PROFESSOR: I want to spend a little bit of time today finishing up what we were saying last time, about orthonormal expansions. Just, essentially, to get a better understanding of what it means to go from finite dimension to infinite dimension.

For the most part, you can just ignore the question of being at an infinite dimensional space, and everything is pretty much the same as it is in finite dimensions. But there are a few things you have to be a little careful about. So, let's suppose that we have a set of orthonormal vectors. Like the sinc vectors for example, that we were using for the sinc functions that we were using, to talk about the Fourier series.

And the thing that we said last time at the end of the hour, we quoted this theorem which was almost obvious, I think, in terms of the other things that we did. Which said that if you start out with some vector v, you can project it successively onto the set of orthonormal functions in such a way that you get closer and closer to the vector that you're looking for. And the theorem that we had was that the projection on the whole space, which we call u, in fact is the same as the limit of this sum of orthonormal functions as we take the limit adding more and more functions into this sum.

So this was sort of the same kind of thing that we were doing when we were going -- it's the same sort of thing that we did when we were doing the projection theorem for finite dimensions. You remember the thing that we did there was to first project this function onto a single waveform. Then we found the part of the waveform that was orthogonal to the first waveform. That gave us our second waveform. And the thing that we're doing here, which is sort of different, is we're starting out with the orthogonal sequence to start with.

Anyway, the thing that we showed was that the vector we start with, minus this approximation, which is in the space generated by all of these p sub n's, that this difference has to be orthogonal to each p sub n. Because that was a way we constructed it. So this difference is -- well, the limit goes to zero, but also the difference is orthogonal to each phi sub x.

This is the thing which is new, which we didn't get from looking at the Fourier series. We could have gotten it from looking at the Fourier series, but this is the thing which is general for every orthonormal expansion in the world. Not just a Fourier series, except all of them.

And then we say that an inner product space has accountably infinite dimension. If accountably infinite set of orthonormal vectors exist, such that only the zero vectors are orthogonal to each other.

This comes back to one of the issues that we faced when we were going through the Fourier series. Namely, we showed all this nice stuff, which was really about approximating a waveform by the Fourier series. The thing that we never showed, and the thing which is messy and difficult, is that if you take a time-limited function and expand it in a Fourier series, how do you know that when you get all done, you're actually going to get the function back again? This is the part of it which we never talked about. We talked about how you generate all of the Fourier coefficients, and all of that was fine. And we showed that when you were all done, this representation function that you had was in fact orthogonal the thing that was left over. But you never showed that the thing that was left over went to 0. And the thing this is saying is you don't have to worry about that too much. Because the thing which is left over has to be orthogonal to everything you started with. And this sum here has to go to some limit.

What is this limit in the case of the Fourier series? Suppose we start out with an arbitrary L2 function. We now try to expand that arbitrary L2 function in a Fourier series from minus t over 2 to plus t over 2. And we can do that. When we're all done, this series here is in fact going to be the part of the function which lies

between minus t over 2 and plus t over 2. And this difference here -- well v minus u, the difference between the original vector and this representation vector is going to be this part of the function which lies outside of minus t over 2 to plus t over 2.

So the next thing that we did then was to show that the truncated sinusoids and the sinc weighted sinusoids both in fact span L2. In other words, any function in L2, after you represent it in terms of that series, what you have left over has to be orthogonal. In other words, if you take this entire set of truncated sinusoids, every non-zero function is orthogonal to all of that. Every non-zero function in this L2 equivalent sense.

So in fact, the thing you get out of this is this part of looking at functions which we always ignored.

There's another issue that we ignored, when we were looking at functions, which comes out of this. Since v, by assumption is an L2 vector. In other words, it's an L2 function aside from this L2 equivalent stuff, and phi sub m is an L2 vector also, this inner product has to be finite, by the Schwarz inequality. How did we get around this when we were dealing with the Fourier series? Does anybody remember? Of course you don't remember. I hardly remembered that.

The way we got around it was by saying that this orthonormal function which was in a truncated sinusoid. It was truncated to minus t over 2 and plus t over 2, and the waveform was just e to the i 2 pi f t. The magnitude of that function was always less than or equal to 1. And because the magnitude was always less than or equal to 1, you could just take the integral of v of t times phi sub n of t, and you could show directly that that integral always existed. Because of the special property of the sinusoids.

Here what we're saying is, you don't have to worry about that any more. You can use arbitrary L2 functions as the components of an L2 orthonormal expansion. And it still works. So every one of these things has to be finite also. So, in fact we're buying something out of this. At the end of the notes on Lectures Eight to Ten, you will notice something that I'm certainly not going to hold you responsible for.

Something called the prolate spiroidal expansion. And it's just given there primarily as one more example of an orthonormal expansion. It's a very interesting orthonormal expansion because it has the property that if you start out asking how much energy can I concentrate within a fixed time interval, and a fixed bandwidth interval. Which is one of the things which make this whole subject a little bit fishy in a whole and certainly very, very messy.

If I start out with a time limited function and go through the Fourier series, these truncated sinusoids spill their energy out of the band enormously. In fact, one of the problems at the end of Lectures Eight to Ten carries you through the process of just how much energy you can have outside of band by using these truncated sinusoids. Because truncated sinusoids, when you look at them in a Fourier series, have a lot of energy outside of where it should be.

So that this particular set of prolate spiroidal functions gives you the answer to the following question. I would like to find that function, which is limited to minus t over 2 plus t over 2, which has the largest amount of energy, largest fraction of its energy, within the band minus w to plus w. What is that function? Well, that function happens to be the zero order prolate spheroidal function. It's the nice property that it has. And it's a nice function, which almost looks like a rectangular function. But it just trails off to 0.

And I say, OK, if I want to construct an orthonormal expansion, and I want to find another function which is orthogonal to that function and has the next biggest amount fraction of its energy, inside of this -- strictly inside of this time limit, and as much of the energy within the frequency limit as possible, what's that next function? Well, you solve that problem. If you're very good at interval equations. And I certainly couldn't solve it. But, anyway, it has been solved.

I mean, physicists for a long time have worried about that particular kind of question. Because it comes up in physics all the time. If you time-limit something and you take the Fourier integral, how can you also come as close to frequency limiting as possible? So this particular prolate spheroidal set of orthogonal functions,

in fact, exactly solves the problem of how do you generate a set of orthonormal functions which in fact have as much energy as possible both frequency-limited and time-limited?

And what you find when you do that is, when you take 2 w t of them, at that point the energy in these orthonormal functions, the part of it that's inside the band, really starts to cut off very, very sharply, when w and t are large. So that in fact you get 2 w t of these functions, which have almost all of their energy in this band. And everything else has almost no energy in the band. So if you ever get interested in the question of how many waveforms really are there, which are concentrated in time and frequency, that's the answer to the problem. And don't bother to read it now, but just remember that if you ever get interested in that problem, which I'm sure you will at some point or other, that's where the solution lies. And I hope I gave a reference to it there. I think I did.

Anyway, we have these functions which span L2. And there's at least one other orthonormal expansion that we have. Which is both time- and frequency-limited. And either at the end of today or at the beginning of Monday, we're going to find another particularly important sequence of orthonormal functions. Which we actually use when we're transmitting data.

So to give an example of what we were just talking about, the Fourier series functions span the space of functions over minus t over 2 to t over 2. And when normalized, these functions become 1 over the square root of t times what we started with before. Namely, a sinusoid truncated. Before we were dealing with the orthogonal functions without the 1 over square root of t in it. If you want to make them orthonormal, you get this square root of 1 over t, because when you take the energy in this function, you get t. If you don't believe me, set k equal to 0 and look at that. And even I can integrate that. And when you integrate 1 from minus t over 2 to t over 2, you get 1. So, then you have to multiply by the square root of 1 over t.

When you view the Fourier series functions in this way, it's nice because they're orthonormal. It's not nice because the square root of 1 over t appears everyplace.

But the nice thing about it is that then when you expand in the Fourier series, you don't find this t anywhere. Namely, v is equal to the limit of these approximations where the n'th approximation is just the sum from minus m to m of alpha sub k. phi sub k and alpha sub k is just this inner product. So, again, you got something nicer by looking at these things in terms of vectors You get nice statements about how these things converge, and you also get a very compact way of writing out what the expressions are.

You also get something a little bit fishy, which a lot of people in the communication field, especially ones who do theoretical work, run into problems with. And the problem is the following: when you start dealing all the time with vectors, and you forget about the underlying functions and the underlying integrals, you start to think that the subject is simpler than it really is. Because you forget about all the limiting issues. And when you forget about all the limiting issues, it's fine almost all the time. But every once in a while, you get trapped. And when you get trapped, you then have to go back behind all of the vector stuff and you have to start looking at these integrals again and it gets rather frustrating. So you ought to keep both of these things in mind.

Let's go on, let's get back from mathematics into worrying about the question of how do you send data over communication channels. And this is just a picture that we saw starting on Day One of this course, which says the usual way of doing this is, you start out with -- you start out with the source. You break the source down into binary data, and then you take the binary data and you transmit it over a channel. And this is the picture of what you get when you're trying to transmit binary data over a channel. And here, we've broken down the encoder, as we called it, into two pieces. One of which we call the discrete encoder and one of which we call modulation.

Now, this is a little bit fishy also. Because there are a lot of people who now talk about coded modulation, where it turns out to be nice to combine this discrete encoder with the modulation function. And when you actually build modern-day full encoder, namely the whole thing from binary digits to what goes out on the channel,

you very often combine these two functions together.

What is it that allows people to do that? The fact that they first studied how to do it when they separate the problems into two separate problems. And when you separate the two problems, what we wind up with is binary digits coming in. You massage those binary digits strictly digitally. And what comes out is a sequence of symbols. And, usually, the sequence of symbols comes out at a slower rate, and the binary digits come in. For example, if you take two binary digits and you convert them into one symbol, from a symbol alphabet with force of size four, then for every two in you get one out. If you take three binary digits coming in, then you need a symbol alphabet of size eight here. If you move down in rate from four binary digits to one symbol, then of course you need an alphabet of size 16. So the size of the alphabet here is going up exponentially, as the great advantage that you get is going down. We will talk about that more as we move on. You should sort of keep in mind that there's a strange relationship between size of alphabet and the rate at which you can transmit. As you try to transmit at higher and higher rates here, the size of your alphabet goes up exponentially, with the rate gain that you get.

In fact, when you look at Shannon's famous formula of how fast you can communicate on a channel, you find a logarithm of a signal to noise ratio. Of 1 plus a signal to noise ratio. When you look at that signal to noise ratio and you look at this alphabet size in here, you can almost see just from that why in fact that logarithm in these capacity formulas. And we'll come back to talk about that again more later also.

But, anyway, what we're going to do at this point is to separate this into the problem of discrete encoding and modulation. And in modulation, you start out with some arbitrary alphabet of symbols. You turn the symbols in that alphabet into waveforms. You transmit the waveforms. You then get the symbol back, and then you put it into a digital encoder, which gets the binary digits back.

Now, what order are we going to study these in? Well, we're going to study the modulation first. And we're not going to study this at all, essentially, except for a few

very, very simple-minded examples. 6.451, which is the course that follows after this, which might or might not be taught in the Spring term, and incidentally those of you who want it to be taught should send me an email saying you really need to take this next term for some reason or other. Because otherwise it will be given in Spring of the following year instead of Spring of this year.

There's another side to that issue, which is a wireless course is going to be given in the Spring of this year. And if you want to take a wireless course and postpone taking the coding course until the following year, then in fact it might be better to do the postponing job. I would recommend that those of you who are interested in wireless take the course now. Because if you're looking for research problems in a communication area, wireless is probably the hottest area around, and the area where the most interesting problems occur. So, you have to make that tradeoff. If you want to take both of the courses, great. Send me an email and say you want to take the coding course. But anyway, there'll be very little coding in this course, very little discrete coding. And mostly we're going to look at simple ways of taking simple symbol sequences, turning them into waveforms and then transmitting them on channels.

AUDIENCE: So, basically, when you do the source coding, you have to [UNINTELLIGIBLE] binary --

PROFESSOR: Binary to symbol to waveform. Yes.

AUDIENCE: [UNINTELLIGIBLE]

PROFESSOR: OK, why don't I go from waveforms directly to waveforms instead of going from waveforms to symbols to binary digits, and then I just have to go back up again. A bunch of reasons. One of the reasons is that some of the stuff that you transmit is already digital to start with. When you look at what goes over a network today, for example, it's carrying digital data, it's carrying analog data. It's carrying images. It's carrying everything you can imagine. If you want to design a modulation system which goes directly from analog data to channel data, and you have a hundred

different kinds of analog source data, and you have a hundred different kinds of channels, then you're going to have to build one encoder for every combination of source and channel.

In other words, you've got to built ten thousand devices. If your interest is in keeping engineers employed, which I think is a very good interest, that's a very good philosophy to take. But, unfortunately, most people who build communication equipment say, we would really rather have just a hundred -- well, two hundred different devices. One hundred devices which turn all these different sources into binary digits, and another hundred devices which turn the binary digits into something that can be transmitted over any channel you want to.

I mean, this is just a general example of what people call layering. You want to turn systems into systems with a bunch of layers in them, where each layer is standardized in some way. And only has to take care of a few particular functions. And, in fact, that's what we're doing here also. Because we're dealing with one layer, which is doing discrete encoding. Which in fact is sort of generating, for the most part, binary digits out of the discrete encoder. Where, as you go through all of this stuff and you come out with binary digits, some of which are wrong, you can do the decoding and get the correct binary digits out. If you look at an awful lot of coding theory, you'll find out it doesn't pay any attention at all to what's going on in the channel. Lots of people who've studied coding all of their lives, and decoding all of their lives, live in this mathematical theory of abstract algebra. And they have no idea of what channels are. And they survive because of this layering principle. So if you want to employ engineers, it's nice to have layering also. Because engineers don't have to know as much then.

Of course, you people should know it all. So because, then you can do anything. And you can be part of those very rare people who can put it all together. I was just at the 70th birthday party of Irwin Jacobs this past week. And Irwin Jacobs is the CEO of Qualcomm, which is the company that started to build CDMA wireless systems. For a long time, the CDMA systems were thought of as probably better than TDMA and FDMA, but much more expensive. And eventually, we're in the

situation where all the new systems being designed are all using CMDA. As a result of this, Irwin Jacobs is a very wealthy person. We went to a symphony Friday night, out in San Diego, which was given specifically for him. Partly because he'd just given $110 million to the San Diego Symphony. So you can figure from that that he's fairly wealthy.

Well the point of all of that is, he started out as a faculty member here. He was one of the authors of Rosencraft and Jacobs, which is sort of the Bible of digital communications systems from the `60s. And people still use it, it's still an excellent book. And in doing that, he really learned the communication trade from soup to -- I guess, soup to nuts is the way we put it now. And he could do the whole thing. And because he could do the whole thing, because he understood coding, because he understood modulation, because he understood channels, this is why he could design systems that really work.

I was talking to the Chief Technology Officer out there, and the Chief Technology Officer said his job was really very easy because Irwin did all of that himself. And he still does it. So it really makes sense to know something about all of these pieces. If you want to become a billionaire. And if you want to become a billionaire without cheating. Now, many people become billionaires by cheating, and you've heard of many of them but. Well, anyway. Enough of that. I mean, none of us really want to be billionaires anyway, -- I mean, there's nothing you can do with more than a billion dollars, right? It's all wasted.

Let's move on. We've gotten rid of digital coding, saying we're not going to deal with that here. Let me take the PAM out of here because I don't want to even say what it is yet. If we take this box I called modulation before, which was one of the two main pieces of a channel encoder, we can break it down into two pieces. Namely, mainly we're going to be layering things again. We start out with this symbol sequence, which is what comes out of the encoder, which is usually just a short sequence of binary digits. So the symbols can be thought of as two binary digits in a sequence, or four binary digits, or six tuples of binary digits, or what have you. And all of those are common.

There's then a signal consolation that turns symbols into signals. Now, notation in the field is totally non-uniform, because most people use symbols and sequences and waveforms and binary n-tuples, all totally synonymously. And the distinction we'll try to make here is that symbols are things like binary digits, that don't have any numerical meaning to them. I mean, a 1 and a 0, the only thing important there is that 1 and 0 are different from each other. All computer scientists call 1 and 0's Alices and Bobs. Lots of other people call them plus-1's. and minus 1's. Doesn't make any difference what you'd call them, it's just an alphabet with two values in it.

When we talk about signals, we're talking about numerical values. So in fact, you could have a signal constellation with two elements in it, where what you're doing is mapping 1 and 0 into plus 1 and minus 1. Now, that's a pretty silly kind of situation, but it still has the value of saying, you're turning things where there's just an alphabeticized 2 into something where you're saying, these are numerical values and you're interested in the difference between the numerical values. And the difference is measured in the ordinary way of measuring difference for numerical values.

Or, these can be vectors also. But vectors in an inner product space where, again, you have length. And you have distance. So you have numerical values here. You don't have numerical values here. So we're going to talk about how you do this. And then, for the most part, we're going to define modulation as what happens when you go from the signal sequence to the waveform. You might realize that our notation is lousy here. Because I'm calling modulation this whole thing. And I'm also calling modulation just this thing. And I'm doing that because there doesn't seem to be any other reasonable word for either one of these things. So, and I'm going to later split this into two pieces also. But that will come later.

So, anyway, we're going from symbols to signals to waveforms. Which might look remarkably similar to what we did with sources. But we just did it backwards with sources. We went from waveforms to signals to symbols there. And here we're doing the same thing.

The modulator often converts a signal sequence to a baseband waveform, and then converts the baseband waveform to a passband waveform. And, just historically, people used to think of modulation as the process of taking something at baseband and converting it up to some passband. Now it's recognized that the interesting problem is not, how do you go from baseband to passband. Which is just multiplying by cosine wave, not much more to it than that. Well, it's a little more to it, but not much.

And the interesting problem is, how do you convert signals into waveforms. Which is why we went, one of the reasons we went through all of this stuff about L2 waveforms and orthonormal expansions and all of this stuff. So, for the time being, we're going to look at modulation and demodulation, without worrying about what bandwidth any of this occurs at. So we'll just look at it at baseband.

So the simplest example of all of this, so simple that it almost looks like it's silly, is to map a sequence of binary digits into a sequence of signals from the constellation 1 and minus 1. So all you're doing there is mapping 0 into 1 and mapping 1 into minus 1. In other words, the 0 and 1 binary digits are mapped into 1 and minus 1. Why do you do it this way, which is a little confusing, mapping 0 into 1 instead of mapping 1 into 1? Well, primarily, so you can look at multiplication of signals in the same way as you look at modular to addition of binary digits. It just turns out to be a little more convenient that way. And it doesn't make a whole lot of difference. The point is, we're going from 0 1's to signals which are 1 and minus 1.

Then this sequence of signals is mapped into a waveform which is the sum of u of k times the sinc function t over capital T minus k. In other words, you're thinking of each one of these signals, now, think of it as being a sum of impulses, delayed impulses. And then think of taking a little sinc, sinc t over t and putting it around each one of those impulses. In other words, think of passing each of those impulses, which is weighted by one of these values, u sub k, into a linear filter whose response is sine of x over x.

Anybody

see anything a little bit fishy about that? Have you ever built a filter whose response was sine x over x? AUDIENCE: It's hard to.

PROFESSOR: It's hard to, why?

AUDIENCE: [UNINTELLIGIBLE]

PROFESSOR: Because it's not causal, yes. We're going to talk about that more later today. Communication engineers hardly ever talk about causality. They hardly ever talk about whether something is realized or not. And the reason I want to say this a number of times is that one of the parts of any receiver is a timing recovery circuit. And what the timing recovery circuit does is, it tries to figure out what the transmitter timing is, in terms of what it's receiving. And when you're figuring out what the transmitter timing is in terms of what you're receiving, the most convenient way of doing that, if you're sending a pulse or something, you would like the receiver timing to peak up at the same time that the transmitter timing is peaking up, in terms of the pulse. In other words, you want to get rid of the propagation delay and just ignore that. The receiver timing is going to be delayed from the transmitter timing exactly by the propagation delay. Now, if we always do that, causality becomes totally unimportant.

Now, one of the problems with a sinc function is, it starts at time minus infinity. And even if you delay the receiver timing by an infinite amount of time you can never use the communications system until it's time to tear it down and bring in a new one. Which, unfortunately, is what happened to the third generation wireless systems. By the time, people figured out how to build them, everybody was saying, ah, old-fashioned stuff, we're going to go directly onto to fourth generation. And they might go directly from fourth generation to fifth generation. Who knows.

Anyway, you can't implement these even with the receiver timing different than the transmitter timing. But you can always approximate things with enough delay between transmitter and receiver. So that you can approximate any filter you want to, without worrying about causality at all. And it's hard enough designing good filters without worrying about casuality that you don't want to bring that into your

picture at the beginning. So communication engineers usually say that those timing issues are not important. And a little bit of delay is not going to hurt anything. All of Shannon's theory was as successful as it has been, and transformed the whole communication industry, because he never talked about delay at all. He just ignored the question of delay from beginning to end. And when you look at his capacity formulas, they are assuming that you have as much delay as you need between transmitter and receiver. And, assuming that that isn't important, and in terms of the propagation delays, and the filtering delays in most modern communication systems, those delays are not really important. The propagation delays you can't avoid. So you might as well ignore them. Which is why you build timing recovery circuits. And the filtering delays are usually unimportant. Relative to all of the other delays that come into these system. So for the most part, we're just going to ignore those questions.

If you have no noise, no delay, and no attenuation, the received waveform is going to be the same as the transmitted waveform. And then what you would like to do is to sample that and convert it back to binary. So, that's what the receiving side of this trivial system is doing. Now, why can you ignore attenuation? Anybody have any idea why we might want to ignore attenuation?

Well, it's like all of these other things. You don't ignore it, but the question you ask is, can I separate the issue of attentuation from the other issues that I want to look at now. In other words, can I layer this problem in such a way that we can deal with our problems one by one. And the problem of attentuation is something that communication engineers have had to deal with from day one. In dealing with it from day one, they have dealt with all of the different ways of losing power that you have. Which includes attenuation in the actual medium between transmitter and receiver. It deals with the attenuation you get in the receiver by building filters and by all the other things you do there. When you get all done with that, there's only one thing that's important. Because you can deal digitally with very, very small signals. And the only thing that's important is what happens to the noise. You get noise in the communication medium. That comes into the receiver. And now, any time you amplify what you receive, they're going to be amplifying the noise as well

as amplifying the signal.

An easy way to deal with that is to assume that there isn't any attentuation in what you're calling the signal, because you're going to amplify that to a reasonable value to operate on it anyway. So in fact you're amplifying the noise. So the only thing you're interested in is, what's the ratio between the signal power and the noise power.

So, those issues, we can deal with completely separately from these issues of what kind of waveforms do we want to choose. What should we choose in place of this sinc function, which is impractical because it causes too much delay and it's also hard to build the filters. So we'd like to deal with that question separately from the question of attenuation.

There's a short section in the notes, which I'm not going to go into, because you can read it just as easily as I can talk about it, about dB. And why people use dB to talk about all of these questions of energy losses. And why, in fact, there's a whole set of engineers whose function is to deal with link budgets. In other words, when you build a communications system, you're losing power everywhere along the way. You're losing it in the median. You're losing it by the way you build the antennas. You're losing some here, you're losing some there. And what these people do is, they look at all of these attenuations together. And they multiply together. So, in fact, it's easier to take the logarithm of all of these terms. And when you take the logarithm, that's where you get decibels from. Because these people, instead of taking natural logs, which would seem like a much more reasonable thing, always take the logarithm to the base 10 and then divide by 10.

And what the section in the notes says is that that's a practice which has grown up because it's very easy to do mental arithmetic with that. The logarithm to the base 10 of the number 2 which is one of the biggest numbers that ever appears. 2's are more important than thousands, OK? And the logarithm to the base 2 of 10 is 0.3. And when you divide it by 10 you get 3 dB. So a factor of 2 is called 3 dB. This means that a factor of 4, which is 2 squared, is 6 dB. Factor of 8 is 9 dB, and so

forth. And this gives this table of all these numbers which all communication engineers memorize.

The other reason for it is that if you're talking to a communication engineer, he will recognize you as a member of his fraternity if you let the word dB slip several times. Don't even have to know what it means. You just talk about dB. And you say, my income is 3 dB lower than your income. And that makes him very happy.

So. Anyway. That's all we need for this simple example. We now want to go into more complicated things. OK pulse amplitude and modulation. This is one of the major ways of turning bits into signals, and then turning the signals into waveforms again. Again, I'm doing this first not because it's the most important scheme to talk about, but because we want to understand these things one by one. And when we understand PAM, we'll then talk about QAM. And you'll understand that. And then we can go on to look at other variations of these things.

So the signals in PAM are one dimensional. The constellation, the only thing it can be, since it's one dimensional, one real dimension, is a set of real numbers. So you're going to modulate these real numbers. And that, we're going to talk about later. How do you find this function here? We're just going to take these real numbers, which are coming into the transmitter one by one out of the digital encoder. You're going to take these numbers. You're going to view them as being multiplied by delayed impulses, and then pass through a filter. And the filter's response is just, impulse response is just, p of t.

In other words, what we're doing is saying, we don't like the sinc function. Therefore, simplest thing to do is replace the sinc function by something we do like. So we're going to replace the sinc function by some filter characteristic, which we like. And that's the way to modify this previous example into something that makes better sense.

So we're doing two things here when we're talking about PAM. One, we're talking about generalizing this binary signal set. And, two, we're talking about generalizing the sinc function into some arbitrary impulse response.

So a standard PAM signal set uses equispaced signals symmetric around 0. And if you look at the picture, it makes it clear what this means. It's the same thing that we were using all along when we were talking about quantization. If you're looking at one dimensional quantization, that's a very natural way to choose the representation points. Here, we're doing everything in the opposite way. So we're starting out with these points. Well, we're starting out with eight symbols and turning them into eight signals. And when you take eight symbols and turn them into eight signals, perfectly natural thing to do is to make each of these signals equispaced from each other, and center them on the origin. This is not something we have to see later. I think you can see that if these symbols all are used with equal probability, and you're trying to reduce the amount of energy that the signals use, which will then go through into reducing the amount of energy in the waveforms that we're transmitting, it's nice to have them centered around 0. Because if they have a mean, that mean is just going to contribute directly to the expected mean square value of the signal that you're using. So why not center it around 0.

Later on we'll see many reasons for not centering it around 0. But for now we're not going to worry about any of those, and we're going to center it around 0.

And the other thing is, why do you want them to be equally spaced? Well, I'll talk about that in just a second.

Anyway, the signal energy in these equally spaced signals, you can calculate it to be d squared times m squared minus 1 divided by 12. I think we sort of did this when we were worrying about quantization. There's a problem at the end of lectures 11 and 12 which guides you by the hand in how to calculate this. Or you can sit down and just calculate it by hand, it's not hard to do. But, anyway, that's the mean square value of these signals. If you assume that they're equiprobable.

Now, if you take m to be 2 to the b, now what's b? b is the number of binary digits which come into this signal former when you produce one signal out. Namely, if you bring in b binary digits, you need an alphabet of size 2 to the b. If you have an alphabet of size 2 to b, then you're going to need m equals 2 to the b different

signals. So, usually when you have a standard PAM system, that number there, 8 PAM, means 8 signals. 8 is usually going to be replaced by 4, or 16, or 32, or 64, or what have you. And, usually, not anything else. Because you're usually going to deal with something which is a power of 2. Because the logarithm of this to the base 2 is the number of bits which are coming into the signal former for each signal that comes out.

This goes up very rapidly as m squared goes up. In other words, you try to transmit data faster by bringing more and more bits in per signal that you transmit, it's a losing proposition very, very quickly. It's this business of a logarithm which comes into everything here.

We're going to talk about noise later. We're not going to talk about it now. But, we have to recognize the existence of noise enough to realize that when you look at this diagram here, when you look at generating a waveform around this, or a waveform around this, however you receive these things, noise is going to corrupt what you receive here by a little bit. Usually it's Gaussian, which means it tails off very, very quickly. With larger amplitudes. And what that means is, when you send a three, the most likely thing to happen is that you're going to detect a three again.

The next most likely thing is you'll detect either a four or a two. In other words, what's important here is this distance here. And hardly anything else. If you send these signals with equal probability, and the noise is additive, the noise does the same thing no matter where you are along here. Which says that this standard PAM set is almost the only thing you want to look at. So long as you assume that the noise is going to be additive. And the noise is going to affect everything along this line equally. It says that you just want to make the spacing between points big enough that it will pretty much avoid the noise.

So, the point of all of that is that d is fixed, ahead of time. You can't play with that. You can play with m. When you play with m, you're playing a losing game with energy. So that's why standard PAM is the thing which is used with PAM. Not much you can do about it.

And say here that the noise is independent of the signal. We will talk about this later. The noise being additive. The noise being independent of the signal are both saying almost the same thing. Which will be obvious to you in a little while. But not until we start talking about noise. We don't want to start talking about noise now. So, for now, we deal with the noise just by saying that every signal must be separated by some minimum amount.

Again, what we said about delay and attenuation. Let me say it again, because it's important enough that you have to understand it. After you go through two or three problem sets, you will not understand it again. Because you'll get so used to dealing the receive signal as the same as the transmitted signal that you'll forget the weirdness in that. I mean, people become accustomed to extraordinarily weird things very, very easily. And especially when you're taking a course and trying to get the problems done, you just take things which are totally ridiculous and accept them if it lets you get through the problem set. So I want to tell you right up front that there is some weirdness associated here. It is something you have to think about. After you think about it once, you then accept this is a layering decision. You ignore delay, since the timing recovery locks the receiver clock to the transmitter clock plus the propagation delay. And in fact, it can lock the receive clock to any place that wants to lock it to. So we're going to lock it in such a way that the receive signal looks like the transmitted signal.

And the attenuation is really part of the link budget. We can separate that from all the things we're going to do. I mean, if we don't separate that, you have to go into antenna design. And all this other stuff. And who wants to do that? I mean, we have enough to do in this course. It's pretty full anyway. So we're just going to scale the signal and noise together. And that's a separate issue.

So now we want to look at the thing which is called PAM modulation. In other words, in this one slide we separated the question of choosing the signal constellation, which we've now solved by saying, we want to use signals that are equally spaced. So that's an easy one. From the question of, how do you choose the filter. So the PAM modulation is going to go by taking a sequence of signals, mapping it into a

waveform, which is this expansion here. We're not assuming that these functions are orthogonal to each other. Although later, we will find out that they should be. But for now, p of t is just some arbitrary waveform. And we will try to figure out how to choose this waveform in such a way as to replace the sinc waveform with something which is better in terms of delay and almost as good in all other ways. We're not going to worry about the fact that p of t has to be realizable. Because with our arbitrary time reference at the receiver, it doesn't have to be realizable. It doesn't have to be causal.

So, p of t could be sinc of t over t. Which would give us a nice baseband limited function. But it could be anything else at all. As we said before, sinc t over t a dies out impractically slowly. And it requires infinite delay at the transmitter. You can't even send these signals with a sinc t over t signal. Because to send them, you have to start out at the beginning of this little bit of wiggling.

Now, you say, OK I'm going to truncate that when it's very small. And I don't worry about that. The point of what we're starting to look at now, though, is we're saying, OK, you truncate it, you do all these practical things. But it turns out that this problem of choosing this filter response has a very elegant and a very nice solution. And when we put noise in, it fits in perfectly with the idea of also choosing this filter in a particular way.

Now, we've talked about many problems here which were solved almost immediately after Shannon came out with his way of looking at communication in 1948. Guess when this problem was solved? It was solved 20 years earlier than that by a guy by the name of Nyquist, who was at Bell Labs back when Bell Labs meant something. I mean, in `28 it was a great place. It was a great place until seven or eight years ago. Nyquist is important in feedback theory. He's done some of the most important things there. His Nyquist criterion in dealing with how do you choose these filters to avoid intersymbol interference is fairly simple. But it's a very, very nice result. So we're going to talk about it. And then we will use that to say, ok, we don't have to worry about intersymbol interference any more, so all we have to worry about is noise. So we're getting rid of these problems one by one.

Our main problem is to choose this filter, so that we get some kind of reasonable compromise between time delay and bandwidth. That's -- that's basically the problem that we're facing. And Nyquist's solution to this was to say, forget about that also. Let's look at what set of filters work, and what set of filters don't work. And then you can take your choice among this set of folders that work.

So our problem is, how do you take the waveform u of t, which you receive, and find these samples out of it? Now, we already know that if you use sinc functions, all you have to do is sample this and you get these u sub k's back again. But if this thing is some absolutely wild waveform, maybe that's not what you get. So we say, OK, how do we retrieve these signals from the waveform that was transmitted and therefore from the waveform that was received. We're separating this from the noise question. Even if there's no noise at all, we still have a question and how do you find those input values directly from the function.

What we're going to do is to assume that the receiver filters u of t, with a linear time invariant filter, with impulse response q of t. Now, since we've said that p of t doesn't have to be causal, we might as well say that q of t doesn't have to be causal either. Because we've already thrown these details of delay to the winds.

So the filtered waveform, then is going to be r of t. Will be the integral of what was transmitted, convolved with q of t. So this is what you receive. And then what we're going to do is, we're going to sample this waveform. So Nyquist said, let's restrict ourselves to looking at receivers which first filter by some filter we're going to decide on, and then sample.

Why do you want to do that? Well, an interesting question. And Nyquist said, that's what we're going to do. And since Nyquist was famous, that's what we're going to do. One of the problems in the homework this week is to show that if you relax this a little bit and you say, well, I don't want to do what Nyquist said, what I want to do is to look at an arbitrary linear receiver which takes this received waveform, goes through any old linear operations that I want to go through, and solves for what -- and from that, tries to pick out these coefficients. And the question is, can you do

anything more than what Nyquist did. And the answer is, no.

If you look at it in another way, you will find that in fact, if you know what the signal constellation is, you can look at non-linear receivers. Which in the absence of noise will let you pick out these coefficients in a much broader context than what Nyquist said. And why don't we do that? Well, because when noise comes in, that doesn't work at all. That's a lousy solution.

So what we're going to do is say, OK, Mr. Nyquist, we'll play your silly game. We will have this filter at the transmitter. We'll have this filter at the receiver. We'll have the sampler and we'll look at what conditions we need in order to make the whole thing work. And then we will fit it in with noise and everything. It will fit in, in a very nice way. So we have a nice layered solution when we do that. And we will find -- I mean, Nyquist had some of Shannon's genes in him, I think. Because what we find when we're all finished with this is that by avoiding -- by being able to receive these coefficients perfectly, which we'll refer to as avoiding intersymbol interference, it doesn't hurt us at all as far as taking care of the noise. In other words, you can have your cake and you can eat it too as far as intersymbol interference and noise is concerned.

We wind up with a received waveform, which is the integral of the transmitted waveform times some filter. And we don't know how to choose this filter. But let's just -- this is a filter. This can be represented now in terms of u of t is equal to this transmitted waveform in terms of this other filter that we don't understand. So we now have two filters that we don't understand. And we have this integral here.

Well, we can take this sum. We can bring this sum outside of the integral and have the sum of u sub k times, just some composite filter g of t where g of t is the convolution of p of t and q of t.

Now, when you look at the notes, the notes are fairly careful in dealing with all these questions of L2 and convergence and all of this stuff that we've been talking about. Again, when you're trying to understand something for the first time, ignore all those mathematical issues. Try to find out what's going on. When you get an intuitive

sense of what's going on, go back and look at the mathematics then. But don't fuss about the mathematics at this point.

OK, so r of t, then, is just going to be the sum over k of these sample values that are coming in, times this composite filter, which is the convolution of the transmit filter and the recieve filter. Now, this shouldn't be surprising. If you think of u of t as being formed by taking a sequence of impulses, and then passing that sequence of impulses through a filter p of t, and then passing the output through a filter q of t, all you're doing is passing this sequence of impulses through the convolution of p of t and q of t.

In other words, in terms of this received waveform, it couldn't care less what filtering you do at the transmitter and what filtering you do at the receiver. It's all one big filter as far as the receiver is concerned. When we study noise, what happens with the transmitter and what happens with the receiver will become important again. But, so far, none of this makes any difference. And this is all we need to worry about.

Then, Nyquist said, a waveform g of t is ideal Nyquist. He didn't call it ideal Nyquist, he was very modest person. But since then, people call it ideal Nyquist because he was the guy that sorted it all out. It's ideal Nyquist with period t, and I usually leave out the period t because that's usually understood, if g of k t is equal to delta of k. In other words, if g of 0 is equal to 1 and g of k times t, for every non-zero integer k is equal to 0. In other words, if g has the same property that the sine function has, sine function is 1. It's 0. And it's 0 at every integer point beyond that. And Nyquist said, well, gee, all we need to do is make this filter has a property. And when you look at this, it's fairly obvious that that works, right? I mean, we want a sample RFT -- say, at j times capital T. Or, if j times capital T, it's going to be the sum here of u sub k times g of j t r of t. r of j t is equal to sum over k of u k times g of j t minus k t.

If the waveform is ideal Nyquist, then this quantity is 0 for all integer k except when k is equal to j. So, this is just equal to u sub j. And conversely, if this waveform is not ideal Nyquist, you have the problem that you can pick two values. u sub k and u sub

j, in such a way that they interfere with each other. In other words, that they add up at some sample point to the wrong value. One of them is going to come in and clobber the other.

So, this is a necessary and sufficient condition for avoiding intersymbol interference. So long as you're not smart enough to look at what those actual values are. In other words, so long as you're only going to use a linear receiver, which is what that amounts to.

While ignoring noise, r of t is determined by g of t, p of t and q of t otherwise irrelevant. That's what we said. We said this. If t of t is ideal Nyquist and r of k t equals u of k for all k and z. If f of t is not ideal Nyquist, and r f k t unequal to u k for some k and some choice of the sequence. Now, so far there's no big deal here. This is pretty easy to figure out. You don't need rocket science to say, once I pose the problem this why, which is where part of Nyquist's genius came from. The hard thing is always to post the right problem. It's not to solve it. Those of you who want to do Ph.D theses, believe me, 80% of the problem is finding the problem. 20% of the problem is doing it. If you do a really outstanding thesis, 99% is finding the problem and 1% of it is doing it. And I believe that. I'm not exaggerating.

An ideal Nyquist g of t implies that no intersymbol interference occurs at the above receiver. In other words, you have a receiver that actually works. We're going to see that choosing g of t to be ideal Nyquist fits in nicely when looking at the real problem, which is coping with both noise and intersymbol interference. And we've also seen that if g of t is sync of t over capital T, that works. It has no intersymbol interference because that's, one, at t equals 0, and it's 0 at every other sample point.

We don't like that, because it has too much delay. We want to make g of t strictly baseband limited to 1 over 2t. And this turns out to be the only solution. And we'll see that in a little while. In other words, if you want to do something which has better delay characteristics than the sinc function, your only way of doing it is spilling out into slightly higher frequencies. So, what Nyquist really wanted to find out, although

you won't find that out by reading his paper because it's all this nice mathematical proof, is how much do you have to expand the bandwidth in order to get nice delay things which do the same thing.

Well, we think about it a little bit. And we have an advantage that Nyquist didn't have. Because we understand about aliasing. Nyquist didn't understand about aliasing. Aliasing hadn't been done at that time. It was probably done as a result of what Nyquist had done. And if we look at the reconstruction from the samples, g of k t of g, we get this function s of t, which is the sum of all the samples times sinc of t over t, minus k. That's for an arbitrary filter. This baseband reconstruction by definition, when we were talking about alising, is just this function.

We've said that g of t is ideal Nyquist, if and only if s of t is equal to sinc of t over capital t. Why is that? You look at this function here. You look at s of 0. And what do you get? You get the sum of g of k t times sinc of t over t minus k. If we have an ideal Nyquist filter, then all of these terms are 0 except when k is equal to 0. s of t, in general, if you have a ideal Nyquist filter, you only have one term in here. So s of t is just equal to sinc of t over t. Because all those other terms go away.

If we take the Fourier transform of s of t equals sinc t over capital T, the Fourier transform is s s tilde of f is equal to t times the rectangle function of f times t. A sinc function of a Fourier transform is a rectangle. And the aliasing theorem then says that this Fourier transform of this low pass representation has to be equal to this sum of different frequency terms. That's what aliasing says. It says that this baseband representation is aliased into by all of these other frequency bands. And each of them come in and add to what you get in frequency here. Remember that diagram that we drew where we took this arbitrary frequency function and we picked up what was in each band, then we stuck it into the center band and then we added all these things up? That's what the aliasing thing says.

So it says that g of t is ideal Nyquist if and only if this sum is equal to that. And that's the Nyquist criteria. That's what Nyquist did. But he did it long before anybody had heard of aliasing.

There's a slightly easier way to look at aliasing criterion, which now becomes the Nyquist criterion. When what you're interested in is a waveform g of t, which is almost band-limited but not quite. So what we're going to assume is that it's limited to, at most, twice this -- this w here is 1 over 2t, which is called the Nyquist bandwidth. Everything is called Nyquist here, so. This value here is the minimum bandwidth you could be using. It's 1 over capital 2t. It's what you would get if you were using the sinc function. If you were using the sinc function, what you would get is something which is a rectangle here. Cut off, right at this point. And cut off right at this point. Nyquist is saying, suppose it's limited to at most 2w. In other words, suppose you have a slopover into other frequencies but, at most into the next frequency band and no more than that. Then, if you look at this thing, which is spilling out, and we take the same picture we were looking at before, we take this quantity. Bring it back down here. We take this quantity, bring it up here. And what do we get? This, added into here. This thing adds up right there.

In other words -- well, let's take this one here. This thing here gets translated over to here. And added to this. This is assuming that g hat of f is real, and we're ignoring the complex part of it. So this gets added to this. This is just enough to turn this into something, which goes across here and down here. Down here. My finger is not perfect. But, anyway, when you add this to this, you get this ideal rectangular shape.

What this is saying, in terms of just this upper side band here, this is going to be the same as this, from symmetry. So it's saying, if you take what's on the positive side of w, and you rotate it around this way, you rotate it around up here, if it's just enough to fill that in, then you've satisfied the Nyquist criteria. In other words, you want band edge symmetry here. You want this to be symmetrical to this. In that rotated 180 degree sense. So it says that anything which has this band edged symmetry condition satisfies the property of no intersymbol interference.

So this makes the problem easy for us. It says, we would like to have a rectangular function. That has too much delay, because in particular the inverse Fourier transform of that, because we have a discontinuity, can drop off, at most, as 1 over

t. Which is pretty slow. We've gotten rid of the discontinuity. We've also gotten rid of the slope discontinuity, the way I drew the figure here. And, therefore, we can wind up with a function which decays as 1 over t cubed. Which is a whole lot better than 1 over t.

And, excuse me for going a little over, but. The things people build in practice usually have something called a raised cosine filter. Which is this messy expression here, for the frequency response. But it really isn't that bad. In fact, it's just what this is. This frequency response is t over most of the frequency interval, up to 1 over 2t. It's t times a raised cosine over this part of the frequency band here. t times cosine squared, and the cosine squared just raises things up to be average out at 1/2, and it's 0 everywhere else.

So what you're doing here in that formula is simply making things t up to here. Making it a raised cosine here. And making it 0 everywhere else. And, depending on what do choose as alpha, that makes this sharper or less sharp. And people usually choose it to be about, alpha to be about 15% or so. Which means these filters chop off very, very rapidly.

Is that hard to build? Doesn't make any difference. I mean, these days, anything which you can figure out how to compute, you can put it on a little chip and it costs nothing if you make enough of them. So you want to raise the cosine filter which cuts off at 15%, fine. Somebody spends a year designing it. And then you cookie-cut it forever after. So it doesn't cost anything. And, well, g of t also has an inverse transform which we won't worry about.