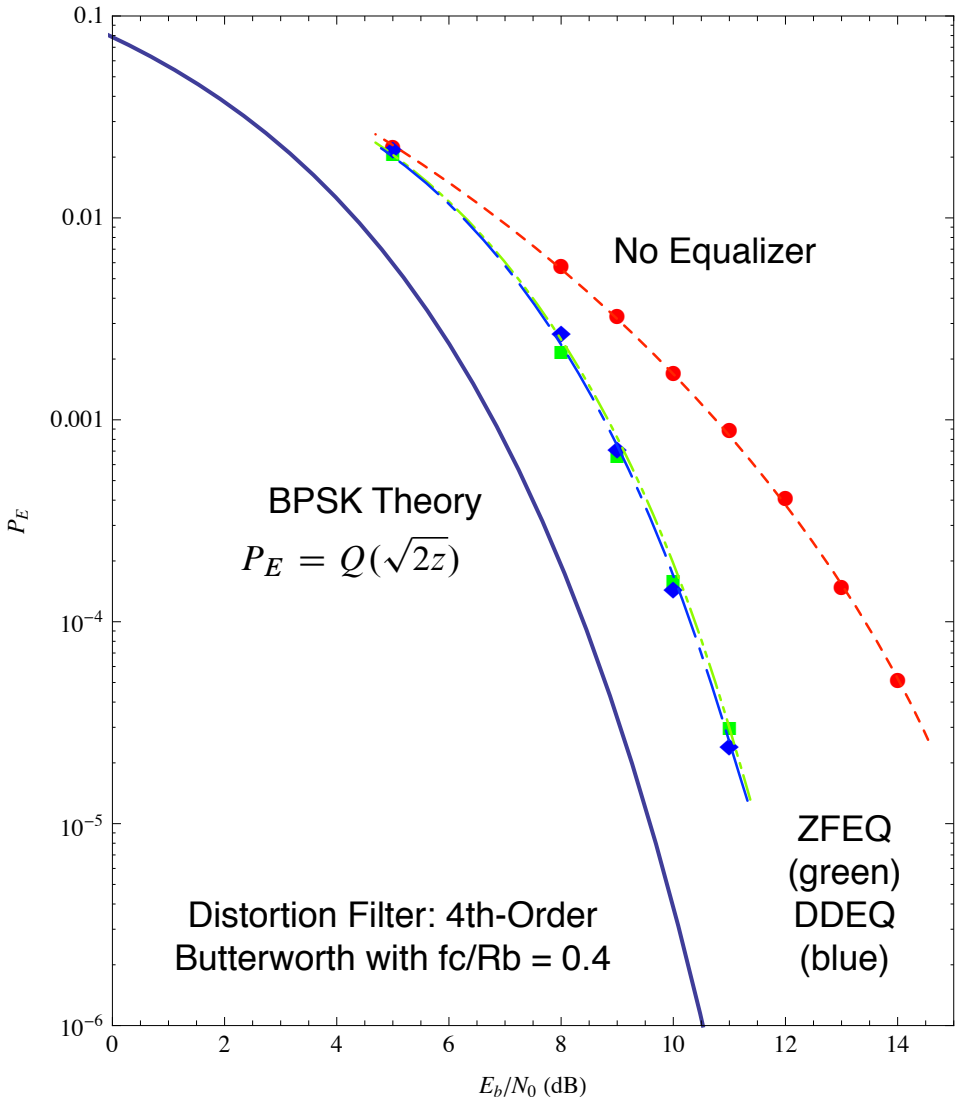
- To see how SNR influences the MMSE design we now consider $E_b/N_0 = 8$ dB

```
>> [Pe,errors,N_RecBits,z,DD_o] = shaped_psk_DDEQ_sim(8,10000,4,0.5,'src',DD_i);
63  z = filter(upsample(DD_o.Aopt,Ns),1,z);
K>> subplot(311)
K>> hold
Current plot held
K>> semilogy(abs(DD_o.e).^2,'r')
K>> clf
K>> subplot(311)
K>> semilogy(abs(DD_o.e).^2)
K>> axis([0 10000 1e-4 1])
K>> axis([0 10000 1e-4 1e1])
K>> grid
K>> xlabel('Symbol Spacing Index n')
K>> ylabel('MSE (dB)')
K>> ylabel('MSE')
K>> subplot(312)
K>> stem(real(DD_o.Aopt),'filled')
K>> axis([0 21 -.2 1.2])
K>> grid
K>> xlabel('Symbol Spacing Index n')
K>> ylabel('DD_o.Aopt[n]')
K>> subplot(313)
K>> f = -.5:1/200:.5;
K>> H = freqz(DD_o.Aopt,1,2*pi*f);
K>> plot(f,20*log10(abs(H)))
K>> grid
K>> axis([-.5 .5 -3 10])
K>> xlabel('Normalized Frequency fT')
K>> ylabel('|DD_o.Aopt(f)| (dB)')
K>> print -tiff -depsc DDE_50dB2.eps
```



21-tap DD-MMSE-EQ, SNR=8dB: MSE, $\mathrm{Re}\{A_{\mathrm{opt}}[n]\}$, $A_{\mathrm{opt}}(e^{j2\pi fT})$

- Next we consider a BEP comparison plot of the system without equalization, with DD-MMSE and ZFEQ, all for the case of a channel filter being a 4th-order Butterworth with $f_c/R_b = 0.4$
- The performance is pulled in about 3.5 dB by the equalizer around 10^{-5}
- The performance improvement at 10^{-6} would be even greater, but data was not taken at this level



BEP comparison curves

Example 5.13: MATLAB Toolbox Functions

Equalizer Features of Communications Toolbox Software

This toolbox supports these distinct classes of equalizers, each with a different overall structure:

- Linear equalizers, a class that is further divided into these categories:
  - Symbol-spaced equalizers
  - Fractionally spaced equalizers (FSEs)
- Decision-feedback equalizers (DFEs)
- MLSE (Maximum-Likelihood Sequence Estimation) equalizer that uses the Viterbi algorithm. To learn how to use the MLSE equalizer capabilities, see Using MLSE Equalizers.

Linear and decision-feedback equalizers are adaptive equalizers that use an adaptive algorithm when operating. For each of the adaptive equalizer classes listed above, this toolbox supports these adaptive algorithms:

- Least mean square (LMS)
- Signed LMS, including these types: sign LMS, signed regressor LMS, and sign-sign LMS
- Normalized LMS
- Variable-step-size LMS
- Recursive least squares (RLS)
- Constant modulus algorithm (CMA)

To learn how to use the adaptive equalizer capabilities, start with Using Adaptive Equalizer Functions and Objects. For brief background material on the supported adaptive equalizer types, see Overview of Adaptive Equalizer Classes. For more detailed background material, see the works listed in Selected Bibliography for Equalizers.

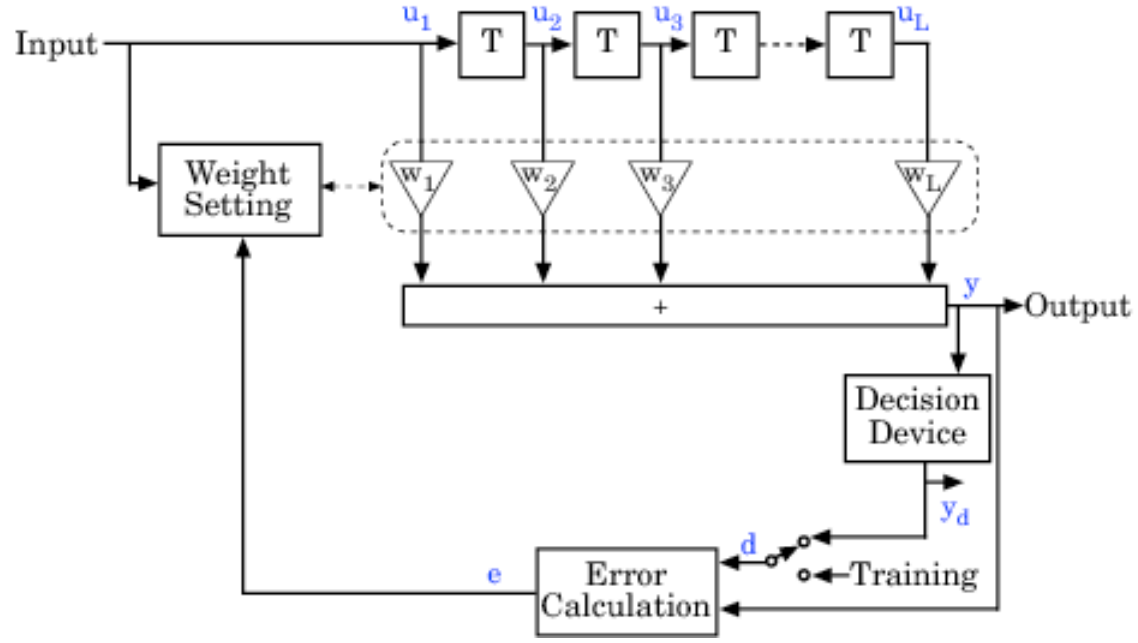
Provide feedback about this page

Equalizers | Overview of Adaptive Equalizer Classes

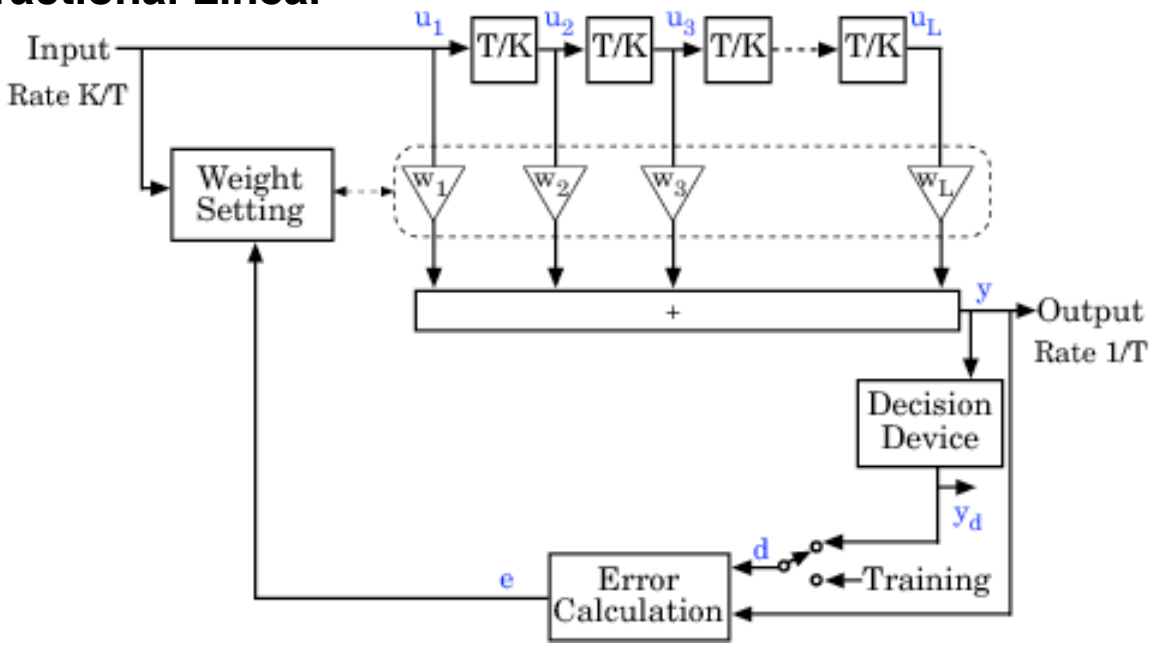
MATLAB equalizer function block diagrams

- The block diagram of common structures implemented in the MATLAB toolbox are shown below:

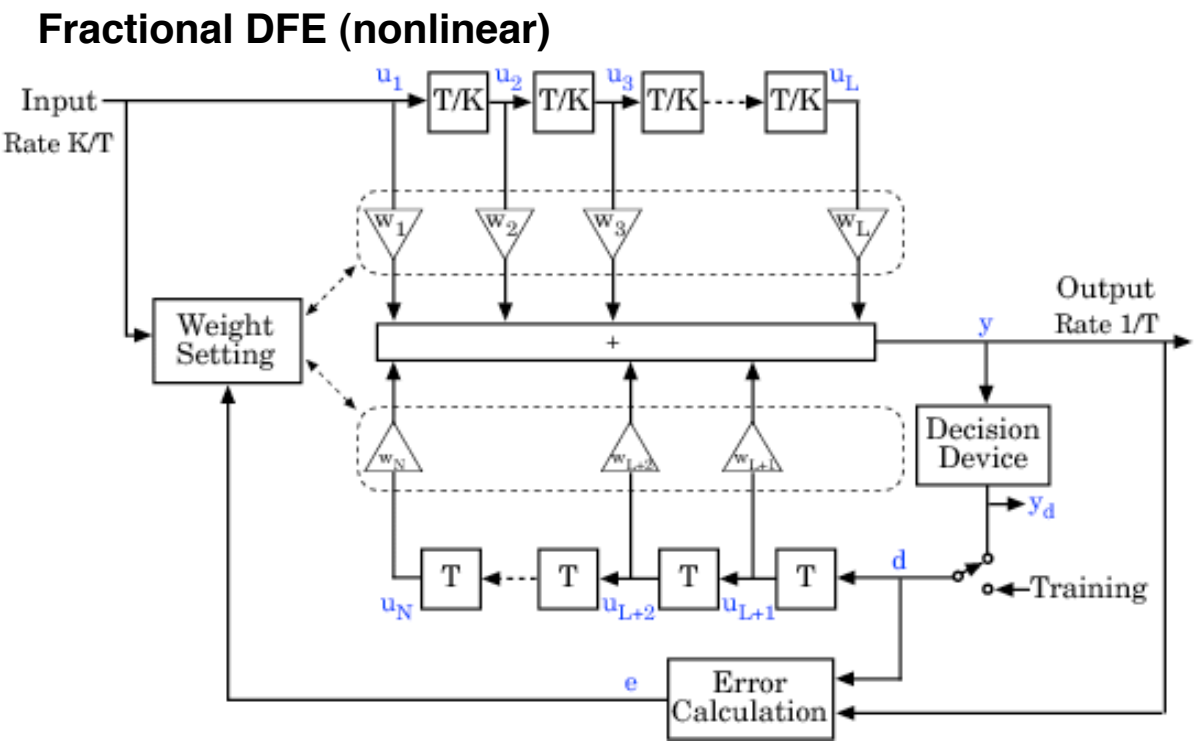
Linear



Fractional Linear



MATLAB linear equalizers, symbol spaced and fractional



MATLAB decision feedback equalizer (DFE)

5.12 The Confidence Interval for $\hat{P}_e$

- In Monte-Carlo simulation you obtain $\hat{P}_e$ as an estimate of the true error probability P_e, via bit error counting and what amounts to the sample mean for a 0/1 random variable[7]

$$\hat{P}_e = \frac{n}{N}$$

where n is the number of bits in error and N is the number of bits processed

[7] M. Jeruchim, "Techniques for Estimating the Bit Error Rate in the Simulation of Digital Communication Systems," IEEE Trans. on Selected Areas in Commun., VOL. SAC-2, NO. 1, January 1984, pp. 153–170.

- For $N \to \infty$ the expectation is that this sample mean becomes the true value P_e

- From statistics, the *confidence interval* is a range of numbers bounded by the endpoints p_1 and p_2, such that we can say $p_2 \leq P_e \leq p_1$ with hopefully high probability (e.g., 0.9, 0.95, or 0.99 are popular values)

- The *confidence level*, $1-\alpha$, is the probability the estimate lies on confidence interval, i.e.,

$$P\left[p_2 \leq P_e \leq p_1\right] = 1-\alpha$$

- Assuming independent bit errors, the error count number n is binomially distributed so $\hat{P}_e = n/N$ is a scaled binomial distribution

- Assuming symmetry, the tail probabilities are each $\alpha/2$

$$\sum_{k=0}^{n} \binom{N}{k} p_1^k (1-p_1)^{N-k} = \alpha/2$$
$$\sum_{k=n}^{N} \binom{N}{k} p_2^k (1-p_2)^{N-k} = \alpha/2$$

  - Solving the two equations yields the confidence interval, but this requires working with the *cumulative beta* distribution

- The typical values of P_e, n, and N are such that a Normal approximation to the binomial distribution is valid

- It can then be shown that the confidence interval is

$$P[y_- \leq P_e \leq y_+] = 1 - \alpha$$

where $d_\alpha = \sqrt{2} \cdot \text{erfcinv}(\alpha)$,

$$y_\pm = \hat{P}_e + \frac{d_\alpha^2}{2} N \hat{P}_e \left(1 \pm \sqrt{\frac{4}{N \hat{P}_e d_\alpha^2}} \right)$$

and we have further assumed that $N/(N + d_\alpha^2) \approx 1$

- Further simplification can be obtained by making the substitution $N \hat{P}_e = n$, the error count value,

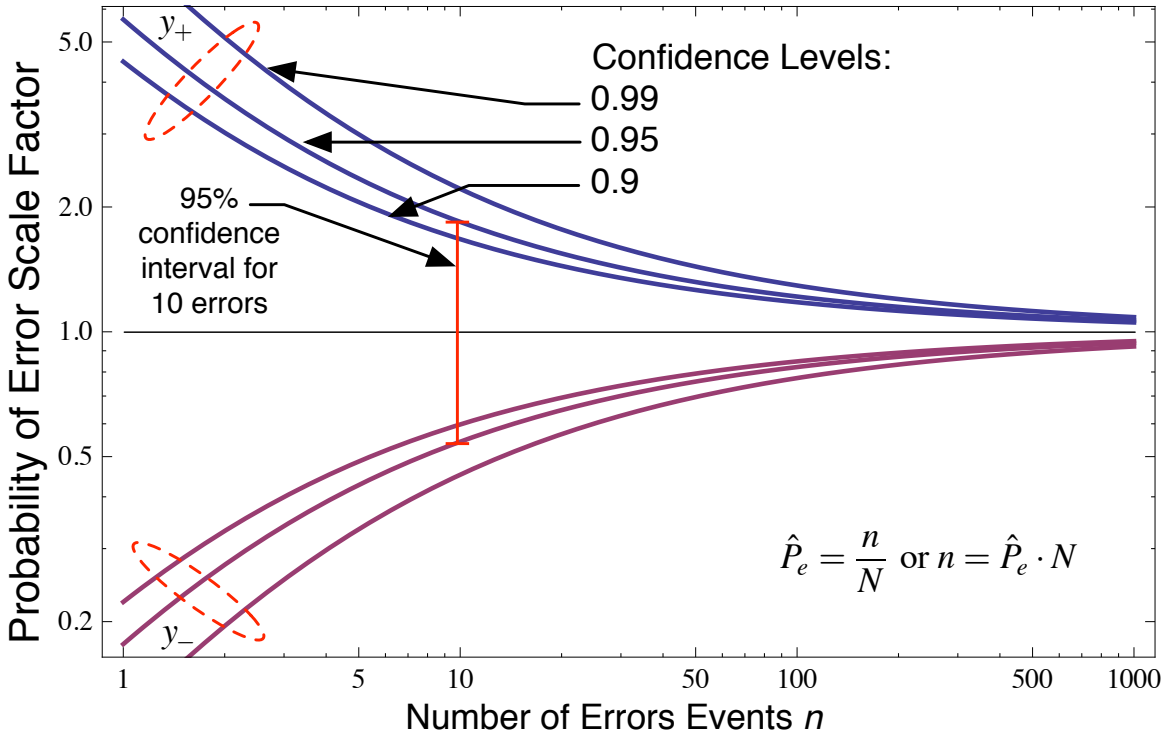
$$y_\pm = \hat{P}_e \left[1 + \frac{d_\alpha^2}{2n} \left(1 \pm \sqrt{\frac{4n}{d_\alpha^2}} \right) \right]$$

$$y_\pm = \frac{n}{N} \left[1 + \frac{d_\alpha^2}{2n} \left(1 \pm \sqrt{\frac{4n}{d_\alpha^2}} \right) \right]$$

$$d_\alpha = \sqrt{2} \cdot \text{erfcinv}(\alpha)$$

- In practice then you can find the confidence interval given the confidence level, observed value n, and the number of trials N

- The first line above makes it clear that the confidence interval endpoints are just a scaled version of $\hat{P}_e$ itself, thus the plot shown below can be constructed, independent of $\hat{P}_e$



$\hat{P}_e$ confidence intervals

```
function [Pe_hat, p2, p1] = Pe_conf_int(n,N,alpha)
% [Pe_hat, p2, p1] = Pe_conf_int(n,N,alpha)
%
% Bit error probabilityestimation Normal approximation to the
% 1 - alpha confidence interval. Typical alpha = 0.05 <=> 0.95 level
%
% Mark Wickert October 2012


d = sqrt(2)*erfcinv(alpha)
yp = (1+d^2./(2*n).*(1+sqrt(4*n/d^2+1)));
ym = (1+d^2./(2*n).*(1-sqrt(4*n/d^2+1)));
Pe_hat = n/N;

p2 = Pe_hat*ym;
p1 = Pe_hat*yp;
```