MITOCW | 8. Noise

PROFESSOR: Today we're going to dig a little deeper into the system that we've been talking about. So we've already talked about source coding and source decoding. And then we talked about channel coding as we've just finished talking about block codes and Viterbi-- convolutional codes and Viterbi decoding. So that's the coding here and the decoding.

And now we're going to drill down to the next level to start to talk about the actual signals going across physical channels. So this is going to actually extend over the entire next module of the course.

I want to describe this in the context of something you're going to be doing in labs 4 through 6. You're actually going to experiment with a specific channel. What you'll have is bits coming in, code words coming in, being translated to signals. In this case, discrete time signals, and I'll give you an example shortly of that. The signals will then be adapted through the modulator for transmission on an analog channel. So there's a modulation process and there's a digital-to-analog conversion process.

You'll be generating waveform that you apply to the speaker in your laptop. That's going to be a transmitter. The channel is going to be just the air around you with all the disturbances of room acoustics and noise and all of that, all the distortions from that. And then you'll pick up the signal on the microphone on your laptop or an external microphone if you want.

Conversion from analog to digital, demodulation and filtering to undo the modulation, and we'll be talking about this in more detail to get another sequence of samples. After which you have a decision rule that then looks at the samples and says, did I get a 0 or a 1? And you spit out the bits of your code word. OK, so this is what we're going to be looking at.

So here is what you might be sending at the transmitting end. You've got the bits coming in. You're going to convert them to signals, and we're going to think of discrete time signals. So this is a signal x of n-- n takes integer value, so that's my discrete time clock. And the typical waveform might look like this. I might decide just very simply to have levels held at 0.5 for, let's say, 16 samples per bit, and then held at 0 for 16 samples to denote a 1 and a 0 respectively.

So here's a 1, a 00, 111, 0101. So we're converting two samples. This is a sample number, and then the next step will be to actually-- in your computer you'll send this to your digital to analog converter which will-- with a particular clock cycle, convert this to real time. What you might imagine is that the actual waveform that goes out on the channel is somehow related to the continuous waveform that you get by just connecting the tops of these discrete time values.

The actual mechanism for transmission through the air we'll talk about next time. So right now we're just going to focus on the level of the discrete time signals. And at the other end, after you've done your transmissions through the channel and you've demodulated and filtered, you get a sequence which ideally is a replication of the sequence that you sent in.

It can't have a scale factor, scale factors don't worry us. In this case, you see that the amplitude is divided by 2. But basically you see the trace of what was sent at the transmitting end. There's some distortion that's introduced by the dynamics of the channel, and we'll be talking about that in more detail later. So we aren't getting quite the straight edges. But after a brief transient period, the waveform seems to settle to the constant value that we had of the input. So this is our received set of samples.

Now in this figure, I've assumed that there's no noise, only the distortion. This lecture is going to be about the noise. I wanted you to get the sense of what distortion does, and then we'll park that issue and come back to it next time and actually for several lectures after that. But this lecture we're going to focus on noise. Before we look at noise, this is what a noise-free received signal might look like with just the distortion in it.

OK. And now you've got to convert to a bit sequence. So a simple way to do that is pick an appropriate point in each bit slot. Each slot of 16 samples long. Pick an appropriate point, taking account of these transient effects and so on, and then sample. And if the sample value is above a threshold, you'll declare a 1. If the sample values below the threshold, you'll declare a 0. And so you reconstruct the sequence that went in.

So we have the sample and threshold feature here. So we're just taking one of the samples in the bit period, comparing with the threshold, and making a declaration. That's a very simple-minded decision rule. OK. So we'll come back to distortion. Today I want to talk about noise, and I want to then suppress distortion. So let's forget about distortion. Let's assume that the received signal yn is exactly what was sent except for some additive noise.

So what we're imagining is you send a nice clean set of samples here into your digital-to-analog converter, and what comes out ideally would be the same set of samples, but actually what happens is that each of these samples is perturbed by noise. And so you get something that might look like this. OK, so this is y, then, and what we had before was x of n.

OK. So nominally you'd get the same thing. The only thing that's different now is you've got an additive noise. We're going to assume that this noise sample wn is independent from one sample to the next. So when the channel and the processing and so on decides to put a noise sample on this, it doesn't pay attention to what noise sample was out of on either side. So every noise sample is picked independently. And it's picked from the same distribution. That's with the identically distributed part of this mean.

So the characteristics of the noise are the same right through our signal. That's what we're assuming. That's the identically distributed part. It's a statement about the stationarity of the noise characteristics. All of this can be generalized, but this is where we're going to have our story and that's all we're going to consider.

OK, a key metric, then, is what's the signal-to-noise ratio? This is something that you see all over the place, the SNR. Usually what people mean is signal power, and power is usually the square of a signal-- that's what you're thinking of. If you think of voltages, for instance, the square of the voltage gives you power in the resistor. So you think of the signal as being x, Its power as being x squared. Except you've got to decide, do you want to talk about the peak power or the time average power or some other measurement of the signal power?

So that's the signal part of this ratio. And then the noise part of the ratio is the noise variance. So we have a noise component wn, it's the expected squared amplitude of that. Oh, by the way, I didn't-- this is on my slide, but I didn't say it yet. I'm going to assume the noise is zero mean. Which means that these excursions from what you expect on average are at 0.

If there was a systematic bias to the noise, if I knew that there was a non-zero mean, I could just factor that into my processing and think of my expected received signal as taking account of that non-zero mean. So there's no loss of generality, really. I'm assuming a zero mean noise. OK.

Now when you come to actually computing numbers, this is another example-- showing another kind of waveform, this is the sum of sinusoids, I assume, to which you're adding some noise. And in this particular simulation, by tweaking the value of A there, that's the-- it's a gain factor on the signal. You can actually vary the signal-to-noise ratio and get a feel for what difference signal-to-noise ratio is represented.

So at high signal-to-noise ratio, the noise isn't perturbing what went down very much. But when you get the lower signal-to-noise ratios, the noise is actually distorting the signal that you started with quite substantially. Now the SNR here is described in dB, decibels. And so let me just say a word about that. That's a unit you'll see all the time. You've seen all the time.

So we're really trying to measure a signal-to-noise ratio. So this is what you would normally think of. But in many applications, a logarithmic scale is really what you want to deal with. For instance, if you're measuring the response of the ear to noise intensities, it turns out there's a logarithmic feature built into our sensors. So usually want to be measuring power and power ratios in terms of a log scale. That should have had a capital B there.

So here's the definition of what a ratio is on dB. It's the ratio log to the base 10 times 10. One caution here. I told you that when we talk about powers, that's the square of the amplitude. So if you're going to compare amplitudes, ratio of amplitudes on a log scale, then actually what you end up doing is taking 20 log 10 ratio of amplitudes.

So you'll sometimes see this definition as 20 log to the base 10 ratio of amplitudes, and what people are doing, then, is comparing amplitude ratios, not power ratios. You have a question?

AUDIENCE: Why do we define power as amplitude squared?

PROFESSOR: In sum-- so the question was, why do we define power as amplitude squared? If you think of an electrical circuit with some signal applied across it, a voltage, the instantaneous power dissipated in the resistor is given by that. So people start to think of square of a quantity as power. In the continuous time domain that's very natural in signals that come from physics, and that terminology is just being carried over to this kind of a discrete time setting. So when people say power, they mean square of the signal. It could've been called something else.

OK. So you can actually span huge ratios in power on this log scale with much more better behaved numbers. 0 dB, then, is a ratio of 1. 3 dB, this is good to carry around in your head. 3 dB, it's actually 3.01-something, but 3 dB is a factor of 2 on the power ratio, or square root of 2 on an amplitude ratio.

So let's actually go back to what I showed you on the previous slide. So here, for instance, is an SNR of 0.4 dB. If I figure that that's close to 0 dB, then I should expect that the noise power and signal power are about equal, and the noise amplitude and signal amplitude are about equal.

So what I expect to see is perturbations of the original signal that are comparable with the signal values themselves, and that's sort of what we see here. The shape of the signal is pretty distorted at this point because the typical amplitude of the noise sample is comparable with the signal sample that I'm interested in.

OK, so when you get to 0 dB, you're starting to get quite disturbed-looking waveforms. When you have 20 dB in power, that's actually 100-- ratio of 100-- sorry, what is that? Yeah, that's a ratio of 100, isn't it? On par? So it's a ratio of 10 on amplitudes, and that's what you're seeing. The noise excursions are about a 10th of what the signal amplitudes are. All right. It takes a little getting used to, but it's fairly standard.

OK. So now we want to figure out how to describe noise and work with it. So let's look at a typical run of a noise sequence. What I've done is just extracted the noise piece of a typical received signal. So it's got excursions above and below 0. Remember, I said it was a zero mean random variable that we're thinking of, zero mean noise.

And you can describe how these values are distributed by just doing a simple histogram. And if you only take a few values like 100 samples, you get a pretty messy-looking histogram, it doesn't seem to have much structure. But as you take more and more samples, you'll typically find that the histogram actually settles out to a nice shape, to some subtle kind of shape.

Normalizing this to have unit area under it gives you what's called the probability density function for the noise. So this is a term-- kind of notion that's critical in working with noise. So here's a step of idealization. We're stepping back from thinking about histograms to just a mathematical way of talking about how random quantities distribute themselves.

So we'll talk about-- by the way, we've been using W for the noise and X for the signal, but if you look in probability books, the first variable that people-- the first symbol people reach for and they want to talk about a random variable is X, and I got stuck with a whole bunch of figures that had X in them, so I didn't want to change it to W. This is anything. We're going to apply it to our W, but for now it's some capital X. The other convention when you talk about random variables as you tend to use a capital letter to denote the random variable.

OK. So we say that X is a random variable governed by a particular probability density function. If you can compute the probability that X lies in some particular interval by taking the corresponding area under that PDF. So the PDF is the object that gives you probabilities from areas under the integrals.

So if you want the probability that the quantity X, take the numerical values in this range, X1 to X2, then you integrate the PDF from X1 to X2, and this area is what you call-- that area is the probability. And the total area under the PDF, of course, has to be 1 because the probability that X lies somewhere is 1. The probability that X takes some value is 1.

So this is how we work with PDFs. Again, you'll find when people want to sketch a PDF, the reflex is to sketch one of these bell-shaped things. And it turns out there's actually a reason for that. This bell-shaped thing or a specific bell-shaped thing called the Gaussian tends to arise in all sorts of applications, and that's a consequence of something called the central limit theorem.

This is considered one of the most important results in probability theory. It actually dates back to about the 1730s as a conjecture, but it was Laplace who-- in I guess the late 1700s, early 1800s who actually proved it. And it wasn't actually called the central limit theorem until much more recently, till about 1930 or so. And was called that because it was the limit theorem that was central to all of probability, that was the thinking.

So here is the central limit theorem. It says that if you sum up a whole bunch of little random quantities that are not necessarily Gaussian, and if they each have finite mean and finite variance, the sum is going to have a distribution that's going to look increasingly Gaussian. So you could start, for instance, with a random variable that's described by this triangular PDF.

Take a whole bunch of random variables generated according to that PDF. When I say generated according to that PDF, what I mean is that the probability that you get a value between any two limits here is the area under that piece of the triangle. Generate a whole bunch of these and sum them together, you find that the resulting histogram starts to look Gaussian.

You can start with another kind of distribution, and again, it starts to look Gaussian. And the more of these you add, the more it looks Gaussian. And so this can be actually made very precise. There's a very precise sense in which the limiting distribution in a situation like this is a Gaussian. So what is a Gaussian? I've got to describe that for you. I'll do it in more detail in a second.

First, let me tell you how we defined these two key parameters. These are things that from other sorts of contexts. The mean and the standard deviation of the variance, you know it from quiz scores at least, but here is the mathematical definition in terms of a PDF. So if you have a PDF for a random variable capital X, the mean value of capital X is-- it's basically the average value of X weighted by the probability, which is what you expect.

So it's X times the PDF integrated over all possible values. That's the definition of the expected value. And what we do when we take the expected value of the mean value on a quiz is a sort of discrete time version of this. So we're seeing how many people in a particular bin and multiply by the score for the people in that bin and sum over all possible bins. That's one way to think of what this is doing, assuming you've got the right normalization of the PDF.

And the variance is the expected squared deviation from the mean value. So here's a deviation from the mean value. You square it, and now you want to take its expected value, so you weight it by the PDF of X and that gives you the variance. So the variance is the expected squared deviation from the mean value. OK, so the PDF is valuable in getting all of this.

And to get a sense of what means and standard deviations and variances do-- I don't know if I said from the previous slide, by the way, that standard deviation is the square root of the variance. Did I say that? Maybe not. But I have it at the bottom of the slide, right? OK. OK.

So shifting the mean of a random variable, if I define a new random variable with the same PPF except for a different mean, what that means is that-- what that signifies is that the PDF has just shifted over by that amount. So changing the mean and nothing else will just shift the PDF over to the corresponding position.

Changing the variance from a small value to a large value will spread out the PDF because you're the variance is capturing the expected squared deviation from the mean. So a higher variance PDF has got to have a larger spread. But because the areas normalized to 1, if it spreads out this way, it's got to come down on top, and that's what you're seeing here. All these pictures actually turn out to be drawn for the Gaussian, but my statements are more general here.

But here's the Gaussian itself. So now I'm going back to my notation W. We're going to think of a random variable W which is going to be typical of all my noise samples. It's going to have some mean which we'll be taking to be 0 and our examples. It's got a variance sigma squared. So if a random variable has this particular PDF, we call it Gaussian. That's the definition of a Gaussian random variable.

The number here, while you've got to remember it at some point, but all it's doing is normalizing to unit area. So the key thing about a Gaussian is that it's an exponential with a negative sign there of the squared deviation from the mean normalized by the variance with that extra factor 2 there.

So different choices of variance will give you different shapes here. So the smaller variances correspond to the more peaked and more sharply-falling PDFs. So let's see. How many standard deviations away from the mean you have to go before you have very low probability of reaching there? Anyone? There's no unique answer to this, but yeah?

AUDIENCE: 3?

PROFESSOR: 3 is not about idea. So let's see. Let's take sigma squared equals 1. That's variance of 1, so the standard deviation is 1. So for the red trace, by the time we get out to the number 3, we expect to actually see a very low value for the PDF. So 3 sounds about right. Does that hold up for the blue one? Sigma squared is 0.25. So the square root of that is a standard deviation, which is 0.5, so 3 times that. So when we get out to about 1.5, we should be essentially at 0. So don't forget the square root.

The other thing-- actually, I should have commented on this earlier, let me show it to you-- on this slide that I had, I labeled this arrow here just schematically to show you that it's a measure of width. But the tag I put on it is standard deviation. Standard deviation is the thing that you want to use when you want to measure width on a distribution. That has the right units.

Standard deviation, the square root of variance has the same units as X. If X is a voltage, the standard deviation is units of voltage. It would be a mistake to label a spread here by the variance. You want to think in terms of standard deviation when you're thinking about spread. So you define the variance and then take the square root to get the standard deviation. OK.

So for our noise in this kind of setting, in our communications setting, we're going to assume that every noise sample was drawn from a Gaussian distribution with zero mean. Just the same kind of distribution that I showed you. So the only thing that's going to change from one example to another will be the variance. But for a given case, we're talking about IID noise. You're going to fix the variance, have zero mean, and all your noise samples will be pulled from that same distribution.

If you were actually looking at data here for these excursions, if you were actually looking at what the excursions from the baseline are, and you wanted in a numerical experiment-- in a simulation setting, for instance, or in a physical experiment to get an estimate of what the mean and variance are, well, we've got very familiar expressions. You would take the sample mean or the sample variance. The square root of the sample variance would then be your estimate of the standard deviation.

So we can come at the same objects-- well, we have the PDF, which is the mathematical construct, but in an experimental setting, this is how you would go about estimating these. And there's a whole big theory of estimation that tells you whether these are good estimates or not and offers alternatives, and we're not getting into any of that. We're staying close to the basics and close to what makes sense intuitively and what's essentially used all over.

So now we have the task at the receiver of getting a bunch of samples like this and then trying to decide whether what we're seeing is a reflection of a 1 or a 0. If we had 0's sent from here, what we're going to see after we receive the noisy signal is perturbed samples. And so we're going to look at a particular sample and try and decide whether in that bit slot what was sent was a 0 or a 1.

I'm going to actually use a scheme for illustration here that's not the scheme that I've suggested here. Here, I suggested something that's sending 0. If I'm communicating a 0 and I'm sending some other voltage level when I want to communicate a 1, I'm going between 0 and 1. It turns out on the physical channel, if you've got a transmitter with a certain peak power, you're probably better off using a plus V to indicate a 1 and a minus V for a 0 because you're using that transmitter at full power all the time. So you're actually trying to overcome the noise as strongly as possible.

So that's the scheme I'm going to consider. I'm going to consider that when you want to signal a 1, what you're doing at the transmitting end is sending out L samples at plus some peak voltage Vp. And when you want to signal a 0, you send L samples at minus Vp. So this is what we refer to as a bipolar signaling scheme.

So it would be something like this. This is the xn. And this is what I'm using to signal a 1, and this is what I'm using to signal a 0. But in terms of actual voltage levels, this is minus Vp and Vp here. And on the receiving end, what I'm getting at any particular samples-- so I pick one particular sample to look at, and when I look at that sample-- let's say at sample n sub j. So maybe I'm looking in the j-th bit slot and I pick one particular sample time, let we call that n sub j.

And I have to decide, am I looking at plus Vp with noise or am I looking at minus Vp with noise? That's a decision. I know the Vp's-- assume that we've taken care of the scaling and so on across the channel. And I know the characteristics of the noise. I know that the noise samples are Gaussian, zero mean, and some variance.

So if I draw a picture that's turned sideways here in terms of the received signal, let's see. I might get something centered around minus Vp or something centered around Vp. If a minus Vp was sent, then it's got a noise added to it. The noise has a Gaussian distribution. So this is the distribution of values I expect if a 0 was sent.

So this is-- let me call it-- it's the distribution of Y-- I'm not going to put all the attachments here-- if a 0 was sent. Because my shorthand notation for the density of Y assuming a 0 was sent. I haven't drawn a very good Gaussian, but you get the idea. And here's the distribution of Y if a 1 was sent.

So what I'm actually measuring is some number out here. I get some number. And I've got to decide, did that come from having sent a 0 and getting this much noise or did it come from sending a 1 and getting this much noise? That's the problem. So if 0's and 1's are equally likely, what do you think is a sensible rule here? Just pick a threshold where these two cross. Threshold in the middle.

So if the sample is above the threshold, you declare a 1. If it's below the threshold, you declare a 0. What if 0's and 1's were not equally likely? Suppose it was much more likely that you would get a 1. And suppose we're still thinking in terms of threshold rules, what might you want to do? Suppose it's much more likely that we get a 1.

AUDIENCE: Move the threshold to --

PROFESSOR: Sorry?

AUDIENCE: Move the threshold to the left.

PROFESSOR: Move it to the left. So you want to actually allow for the fact that most of the time you're getting 1's, and so you really have to get close to the 0 before you going to declare a 0. So your bias kind of gets built in. Now this is just thinking as an engineer what you might do. It turns out that for Gaussian noise, the optimum decision rule in terms of minimizing the probability of error is exactly a threshold rule of this kind. And the analysis will tell you where that threshold should be.

So we're not getting into proving that this is the optimum, but it turns out with Gaussian noise, the minimum probability of error decision rule for this kind of a hypothesis test-- this is a classic hypothesis test-- is to pick a threshold. Now that's not true necessarily for other sorts of distributions, it's not true for the settings, but for the Gaussian it turns out it's what you have to do.

So let's just assume equal prior probabilities. So 0's and 1's come at you with equal probability, and we now have to figure out what the probability of error is. So there's a slide here with some computation. Let me just walk you through that. We don't have to follow all the details and you can study it and more-- I mean, you can study it at leisure, but it's the same picture I showed. OK?

AUDIENCE: [INAUDIBLE]

PROFESSOR: Yeah?

AUDIENCE: [? Sorry ?] [? to ?] [? interrupt, but I ?] have a question--

PROFESSOR: Yeah.

AUDIENCE: --the Gaussian.

PROFESSOR: [? About ?] [INAUDIBLE]?

AUDIENCE: [INAUDIBLE].

PROFESSOR: Yeah.

AUDIENCE: Is that true when the two Gaussians have different variances?

PROFESSOR: No. OK, I'm assuming-- OK, the question-- the comment was that this rule of the threshold being the optimum is not necessarily true if the Gaussians have unequal variances. But I'm assuming IID noise. I'm assuming Independent Identically Distributed noise. So the noise samples are governed by the same Gaussian right through, and then this turns out to be the optimum rule. Thanks for catching that.

So you can imagine the picture with-- suppose the noise is very sharply peaked for one of these cases and very shallow for the other one. So there's high variance for the 1's and there's low variance for the 0's. You might then anticipate that if you got a signal way over to the left here, you're going to call it a 1, not a 0. So each case needs to be dealt with separately. But assuming these are equal variance, which goes with the IID case, this is the optimum rule.

OK. So let me just step through this. What we're saying now is that what's the probability of making an error? Well, let me actually write down an expression here. So the probability of an error-- this is the general expression. It's the probability that I send a 0-- let me just say that this is the probability of sending 0 times the probability of declaring 1 given that 0 was sent. And then there's the other possibility. The probability that I sent a 1, and here's the probability of declaring a 0 given 1 was sent.

So it turns out these are the only two ways you can make an error, and these are mutually exclusive, and so what you're doing is adding the probabilities of the two ways of making an error. You can either have a 0 sent, and then the question is, what's the probability of declaring a 1 if a 0 was sent? And then you have the corresponding term on the other side.

If P0 equals P1-- in other words, if both of them are 0.5, this is going to be 1 minus P0. If they're both 0.5, then you can pull that out, and what you're looking at for the probability of error is just the sum of the areas under these two tails. Oh sorry, not the sum of the areas. If these are both 0.5, you pull out 0.5-- yeah. It's the sum of those two areas. OK. So 0.5 times the sum of those two areas.

Well in the symmetric case, these two areas are the same. The area to the right of this threshold under the Gaussian here is the probability of declaring a 1 given that a 0 was sent. The area under the tail to the left here is the probability of declaring a 0 given that a 1 was sent. Those two areas are the same. So you'll discover that the probability of error is just the area under one of these tails. Just the area under one of those tails. So that's all you have to compute.

So how do we do that? Well, as the area under a Gaussian. We write down the Gaussian. Let's pretend that this was 0 and this was Vp. It doesn't make a difference as far as the computation of areas goes, but it makes the expressions easier to write. So I'm saying that the area under the table here is equal to the area under the tail there. I can do it either way. I can either center the Gaussian at minus Vp and look at the area to the right of 0, or I can center the Gaussian at 0 and look at the area to the right of Vp.

And the way the expression is written here, it chooses to do it the second way. So what we're saying is, here is the Gaussian. It's centered at 0, so I don't have to subtract any term off that term in the numerator. Here's the 2 sigma squared in the denominator. And I integrate it from Vp onwards.

There's this notation introduced. Vp is square root of ES. The reason is that we're thinking in terms of the energy of a single sample-- or the power of a single sample, they turn out to be the same thing because it's just a single sample. So it's just the notation that's traditionally used. But what we're talking about as Vp there. So the area from Vp to infinity under the Gaussian with the normalization factor here.

Now this is not an integral you can evaluate in closed form. It is a tabulated integral. Tabulated most conveniently in terms of something called the error function. And so when you work through the calculus-- and I won't show it to here, it's in the book, you might do it in recitation, you discover that the probability of error is this error function of the square root of ES over N0. N0's notation for 2 sigma squared. If I translate that back to notation we've been using, it's just Vp over sigma.

So the error performance, the probability of error is a function of the ratio of the peak amplitude on the signal to the standard deviation of the noise. That's sort of the square root of the SNR. The SNR would be square of the amplitude to square of the standard deviation. So this is the square root of the SNR.

And what does this function look like? We can plot it. So that's exactly that computation. This is a simulation on the theory overlaid on each other, but we have 0.5. This function is called the complementary error function. The C is for complementary, erf is for error function, and here's the square root of ES over N0 which we had in the previous expression.

So you're really thinking of signal-to-noise ratio along this axis in dB and the probability of error on a logarithmic scale down here. So as the signal-to-noise ratio increases, as a signal becomes more powerful relative to the noise, the probability of error decreases. Visually what's going on? Let's go back to this picture.

When the noise decreases relative to the signal, what's happening is that these Gaussians are getting more peaked and they're pulling in more tightly, and so there's less chance of confusing the two cases. So it's as simple as that. It's the separation between these two levels divided by standard deviation of the noise that's really going to determine performance. How far apart are these two cases relative to the standard deviation of the noise? That's the square root of the signal-to-noise ratio. That's what determines the probability of error in this case.

OK. So are we done or could we be doing better? If you think of what we did, we looked at the samples in a bit slice, in a bit slot. We took one of those samples and we carried out this decision rule on it. Could we be doing better than that? Yeah?

AUDIENCE: We could look at one more sample?

PROFESSOR: We could look at more than one sample. This was a little bit arbitrary. It was conservative. Why you often do that is because the number of samples in a bit slot is small and you don't want to get near the edges because you're little worried about the transience. You've got a long enough-- if you've got enough samples in a bit slot and the transience have died out, then maybe you can just pick out a bigger chunk in the middle. And so that's what we're going to think to do here. OK.

So it's the same setting, but we're going to average M samples. We've got L samples per bit. We may not be confident capturing all of those were averaging because there's some stuff at the edges, so let's pick M of them. Maybe less than L. Take M of them and compute the average. And I'm doing this just for one of the cases. You'd have to do the same thing for the minus Vp case. So the question is, what does the average do? So why did you want to average them? What was your intuition?

AUDIENCE: Because that would-- it [? would be ?] [INAUDIBLE].

PROFESSOR: OK. So here's the key thing. If you've got independent noise samples and you average them, you're going to decrease the variance. If you've got M independent noise samples from an IID process, you decrease the variance by M. This doesn't hold if the noise samples are not independent. In fact, if one noise sample equals the other, then when you add the two, you get something whose variances is four times rather than just twice. So it's critical that these be independent.

So if we've got independent samples-- independent noise samples from one sample to the next and we average them-- well, let's just average both sides of this equation. We've got the average of Y going to be the average of these values, which is just going to be Vp again because it's constant at Vp, plus the average of W.

Here's the other interesting thing. We're not going to try proving this, but it turns out that the average of a sum of independent Gaussians is, again, Gaussian. You might believe that if you think of the central limit theorem. You think of each of these Gaussians being approximated by sums of random variables. So the sum of these Gaussians is then a sum of just a larger number of random variables that should still be Gaussian. So the sum of an independent set of Gaussians is, again, Gaussian.

So all I need to know for this average W since it's Gaussian is what is its mean and what is its variance? It turns out if you add up a bunch of zero mean random variables, you get something with zero mean, no surprise. And if you add-- if you take the average, then the variance actually drops by that factor M.

So what you're going to do is take the average of the signal, average of the noise. That shrinks the noise component. You have the same kind of picture but now with a higher signal-to-noise ratio. Now what you've got in the numerator instead of ES is EB, which is M times ES. You're summing the energies of all the samples that you've taken. And that's what we refer to as EB, it's the energy of the bit.

All right. It turns out that that has all sorts of implications. You certainly want to be averaging if you've got this kind of setting, because otherwise you're leaving all these samples on the table and not making good use of them. So if you're really getting ambitious, you really want to be extracting all of that.

Also, if you want to maintain the same error performance and the noise intensity increases, then you're going to want to have more samples per bit. You may want to slow down your signaling rate so you can put more samples per bit. It turns out in the deep space probe examples that we've been talking about, that's exactly what's happening. If you look at *Voyager 2,* it was transmitting at 115 kilobits in 1979. That's the year, I joined the faculty, that's a long time ago. That was near Jupiter.

Last month-- I mean, it's gone past Jupiter, Saturn. The other planet I only like to say the Greek name of because it comes out wrong when I say it. It's Ouranos. And then Neptune. So it went past all of these. And now it's about 9 billion miles away. It's twice as far away from the sun as Pluto is.

But look at the transmission rate now. It's 160 bits per second. So it's greatly reduced. And the reason is that over this extended interval, the energy per sample that arrives at Earth is just minuscule. I mean, it was small enough to begin with from Jupiter and look at what it does now. So it's about 1,000 times less in power and you've gone down 1,000 times less more or less in your signaling rate because you're trying to put that much more time in the signal.

So these trade-offs are driven by trying to get the same energy per bit for a given noise to maintain the performance. As I was reading up on this, there were little references to things that went wrong. The only a handful of things that are listed as having gone wrong, but they turn out to be related to decoding. So there was a command that was incorrectly decoded and kept some heaters on for very long and caused some malfunction. Here was a flipped bit.

This is one of only-- these are a few of only a small list of things that are listed as having gone wrong. But a flipped bit here caused a problem. You've got very few bits in these computers to begin with. Remember the numbers we had last time. So a flipped bit can cause trouble. OK. Let's do one last piece here. We're going to try and be even less conservative.

So suppose I know that when a 1 is sent, what I receive is a waveform of a particular type. So the piece of the response corresponding to this has some particular shape. Suppose I know that. OK. So nothing is constant here. This is the actual y of n sequence. And then to this, I'm adding noise.

So here's the thing. I've got a yn which is no longer just a constant plus noise, it's some known profile plus noise. That known profile is actually what the xn is going to look like when it goes through the channel. I should perhaps have called it y0 of n, but let's stick to x0 of n. So x0 of n is known, and we've got the noise.

The question is, do you want to just be averaging or do you want to try something else? If I've got this kind of signal received and I've got the same amount of noise added to each sample, which of these samples is more trustworthy? Which sample do you want to weight more? I've got some amount of noise adding into all of these samples, so there's some standard deviations' worth on each of these. Which is the most trustworthy sample here? Yeah?

AUDIENCE: The one on the right?

PROFESSOR: Yeah. It's the one on the right because it's got the largest amplitude. By itself it has the largest signal-to-noise ratio. So if you're going to combine these samples, you would think that you would want to put more weight on the sample that was larger. So you can actually formulate that analytically.

So we're going to combine the received samples with some set of weights an. Here's what it's going to do on the right-hand side of that equation. Again, when you take a weighted combination of zero mean Gaussians, as you get a zero mean Gaussian. So all you need to know is what's the variance of a scaled Gaussian? So let's see.

If I have a wn having variance sigma squared, what do you think is the variance of 3 times wn? 3 times wn means the excursions are scaled by 3, so what's the variance? 9. So scaling by a particular number scales the variance by the square of that number.

So the Gaussian you're adding in here has a variance which is sigma squared times the sum of the W squared. Sorry. The sigma squared times to sum of the A squareds. That's what the variance of the Gaussian is. So you can actually write a very simple optimization problem. What choice of weights maximizes the signal-to-noise ratio? And you discover, indeed, exactly that you're going to put the largest weight on the largest sample.

And when you do that, the resulting signal-to-noise ratio is, again, energy of the signal that was transmitted divided by the variance. So if you do the optimum processing with this so-called matched filtering, you're going to get to energy of the sample-- sorry, energy of the bit over the noise variance governing the performance. So it's the bit energy over the noise variance that's going to determine performance provided you milk that bit slot for everything it's worth by doing the match filtering. OK. We'll leave it at that for today.