MIT 6.02 **DRAFT** *Lecture Notes*
Fall 2010 (Last update: September 20, 2010)
Comments, questions or bug reports?
Please contact 6.02-staff@mit.edu

LECTURE 5
Noise and ISI

If there is intersymbol interference (ISI) in a communication channel, then the signal detected at the receiver depends not just on the particular bit being sent by transmitter, but on that bit and its neighbors in time. As you will see below, if there is ISI *and* noise, then determining the probability of a bit error is more complicated than the ISI-free case. We will examine the ISI plus noise case by returning to the example of additive white Gaussian noise, and make use of the **Unit Normal Cumulative Distribution Function**, denoted Φ, as well as **Conditional Probabilities**. Following the analysis of bit error in the face of ISI, we will return to the subject of eliminating ISI using deconvolution. But this time, we will take the view that we are willing to accept imperfect deconvolution in the noise-free case, if the resulting strategy produces reasonable results in the noisy case.

■ 5.1 The Unit Normal Cumulative Distribution Function

The reason we emphasize the Gaussian (or Normal) cumulative distribution function (CDF) is that a wide variety of noise processes are well-described by the Gaussian distribution. Why is this the case? The answer follows from the *central limit theorem* that was mentioned, but not described, in the previous lecture. A rough statement of the central limit theorem is that the CDF for the *sum* of a large number of independent random variables is nearly Gaussian, almost regardless of the CDF's of the individual variables. The *almost* allows us to avoid delving in to some technicalities.

Since noise in communication systems is often the result of the combined effects of many sources of interference, it is not surprising that the central limit theorem should apply. So, noise in communication systems is often, but not always, well-described by a the Gaussian CDF. We will return to this issue in subsequent lectures, but for now we will assume that noise is Gaussian.

The unit Normal probability density function (PDF) is just the Gaussian PDF for the zero-mean ($\mu = 0$), unit standard-deviation ($\sigma = 1$) case. Simplifying the formula from

Lecture 4, the unit Normal PDF is

$$f_X(x) = \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}}, \tag{5.1}$$

and the associated unit Normal cumulative distribution function is

$$\Phi(x) = \int_{-\infty}^{x} f_X(x')\,dx' = \int_{-\infty}^{x} \frac{e^{-\frac{x'^2}{2}}}{\sqrt{2\pi}} dx'. \tag{5.2}$$

There is no closed-form formula for the unit Normal CDF, Φ, but most computer math libraries include a function for its evaluation. This might seem foolish given one is unlikely to be lucky enough to have a noise process with a standard deviation of exactly one. However, there is a simple way to use the unit Normal CDF for any Gaussian random variable. For example, suppose we have a Gaussian zero-mean noise process, $noise[n]$, with standard deviation σ. The probability that $noise[n] < x$ is given by

$$P(noise[n] < x) = \int_{-\infty}^{x} \frac{e^{-\frac{x'^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}} dx' = \Phi\left(\frac{x}{\sigma}\right). \tag{5.3}$$

That is, we can evaluate the CDF for a zero-mean, σ standard-deviation process just by scaling the argument before evaluating Φ. Note, this does **not** imply that one can just scale the argument when evaluating the PDF!

As with any CDF, $\lim_{x\to-\infty} \Phi(x) = 0$ and $\lim_{x\to\infty} \Phi(x) = 1$. In addition, the symmetry of the zero-mean Gaussian PDF, $f_X(x) = f_X(-x)$, implies $\Phi(0) = 0.5$ (half the density in the PDF corresponds to negative values for the random variable). Another identity that follows from the symmetry of the Gaussian PDF, and one we will use subsequently, is

$$\Phi(x) = 1 - \Phi(-x). \tag{5.4}$$

■ 5.2 ISI and BER

Recall from last lecture that if our noise model is additive white Gaussian noise, and if we assume the receiver and transmitter have exactly the same bit period and never drift apart, then

$$y[i + ks] = y_{nf}[i + ks] + noise[i + ks] \tag{5.5}$$

where $i + ks$ is the index of the bit detection sample for the k^{th} transmitted bit, $y[i + ks]$ is the value of the received voltage at that sample, and $y_{nf}[i + ks]$ is what the received voltage would have been in the absence of noise.

If there are *no* ISI effects, then $y_{nf}[i + ks]$ is equal to either the receiver maximum voltage or the receiver minimum voltage, depending on whether a '1' bit or a '0' bit is being received, as shown in the eye diagram in Figure 5-1. For the eye diagram in Figure 5-2, there are three bits of intersymbol interference. And as can be seen in the figure, $y_{nf}[i + ks]$ can take on any of sixteen possible values. The eight upper values are associated with receiving a '1' bit, and the eight lower values are associated with receiving a '0' bit.

In order to determine the probability of a bit error in the case shown in Figure 5-2,

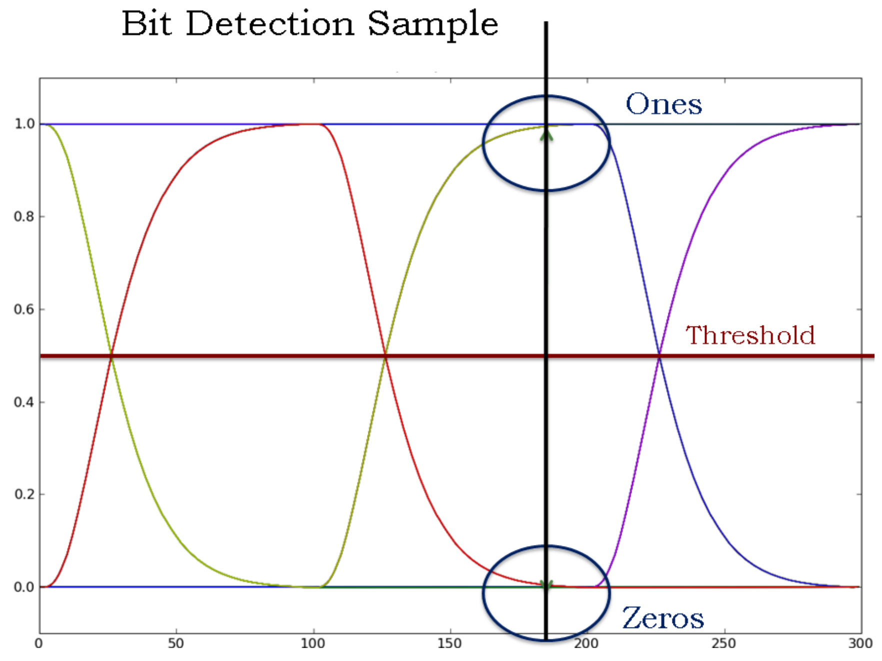

Figure 5-1: A noise-free eye diagrams, showing a bit detection sample with no ISI effects

we must determine the probability of a bit error for each of sixteen cases associated with the sixteen possible values for $y_{nf}[i+ks]$. For example, suppose v_L^j is one of the possible voltage values for the bit detection sample associated with receiving a '0' bit. Then for a digitization threshold voltage v_{th}, the probability of a bit error, *given* $y[i+ks] = v_L^j$,

$$P(y[i+ks] > v_{th}|y_{nf}[i+ks] = v_L^j) = P(noise[i+ks] > (v_{th} - v_L^j)|y_{nf}[i+ks] = v_L^j) \tag{5.6}$$

where we have used the notation $P(a|b)$ to indicate the probability that a is true, given it is known that b is true.

Similarly, suppose v_H^j is one of the possible voltage values for a bit detection sample associated with receiving a '1' bit. Then the probability of a bit error, *given* $y[i+ks] = v_H^j$, is

$$P(noise[i+ks] < (v_{th} - v_H^j)|y_{nf}[i+ks] = v_H^j). \tag{5.7}$$

Comparing (5.7)to (5.6), there is a flip in the direction of the inequality. Note also that $(v_{th} - v_H^j)$ must be negative, or bit errors would occur even in the noise-free case.

Equation (5.6) means that if the transmitter was sending a '0' bit, and if the sequence of transmitted bits surrounding that '0' bit would have produced voltage v_L^j at the receiver (in the absence of noise), then there will be a bit error if the noise is more positive than the distance between the threshold voltage and v_L^j. Similarly, (5.7) means that if the transmitter was sending a '1' bit, and if the sequence of transmitted bits surrounding that '1' bit would have produced voltage v_H^j at the receiver (in the absence of noise), then there will be a bit error if the noise is negative enough to offset how far v_H^j is above the threshold voltage.

If the noise samples are Gaussian random variables with zero mean ($\mu = 0$) and standard deviation σ, the probabilities in (5.6) and (5.7) can be expressed using the unit normal

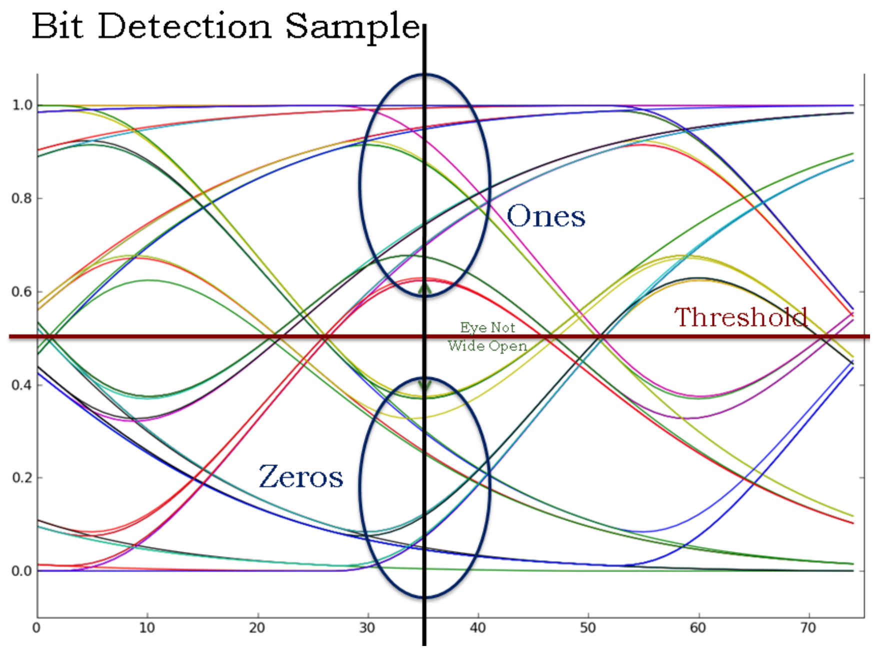

Figure 5-2: A noise-free eye diagram showing a bit detection sample with three bits of ISI

CDF. Specifically,

$$P(noise[i+ks] > (v_{th} - v_L^j) | y_{nf}[i+ks] = v_L^j) = 1 - \Phi\left(\frac{v_{th} - v_L^j}{\sigma}\right) = \Phi\left(\frac{v_L^j - v_{th}}{\sigma}\right) \quad (5.8)$$

and

$$P(noise[i+ks] < (v_{th} - v_H^j) | y_{nf}[i+ks] = v_H^j) = \Phi\left(\frac{v_{th} - v_H^j}{\sigma}\right) \quad (5.9)$$

where the right-most equality in (5.8) follows from (5.4).

If all bit sequences are equally likely, then each of the possible voltage values for the bit detection sample is equally likely[1]. Therefore, the probability of a bit error is just the sum of the conditional probabilities associated with each of the possible voltage values for a bit detection sample, divided by the number of possible values. More specifically, if there are J_L voltage values for the bit detection sample associated with a transmitted '0' bit and J_H voltage values associated with a transmitted '1' bit, and all voltage values are equally likely, then the probability of a bit error is given by

$$P(bit \;\; error) = \frac{1}{J_H + J_L}\left(\sum_{j=1}^{j=J_L} \Phi\left(\frac{v_L^j - v_{th}}{\sigma}\right) + \sum_{j=1}^{j=J_H} \Phi\left(\frac{v_{th} - v_H^j}{\sigma}\right)\right). \quad (5.10)$$

A number of popular bit encoding schemes, including the 8b/10b encoding scheme described in the first lab, reduce the probability of certain transmitter bit patterns. In cases

[1]the degenerate case, where multiple bit pattern result in identical voltage values for the bit detection sample, can be treated without altering the following analysis, as is shown in the worked example companion to this text

like these, it may be preferable to track transmitter bit sequences rather than voltage values of the bit detection sample. A general notation for tracking the error probabilities as a function of transmitter bit sequence is quite unwieldy, so we will consider a simple case. Suppose the ISI is such that only the previous bit interferes with the current bit. In this limited ISI case, the received bit detection sample can take on one of only four possible values. Let v_L^1 denote the value associated with transmitting two '0' bits in a row (bit sequence 00), v_L^2, the value associated with transmitting a '1' bit just before a '0' bit (bit sequence 10), v_H^1, the value associated with transmitting a '0' bit just before a '1' bit (bit sequence 01), and v_H^2, the value associated with transmitting two '1' bits in a row (bit sequence 00).

See, even in this simple case, the notation is already getting awkward. We use $P_{(00)1}$ to denote the probability that the receiver erroneously detected a '1' bit *given* a transmitted bit sequence of 00, $P_{(01)0}$ to denote the probability that the receiver erroneously detected a '0' bit *given* a transmitted bit sequence of 01, and so forth. From (5.6) and (5.7),

$$\begin{aligned} P_{(00)1} &= P(noise[i+ks] > (v_{th} - v_L^1)|y_{nf}[i+ks] = v_L^1), & (5.11)\\ P_{(10)1} &= P(noise[i+ks] > (v_{th} - v_L^2)|y_{nf}[i+ks] = v_L^2), & (5.12)\\ P_{(01)0} &= P(noise[i+ks] < (v_{th} - v_H^1)|y_{nf}[i+ks] = v_H^1), & (5.13)\\ P_{(11)0} &= P(noise[i+ks] < (v_{th} - v_H^2)|y_{nf}[i+ks] = v_H^2). & (5.14) \end{aligned}$$

If we denote the probability of transmitting the bit sequence 00 as P_{00}, the probability of transmitting the bit sequence 01 as P_{01}, and so forth, then the probability of a bit error is given by

$$P(bit\ \ error) = P_{(00)1}P_{00} + P_{(10)1}P_{10} + P_{(01)0}P_{10} + P_{(11)0}P_{11}. \tag{5.15}$$

Whew! Too much notation.

■ 5.3 Deconvolution and noise

Many problems in inference and estimation have deconvolution as the noise-free optimal solution, and finding methods for succeeding with deconvolution in the presence of noise arises in a variety applications including: communication systems, medical imaging, light microscopy and telescopy, and audio and image restoration. The subject is enormous, and we will touch on a very small part of it, focussing on a few techniques that will help us improve the performance of the IR channel.

When we use deconvolution, we are trying to eliminate the effects of an LTI channel that lead to intersymbol interference. More generally though, deconvolution is just one of many strategies for estimating the transmitted samples from noisy received samples. With additive noise and deconvolution, the diagram for transmission though our channel is now

$$X \rightarrow CHANNEL \rightarrow Y_{nf} \rightarrow Add\ NOISE \rightarrow Y \rightarrow DECONVOLVER \rightarrow W \approx X \tag{5.16}$$

where X is the sequence of transmitted samples, Y_{nf} is the sequence of noise-free samples produced by the channel, Y is the sequence of receiver-accessible noisy samples, and the sequence W is the estimate of X generated by the deconvolver.

If the LTI channel has a unit sample response, H, and that H is used to deconvolve the noisy received samples, W satisfies the deconvolving difference equation

$$\sum_{m=0}^{m=n} h[m]w[n-m] = y_{nf}[n] + noise[n], \tag{5.17}$$

where the right-hand side of (5.17), in this case $y_{nf}[n] + noise[n]$, is referred to as the input to the deconvolving difference equation. Also, as the channel is LTI, X is related to Y_{nf} through convolution with the channel's unit sample response,

$$y_{nf}[n] = \sum_{m=0}^{m=n} h[m]x[n-m]. \tag{5.18}$$

If there is no noise, $W = X$, and we have perfect deconvolution, though we already know that for some H's deconvolution can be very sensitive to noise. Since we cannot completely eliminate noise, and we cannot change H, our plan will be to approximate H by $\tilde{H}$ when deconvolving, and then try to design $\tilde{H}$ so that this approximate deconvolution will be less sensitive to noise, yet still provide good estimates of the input sequence X. As we analyze approximate deconvolution in more detail, we will discover that the design of $\tilde{H}$ involves a trade-off. The trade off is between reducing noise sensitivity, which we will relate to the **stability** properties of the deconvolving difference equation, and how close $\tilde{H}$ is to H, which we will relate to the **accuracy** of the approximate deconvolver in the absence of noise.

■ 5.3.1 Convolution Abstraction

Manipulating convolution sums can be cumbersome, but more importantly, the complexity of the details can obscure larger issues. In mathematics, engineering, science, or for that matter life in general, the key to managing complexity is to find the right abstraction.

In our case, a useful abstraction is to represent convolution using the notation

$$H * X \equiv \sum_{m=0}^{m=n} h[m]x[n-m], \tag{5.19}$$

where H is a unit sample response and X is an input sequence.

That convolution is an LTI operation leads to two identities that can be stated quite compactly using our abstract notation. First, given two sequences, X and W, and a unit sample response, H,

$$\begin{aligned} H * W - H * X &\equiv \sum_{m=0}^{m=n} h[m]w[n-m] - \sum_{m=0}^{m=n} h[m]x[n-m] \qquad (5.20) \\ &= \sum_{m=0}^{m=n} h[m](w[n-m] - x[n-m]) = H * (W - X). \end{aligned}$$

For the second identity, consider a second unit sample response, $\tilde{H}$,

$$\begin{aligned} H * X - \tilde{H} * X &\equiv \sum_{m=0}^{m=n} h[m]x[n-m] - \sum_{m=0}^{m=n} \tilde{h}[m]x[n-m] \\ &= \sum_{m=0}^{m=n} (h[m] - \tilde{h}[m])x[n-m] = (H - \tilde{H}) * X. \end{aligned} \tag{5.21}$$

The first identity in (5.21) follows directly from the linearity of convolution, and second identity can be derived from the first using the commutative property of convolution ($H * X = X * H$).

■ 5.3.2 Deconvolution Noise Sensitivity

Armed with the notational abstraction and the two identities above, we can get a much clearer picture of exact deconvolution and approximate deconvolution in the presence of additive noise. To begin, note that noise-free exact deconvolution is, in our more abstract notation,

$$\begin{aligned} Y_{nf} &= H * X \\ H * W &= Y_{nf}, \end{aligned} \tag{5.22}$$

Note that in this noise free case, $W = X$.

Including a sequence of additive noise, $NOISE$, before deconvolving yeilds

$$\begin{aligned} Y_{nf} &= H * X \\ Y &= Y_{nf} + NOISE \\ H * W &= Y, \end{aligned} \tag{5.23}$$

where Y is the sequence of received noisy samples. Note that in this noisy case, $W \approx X$, *but only if deconvolution with H is insensitive to noise*. As we have noted before, deconvolution with typical H's can be extremely sensitive to noise, though now we will be more precise about what we mean by noise sensititivy.

Collapsing the three equations in (5.23),

$$H * W = NOISE + H * X, \tag{5.24}$$

or

$$H * W - H * X = NOISE, \tag{5.25}$$

or, by using one of the identities from the previous section,

$$H * (W - X) = H * E = NOISE, \tag{5.26}$$

where $E \equiv W - X$ is the sequence of estimation errors, and E indicates how different our deconvolved W is from the transmitted X.

We can now state what we mean by noise sensitivity.

Deconvolving with H is sensitive to noise if E can be very large even when $NOISE$ is small.

To understand what unit sample responses(USRs) lead to noise-sensitive deconvolvers, consider the case of a finite length USR, $h[n] = 0, n > K$. Then from (5.26), E satisfies the difference equation

$$h[0]e[n] + h[1]e[n-1] + ... + h[K]e[n-K] = noise[n]. \tag{5.27}$$

Now suppose the input to the difference equation, $noise[n]$, is zero for $n > N$. If the difference equation in (5.27) is **unstable**, then the $\lim_{n\to\infty} |e[n]|$ will typically approach ∞. Clearly, if (5.27) is unstable, then for large values of n, $w[n]$ will be a terrible estimate of $x[n]$.

It is possible to determine precisely the stability of the difference equation in (5.27) by examining the roots of a $K + 1^{th}$-order polynomial whose coefficients are given by $h[0], h[1], ..., h[K]$ (For details, see, for example, the 6.01 notes). Unfortunately, the root condition offers little intuition about what kinds of H's will result in deconvolvers that are very sensitive to noise.

In order to develop some intuition, consider reorganizing (5.27) in a form to apply plug and chug,

$$e[n] = -\sum_{m=1}^{m=K} \frac{h[m]}{h[0]} e[n-m] + \frac{1}{h[0]} noise[n]. \tag{5.28}$$

When the deconvolving difference equation is viewed in this plug-and-chug form, one can easily see that small values of $h[0]$ will be a problem. Not only is the input noise sample magnified by $\frac{1}{h[0]}$, but if in addition $|\frac{h[m]}{h[0]}| > 1$ for any $m > 0$, then errors from previous steps are likely to accumulate and cause $e[n]$ to grow rapidly with n.

One can be somewhat more precise by introducing a *sufficient*, but not necessary, condition for stability. If

$$\sum_{m=1}^{m=K} |\frac{h[m]}{h[0]}| = \Gamma < 1 \tag{5.29}$$

then (5.27) is a stable difference equation. The justification for this perhaps unsurprising result is mercifully brief, and can be found in the appendix.

It is important to note that (5.29) is a *sufficient* condition for stability of the deconvolving difference equation, and quite a conservative condition at that. It is easy to generate very stable difference equations that violate (5.29). What we should retain is the insight that *making* $h[0]$ *larger is likely to improve stability.*

■ 5.3.3 Analyzing Approximate Deconvolution

In order to reduce noise sensitivity, so that $W \approx X$ even if Y is noisy, we will consider performing the deconvolution using a different unit sample response, $\tilde{H}$. In this case,

$$\begin{aligned} Y_{nf} &= H * X \qquad (5.30) \\ Y &= Y_{nf} + NOISE \\ \tilde{H} * W &= Y, \end{aligned}$$

where we will try to design $\tilde{H}$ so that $W \approx X$ even when Y is noisy.

Collapsing the three equations in (5.31),

$$\tilde{H} * W = NOISE + H * X, \tag{5.31}$$

and therefore

$$\tilde{H} * W - H * X = NOISE. \tag{5.32}$$

Deriving an equation for the error, $E \equiv W - X$, in this approximate deconvolver case requires a small trick. If we add $\tilde{H} * X - \tilde{H} * X = 0$ to (5.32),

$$\tilde{H} * W + (\tilde{H} * X - \tilde{H} * X) - H * X = \tilde{H} * (W - X) - (\tilde{H} - H) * X = NOISE, \tag{5.33}$$

and we can now rewrite (5.32) in terms of the error,

$$\tilde{H} * E = NOISE + (\tilde{H} - H) * X, \tag{5.34}$$

or in sum form

$$\sum_{m=0}^{m=n} \tilde{h}[m]e[n-m] = noise[n] + \sum_{m=0}^{m=n} (\tilde{h}[m] - h[m])x[n-m] \tag{5.35}$$

where the right-hand side of (5.34) or (5.35) is the *input* to the approximate deconvolver difference equation for the error.

If we compare (5.34) to (5.26), we note two differences. First, the deconvolving difference equations, the left-hand sides, are different. To make the approximate deconvolver less sensitive to noise, we should pick $\tilde{H}$ so that the approximate deconvolving difference equation is as stable as possible.

The second difference between (5.34) to (5.26) is that the input in (5.34) has an extra term,

$$(\tilde{H} - H) * X. \tag{5.36}$$

If there is no noise, the input to (5.26) will be zero, and therefore E will be zero (perfect deconvolution). But even if there is no noise, there will be a nonzero input in (5.34) equal to (5.36). If (5.36) is small, the input to the approximate deconvolving difference equation in (5.34) will be small. If we assume that $\tilde{H}$ was properly designed, and its associated deconvolving difference equation that is quite stable, then if (5.36) is small, the associated E will be small.

So, we have two competing concerns. We want to pick $\tilde{H}$ so that the deconvolving difference equation based on $\tilde{H}$ will be as **stable** as possible, yet we also want to ensure that $(\tilde{H} - H) * X$ is as small as possible so the deconvolver will be **accurate**. Or said another way, if H leads to deconvolution that is sensitive to noise, then the H-generated deconvolving difference equation must be unstable or nearly unstable. if $\tilde{H}$ is too close to H, then the deconvolving difference equation generated from $\tilde{H}$ will have the same bad stability properties. But, if $\tilde{H}$ is too far from H, then the approximate deconvolver will be inaccurate.

5.3.4 Summarizing

After all the above analysis, we know we want a stable yet accurate deconvolver, but we still do not have an explicit formula, or even a general receipe, for finding a good $\tilde{H}$. What we do have is some guidance about what to try.

First, we have good evidence that the bigger we make the value of $|\tilde{h}[0]|$ compared to $|\tilde{h}[m]|, m > 0$, the more stable we make the approximate deconvolving difference equation, thereby enhancing noise immunity.

Second, we know that to improve accuracy, we must make $(\tilde{H} - H) * X$ as small as possible. Since X is the sequence of transmitter samples, and is representing bits, the unit step might be a reasonable typical X to consider (certainly important if the transmitter is sending many '1' bits in a row). If X is the unit step, then $(\tilde{H} - H) * X = \tilde{H} * X - H * X = \tilde{S} - S$ where S and $\tilde{S}$ are the step responses associated with the unit sample responses H and $\tilde{H}$, respectively. This suggests that a good $\tilde{H}$ should have an associated step response that matches the channel step response as well as possible. That is, try to make

$$(\tilde{s}[n] - s[n]) = (\sum_{0}^{n} \tilde{h}[n] - \sum_{0}^{n} h[n]) \tag{5.37}$$

as close to zero as possible for as many values of n as possible, while still ensuring stability.

Our analysis suggested some heuristics for generating an effective deconvolving $\tilde{H}$, though we gained no insight in to how close we came to finding the best $\tilde{H}$. So, one might ask, is there a way to determine the optimal $\tilde{H}$? To answer that question, one first needs a precise definition of optimal. For example, in this setting, an optimal deconvolving $\tilde{H}$ might be defined as the one that results in the lowest BER. And then, after one has decided on a definition of optimality, one needs to determine if the reaulting optimization problem is tractable. To address these issues requires quite a few more mathematical tools, a few of which we will see in later lectures, but most will require study beyond 6.02. But, what we hope you will see in lab, is that there is a lot you can do with the tools you have now.

5.4 Appendix

5.4.1 Justification of the Sufficient Condition for Stability.

Given a deconvolving difference equation,

$$h[0]e[n] + h[1]e[n-1] + ... + h[K]e[n-K] = noise[n], \tag{5.38}$$

if the difference equation coefficients satisfy

$$\sum_{m=1}^{m=K} |\frac{h[m]}{h[0]}| = \Gamma < 1, \tag{5.39}$$

then (5.38) is a stable difference equation.

We will show that (5.39) implies that

$$\lim_{n\to\infty} e[n] \to 0 \tag{5.40}$$

if *noise*$[n] = 0$ for all n greater than some finite N. In this difference equation setting, stability is equivalent to saying that the deconvolver error would eventually decay to zero if the noise suddenly became zero.

To show that $e[n] \to 0$, we first reorganize (5.38) in to plug-and-chug form for the $n > N$ case,

$$e[n] = -\sum_{m=1}^{m=K} \frac{h[m]}{h[0]} e[n-m]. \tag{5.41}$$

Taking absolute values yields the inequality

$$|e[n]| \leq \sum_{m=1}^{m=K} |\frac{h[m]}{h[0]}||e[n-m]|. \tag{5.42}$$

To simplify the steps of the proof, we define a sequence whose n^{th} value is the maximum absolute value of the error over the last K samples,

$$\Upsilon[n] \equiv \max_{n-K<m\leq n} |e[m]|. \tag{5.43}$$

From (5.42) and the fact that, by definition, $|e[n-m]| \leq \Upsilon[n-1]$ for $1 \leq m \leq K$,

$$|e[n]| \leq \left(\sum_{m=1}^{m=K} \left| \frac{h[m]}{h[0]} \right| \right) \Upsilon[n-1]. \tag{5.44}$$

Equation (5.44) can be simplified using (5.39),

$$|e[n]| \leq \Gamma \ \Upsilon[n-1], \tag{5.45}$$

which implies that $\Upsilon[n] \leq \Upsilon[n-1]$ as $\Gamma < 1$. Then by iterating the above result,

$$\begin{aligned} |e[n+1]| &\leq \Gamma \ \Upsilon[n] \leq \Gamma \ \Upsilon[n-1] \qquad (5.46) \\ |e[n+2]| &\leq \Gamma \ \Upsilon[n+1] \leq \Gamma \ \Upsilon[n-1] \\ &\vdots \\ |e[n+K-1]| &\leq \Gamma \ \Upsilon[n+K-2] \leq \Gamma \ \Upsilon[n-1], \end{aligned}$$

which together with (5.45) implies

$$\Upsilon[n+K-1] \leq \Gamma \ \Upsilon[n-1]. \tag{5.47}$$

since, by definition, $\Upsilon[n+K-1] \equiv \max_{n-1<m\leq n+K-1} |e[m]|$. From (5.47) it follows that

$$\Upsilon[n+lK-1] \leq \Gamma^l \Upsilon[n-1]. \tag{5.48}$$

Since $\Gamma < 1$, $lim_{l\to\infty}\Gamma^l = 0$ and therefore $\lim_{l\to\infty} \Upsilon[n+lK-1] \to 0$, from which it follows that $\lim_{n\to\infty} e[n] \to 0$.