PROFESSOR: In the last lecture, we discussed discrete time processing of continuous time signals. And, as you know, the basis for that arises essentially out of a sampling theorem. Now in that context, and also in its own right, another important sampling issue is the sampling of discrete time signals, in other words, the sampling of a sequence. One common context in which this arises, for example, is, if we've converted from a continuous time signal to a sequence, and we then carry out some additional filtering, then there's the possibility that we can resample that sequence, and as we'll see as we go through the discussion, save something in the way of storage or whatever.

So discrete time sampling, as I indicated, has important application in a context referred to here, namely resampling after discrete time filtering. And closely related to that, as we'll indicate in this lecture, is the concept of using discrete time sampling for what's referred to as sampling rate conversion. And also closely associated with both of those ideas is a set of ideas that I'll bring up in today's lecture, referred to as decimation and interpolation of discrete time signals or sequences.

Now the basic process for discrete time sampling is the same as it is for continuous time sampling. Namely, we can analyze it and set it up on the basis of multiplying or modulating a discrete time signal by an impulse train, the impulse train essentially, or pulse train, pulling out sequence values at the times that we want to sample.

So the basic block diagram for the sampling process is to modulate or multiply the sequence that we want to sample by an impulse train. And here, the impulse train has impulses spaced by integer multiples of capital N. This then becomes the sampling period. And the result of that modulation is then the sample sequence x of p of n.

So if we just look at what a sequence and a sampled version of that sequence might look like, what we have here is an original sequence x of n. And then we have the sampling impulse train, or sampling sequence, and it's the modulation or product of these two that gives us the sample sequence x of p of n. And so, as you can see, multiplying this by this essentially pulls out of the original sequence sample values at the times that this pulse train is on. And of course here, I've drawn this for the case where capital N, the sampling period, is equal to 3.

Now the analysis of discrete time sampling is very similar to the analysis of continuous time sampling. And let's just quickly look through the steps that are involved. We're modulating or multiplying in the time domain. And what that corresponds to in the frequency domain is a convolution. And so the spectrum of the sampled sequence is the periodic convolution of the spectrum of the sampling sequence and the spectrum of the sequence that we're sampling.

And since the sampling sequence is an impulse train, as we know, the Fourier transform of an impulse train is itself an impulse train. And so this is the Fourier transform of the sampling sequence. And now, finally, the Fourier transform of the resulting sample sequence, being the convolution of this with the Fourier transform of the sequence that we're sampling, gives us then a spectrum which consists of a sum of replicated versions of the Fourier transform of the sequence that we're sampling.

In other words, what we're doing, very much as we did in continuous time, is, through the sampling process when we look at it in the frequency domain, taking the spectrum of the sequence there were sampling and shifting it and then adding it in-- shifting it by integer multiples of the sampling frequency. In particular, looking back at this equation, what we recognize is that this term, k times 2 pi over capital N, is in fact an integer multiple of the sampling frequency. And the same thing is true here. This is k times omega sub s, where omega sub s, the sampling frequency, is 2 pi divided by capital N.

All right, so now let's look at what this means pictorially or graphically in the frequency domain. And as you can imagine, since the analysis and algebra is similar to what happens in continuous time, we would expect the pictures to more or less be identical to what we've seen previously for continuous time. And indeed that's the case.

So here we have the spectrum of the signal that's we're sampling. This is its Fourier transform, with an assumed highest frequency omega sub m, highest frequency over a 2pi range, or over a range of pi, rather. And now the spectrum of the sampling signal is what I show below, which is an impulse train with impulses occurring at integer multiples of the sampling frequency. And then finally, the convolution of these two is simply this one replicated at the locations of these impulses. And so that's finally what I show below.

And here I made one particular choice for the sampling period. This in particular corresponds to a sampling period which is capital N equal to 3. And so the sampling frequency, omega sub s, is 2pi divided by 3.

Now when we look at this, what we recognize is that we have basically the same issue here as we had in continuous time, in the sense that when these individual replications of the Fourier transform, when the sampling frequency is chosen high enough so that they don't overlap, then we see the potential for being able to get one of them back. On the other hand, when they do overlap then what we'll have is aliasing, in particular, discrete time aliasing, much as we had continuous time aliasing in the continuous time case.

Well notice in this picture that what we have is we've chosen this picture so that omega sub s minus omega sub m is greater than omega sub m, or equivalently, so that omega sub s is greater than 2 omega sub m. And so with omega sub s greater than 2 omega sum m, that corresponds to this picture. Whereas, if that condition is violated then, in fact, the picture that we would have is a picture that looks like.

And in this picture, the individual replications of the Fourier transform of the original signal overlap. And we can no longer recover the Fourier transform of the original signal. And this, just as it was in continuous time, is referred to as aliasing.

Now let's look more closely at the situation in which there is no aliasing. So in that case, what we have is a Fourier transform for the sampled signal, which is as I indicated here, and the Fourier transform for the original signal, as I indicate at the top. And the question now is how do we recover this one from this one.

Well, the way that we do that, just as we did in a continuous time case, is by a low pass filtering. In particular, processing in the time domain or in the frequency domain, this with an ideal low pass filter has the effect of extracting that part of the spectrum that in fact we identify with the original signal that we began with. So what we see, again, is that the process is very much the same. As long as there's no aliasing, we can recover the original signal by ideal low pass filtering.

So the overall system is, just to reiterate, a system which consists of modulating the original sequence with a pulse train or impulse train. And then that is going to be processed with a low pass filter. The spectrum of the original signal x of n is what I show here. The spectrum of the sampled signal, where I'm drawing the picture on the assumption that the sampling period is 3, is now what's indicated, where these are replicated, where the original spectrum is replicated.

This is now processed through a filter which, for exact reconstruction, is an ideal low pass filter. And so we would multiply this spectrum by this one. And the result, after doing that, will generate a reconstructed spectrum which, in fact, is identical to the original.

So the frequency domain picture is the same. And what we would expect then is that the time domain picture would be the same. Well, let's in fact look at the time domain. And in the time domain, what we have is an analysis more or less identical to what we had in continuous time. We of course have the same system.

And in the time domain, we are multiplying by an impulse train. Consequently, the sample sequence is an impulse train whose values are samples of x of at integer multiples of capital N. For the reconstruction, this is now processed through an ideal low pass filter. And that implements a convolution in the time domain.

And so the reconstructed signal is the convolution of the sample sequence and the filter impulse response. And expressed another way, namely writing out the convolution as a sum, we have this expression. And so it says then that the reconstruction is carried out by replacing the impulses here, these impulses, by versions of the filter impulse response.

Well, if the filter is an ideal low pass filter, then that corresponds in the time domain to sine nx over sine x kind of function. And that is the interpolation in between the samples to do the reconstruction.

Also, as is discussed somewhat in the text, we can consider other kinds of interpolation, for example discrete time zero order hold or discrete time first order hold, just as we had in continuous time. And the issues and analysis for the discrete times zero order hold and first order hold are very similar to what they were in continuous time-- the zero order hold just simply holding the value until the next sampling instant, and the first order hold carrying out linear interpolation in between the samples.

Now in this sampling process, if we look again at the wave forms involved, or sequences involved, the process consisted of taking a sequence and extracting from it individual values. And in between those values, we have sequence values equal to 0. So what we're doing in this case is retaining the same number of sequence values and simply setting some number of them equal to 0.

Well, let's say, for example, that we want to carry out sampling. And what we're talking about is a sequence. And let's say this sequence is stored in a computer memory. As you can imagine, the notion of sampling it and actually replacing some of the values by zero is somewhat inefficient. Namely, it doesn't make sense to think of storing in the memory a lot of zeros, when in fact those are zeros that we can always put back in. We know exactly what the values are. And if we know what the sampling rate was in discrete time, then we would know when and how to put the zeros back in.

So actually, in discrete time sampling, what we've talked about so far is really only one part or one step in the process. Basically, the other step is to take those zeros and just throw them away because we could put them in any time we want to and really only retain, for example in our computer memory or list of sequence values or whatever, only retain the non-zero values. So that process and the resulting sequence that we end up with is associated with a concept called decimation.

What I mean by decimation is very simple. What we're doing is, instead of working with this sequence, we're going to work with this sequence. Namely, we'll toss out the zeros in between here and collapse the sequence down only to the sequence values that are associated with the original x of n. Now, in talking about a decimated sequence, we could of course do that directly from this step down to here, although again in the analysis it will be somewhat more convenient to carry that out by thinking, at least analytically, in terms of a 2-step process-- one being a sampling process, then the other being a decimation. But basically, this is a decimated version of that.

Now for the grammatical purists out there, the word decimation of course means taking every tenth one. The implication is not that we're always sampling with a period of 10. The idea of decimating is to pick out every nth sample and end up with a collapsed sequence.

Let's now look at a little bit of the analysis and understand what the consequence is in the frequency domain. In particular what we want to develop is how the Fourier transform of the decimated sequence is related to the Fourier transform of the original sequence or the sample sequence. So let's look at this in the frequency domain.

So what we have is a decimated sequence, which consists of pulling out every capital Nth value of x of n. And of course that's the same as we can either decimate x of n or we can decimate the sample signal. Now in going through this analysis, I'll kind of go through it quickly because again there's the issue of some slight mental gymnastics. And if you're anything like I am, it's usually best to kind of try to absorb that by yourself quietly, rather than having somebody throw it at you.

Let me say, though, that the steps that I'm following here are slightly different than the steps that I use in the text. It's a slightly different way of going through the analysis. I guess you could say for one thing that if we've gone through it twice, and it comes out the same, well of course it has to be right.

Well anyway, here we have then the relationship between the decimated sequence, the original sequence, and the sampled sequence. And we know of course that the Fourier transform of the sample sequence is just simply this summation. And now kind of the idea in the analysis is that we can collapse this summation by recognizing that this term is only non-zero at every nth value.

And so if we do that, essentially making a substitution of variables with n equal to small m times capital N, we can turn this into a summation on m. And that's what I've done here. And we've just simply used the fact that we can collapse the sum because of the fact that all but every nth value is equal to zero.

So this then is the Fourier transform all of the sampled signal. And now if we look at the Fourier transform of the decimated signal, that Fourier transform, of course, is this summation on the decimated sequence. Well, what we want to look at is the correspondence between this equation and the one above it. So we want to compare this equation to this one.

And recognizing that this decimated sequence is just simply related to the sample sequence this way, these two become equal under a substitution of variables. In particular, notice that if we replace in this equation omega by omega times capital N, then these two equations become equal. So the consequence of that, then, what it all boils down to and says, is that the relationship between the Fourier transform of the decimated sequence and the Fourier transform of the sampled sequence is simply a frequency scaling corresponding to dividing the frequency axis by capital N.

So that's essentially what happens. That's really all that's involved in the decimation process. And now, again, let's look at that pictorially and see what it means.

So what we want to look at, now that we've looked in the time domain in this particular view graph, we now want to look in the frequency domain. And in the frequency domain, we have, again, the Fourier transform of the original sequence and we have the Fourier transform of the sampled sequence. And now the Fourier transform of the decimated sequence is simply this spectrum with a linear frequency scaling.

And in particular, it simply corresponds to multiplying this frequency axis by capital N. And notice that this frequency now, 2 pi over capital N, that frequency ends up getting rescaled to a frequency of 2 pi. So in fact now, in the rescaling, it's that this point in the decimation gets rescaled to this point. And correspondingly, of course, this whole spectrum broadens out.

Now we can also look at that in the context of the original spectrum. And you can see that the relationship between the original spectrum and the spectrum of the decimated signal corresponds to simply linearly scaling this. But it's important also to keep in mind that that analysis, that particular relationship, assumes that we've avoided aliasing.

The relationship between the spectrum of the decimated signal and the spectrum of the sample signal is true whether or not we have aliasing. But being able to clearly associate it with just simply scaling of this spectrum of the original signal assumes that the spectrum of the original signal, the shape of it, is preserved when we generate the sample signal.

Well, when might discrete time sampling, and for that matter, decimation, be used? Well, I indicated one context in which it might be useful at the beginning of this lecture. And let me now focus in on that a little more specifically.

In particular, suppose that we have gone through a process in which the continuous time signal has been converted to a discrete time signal. And we then carry out some additional discrete time filtering. So we have a situation where we've gone through a continuous to discrete time conversion. And after that conversion, we carry out some discrete time filtering. And in particular, in going through this part of the process, we choose the sampling rate for going from the continuous time signal to the sequence so that we don't violate the sampling theorem.

Well let's suppose, then, that this is the spectrum of the continuous time signal. Below it, we have the spectrum of the output of the continuous to discrete time conversion. And I've chosen the sampling frequency to be just high enough so that I avoid aliasing. Well that then is the lowest sampling frequency I can pick.

But now, if we go through some additional low pass filtering, then let's see what happens. If I now low pass filter the sequence x of n, then in effect, I'm multiplying the sequence spectrum by this filter. And so the result of that, the product of the filter frequency response and the Fourier transform of x of n would have a shape somewhat like I indicate below.

Now notice that in this spectrum, although in the input to the filter this entire band was filled up, in the output of the filter, there is a band that in fact has zero energy in it. So what I can consider doing is taking the output sequence from the filter and in fact resampling it, in other words sampling it, which would be more or less associated with a different sampling rate for the continuous time signals involved.

So I could now go through a process which is commonly referred to as down sampling that is lowering the sampling rate. When we do that, of course, what's going to happen is that in fact this spectral energy will now fill out more of the band. And for example, if this was a third, then in fact if I down sampled by a factor of three, then I would fill up the entire band with this energy. But since I've done some additional low pass filtering, as I indicate here, there's no problem with aliasing.

If I had, let's say, down sampled by a factor of three and I'm now taking that signal and converting it back to a continuous time signal, then of course the way I can do that is by simply running my output clock for the discrete to continuous time converter. I can run my output clock at a third the rate of the input clock. And that, in effect, takes care of the bookkeeping for me.

So here we have now the notion of sampling a sequence, and very closely tied in with that, the notion of decimating a sequence, and related to both of those, the notion of down sampling, that is changing the sampling rates so that, if we were trying this in with continuous time signals, we've essentially changed our clock rate. And we might also want to, and it's important to, consider the opposite of that.

So now a question is what's the opposite of decimation. Suppose that we had a sequence and we decimate it. Thinking about it as a 2-step process, that would correspond to first multiplying by an impulse train, where there are bunch of zeros in there, and then choosing, throwing away the zeros and keeping only the values that are non-zero, because the zeros we can always recreate. Well, in fact, the inverse process is very specifically a process of recreating the zeros and then doing the desampling.

So in the opposite operation, what we would do is undo the decimation step. And that would consist of converting the decimated sequence back to an impulse train and then processing that impulse train by an ideal low pass filter to do the interpolation or reconstruction, filling in the values which, in this impulse train, are equal to zero. So we now have the two steps. We take the decimated sequence and we expand it out, putting in zeros. And then we desample that by processing it through a low pass filter.

So just kind of looking at sequences again, what we have is an original sequence, the sequence x of n. And then the sample sequence is simply a sequence which alternates, in this particular case, those sequence values was zero. Here what we're assuming is that the sampling period is 2. And so every other value here is equal to zero. The decimated sequence then is this sequence, collapsed as I show in the sequence above. And so it's, in effect, time compressing the sample sequence or the original sequence so that we throw out the sequence values which were equal to zero in the sample sequence.

Now in recovering the original sequence from the decimated sequence, we can think of a 2-step process. Namely, we spread this out alternating with zeros, and again, keeping in mind that this is drawn for the case where capital N is 2. And then finally, we interpolate between the non-zero values here by going through a low pass filter to reconstruct the original sequence. And that's what we show finally on the bottom curve. So that's what we would see in the time domain.

Let's look at what we would see in the frequency domain. In the frequency domain, we have to begin with the sequence on the bottom, or the spectrum on the bottom, which would correspond to the original spectrum. Then, through the sampling process, that is periodically replicated. Again, this is drawn on the assumption that the sampling frequency is pi or the sampling period is equal to 2. And so this is now replicated.

And then, in going from this to the spectrum of the decimated sequence, we would rescale the frequency axis so that the frequency pi now gets rescaled in the spectrum for the decimated sequence to a frequency which is 2 pi. And so this now is the spectrum of the decimated sequence.

If we now want to reconvert to the original sequence we would first intersperse in the time domain with zeros, corresponding to compressing in the frequency domain. This would then be low pass filtered. And the low pass filtering would consist of throwing away this replication, accounting for a factor which is the factor capital N, and extracting the portion of the spectrum which is associated with the spectrum of the original signal which we began with.

So once again, we have decimation and interpolation. And the decimation can be thought of as a time compression that corresponds to a frequency expansion then. And the interpolation process is then just the reverse.

Now there are lots of situations in which decimation and interpolation and discrete time sampling are useful. And one context that I just want to quickly draw your attention to is the use of decimation and interpolation in what is commonly referred to as sampling rate conversion. What the basic issue and sampling rate conversion is is that, in some situations, and a very common one is digital audio, a continuous time signal is sampled. And those sampled values are stored or whatever.

And kind of the notion is that, perhaps when that is played back, it's played back through a different system. And the different system has a different assumed sampling frequency or sampling period. So that's kind of the issue and the idea.

We have, let's say, a continuous time signal which we've converted to a sequence through a sampling process using an assumed sampling period of T1. And these sequence values may then, for example, be put into digital storage. In the case of a digital audio system, it may, for example, go onto a digital record. And it might be the output of this that we want to recreate. Or we might in fact follow that with some additional processing, whatever that additional processing is. And I'll kind of put a question mark in there because we don't know exactly what that might be.

And then, in any case, the result of that is going to be converted back to a continuous time signal. But it might be converted through a system that has a different assumed sampling period. And so a very common issue, and it comes up as I indicated particularly in digital audio, a very common issue is to be able to convert from one assumed sampling period, T1, our sampling frequency, to another assumed sampling period.

Now how do we do that? Well in fact, we do that by using the ideas of decimation and interpolation. In particular, if we had, for example, a situation where we wanted to convert from a sampling period, T1, to a sampling period which was twice as long as that, then essentially, we're going to take the sequence and process it in a way that would, in effect, correspond to assuming that we had sampled at half the original frequency.

Well how do we do that? The way we do it is we take the sequence we have and we just throw away every other value. So in that case, we would then, for this sampling rate conversion, down sample and decimate. Or actually, we might not go through this step formally. We might just simply decimate.

Now we might have an alternative situation where in fact the new sampling period, or the sampling period of the output, is half the sampling period of the input, corresponding to an assumed sampling frequency, which is twice as high. And in that case, then., we would go through a process of interpolation. And in particular, we would up sample and interpolate by a factor of 2 to one.

So in one case, we're simply throwing away every other value. In the other case, what we're going to do is take our sequence, put in zeros, put it through a low pass filter to interpolate. Now life would be simple if everything happened in simple integer amounts like that.

A more common situation is that we may have an assumed output sampling period which is 3/2 of the input sampling period. And now the question is what are we going to do to convert from this sampling period to this sampling period. Well, in fact, the answer to that is to use a combination of down sampling and up sampling, or up sampling and down sampling, equivalently interpolation and decimation.

And for this particular case, in fact, what we would do is to first take the data, up sample by a factor of 2, and then down sample the result of that by a factor of 3. And what that would give us is a sampling rate conversion, overall, of 3/2, or a sampling period conversion of 3/2.

And more generally, what you could think of is how you might do this if, in general, the relationship between the input and output sampling periods was some rational number p/q. And so in fact, in many systems, in hardware systems related to digital audio, very often the sampling rate conversion, most typically the sampling rate conversion, is done through a process of up sampling or interpolating and then down sampling by some other amount.

Now what we've seen, what we talked about in a set of lectures, is the concepts of sampling a signal. And what we've seen is that the signal can be represented by samples under certain conditions. And the sampling that we've been talking about is sampling in the time domain. And we've done that for continuous time and we've done it for discrete time.

Now we know that there is some type of duality both continuous time and discrete time, some type of duality, between the time domain and frequency domain. And so, as you can imagine, we can also talk about sampling in the frequency domain and expect that, more or less, the kinds of properties and analysis will be similar to those related to sampling in the time domain.

Well I want to talk just briefly about that and leave the more detailed discussion to the text and video course manual. But let me indicate, for example, one context in which frequency domain sampling is important. Suppose that you have a signal and what you'd like to measure is its Fourier transform, its spectrum.

Well of course, if you want to measure it or calculate it, you can never do that exactly at every single frequency. There are too many frequencies, namely, an infinite number of them. And so, in fact, all that you can really calculate or measure is the Fourier transform at a set of sample frequencies.

So essentially, if you are going to look at a spectrum, continuous time or discrete time, you can only really look at samples. And a reasonable question to ask, then, is when does a set of samples in fact tell you everything that there is to know about the Fourier transform. That, and the answer to that, is very closely related to the concept of frequency domain sampling.

Well, frequency domain sampling, just to kind of introduce the topic, corresponds and can be analyzed in terms doing modulation in the frequency domain, very much like the modulation that we carried out in the time domain for time domain sampling. And so we would multiply the Fourier transform of the signal whose spectrum is to be sampled by an impulse train in frequency. And so shown below is what might be a representative spectrum for the input signal.

And the spectrum, then for the signal associated with the frequency domain sampling, consists of multiplying the frequency domain by this impulse train. Or correspondingly, the Fourier transform of the resulting signal is an impulse train in frequency with an envelope which is the original spectrum that we were sampling.

Well, this of course is what we would do in the frequency domain. It's modulation by an impulse train. What does this mean in the time domain?

Well, let's see. Multiplication in the time domain is convolution in the frequency domain. Convolution in the frequency domain is multiplication-- I'm sorry. Multiplication in the frequency domain, then, is convolution in the time domain. And in fact, the process in the time domain is a convolution process. Namely, the time domain signal is replicated at integer amounts of a particular time associated with the spacing in frequency under which we're doing the frequency domain sampling.

So in fact, if we look at this in the time domain, the resulting picture corresponds to an original signal whose spectrum or Fourier transform we've sampled. And a consequence of the sampling is that the associated time domain signal is just like the original signal, but periodically replicated, in time now, not frequency, but in time, at integer multiples of 2 pi divided by the spectral sampling interval omega 0. And so this then is the time function associated with the sample frequency function.

Now, that's not surprising because what we've done is generated an impulse train and frequency with a certain envelope. We know that an impulse train in frequency is the Fourier transform of a periodic time function. And so in fact, we have a periodic time function. We also know that the envelope of those impulses-- we know this from way back when we talked about Fourier transforms-- the envelope, in fact, is the Fourier transform of one period. And so all of this, of course, fits together as it should in a consistent way.

Now given that we have this periodic time function whose Fourier transform is the samples in the frequency domain, how do we get back the original time function? Well, with time domain sampling, what we did was to multiply in the frequency domain by a gate, or window, to extract that part of the spectrum. What we do here is exactly the same thing, namely multiply in the time domain by a time window which extracts just one period of this periodic signal, which would then give us back the original signal that we started with.

Now also let's keep in mind, going back to this time function and the relationship between them, then again, there is the potential, if this time function is too long in relation to 2 pi divided by omega 0, there's the potential for these to overlap. And so what this means is that, in fact, what we can end up with, if the sample spacing and the frequency is not small enough, what we can end up with is an overlap in the replication in the time domain. And what that corresponds to and what it's called is, in fact, time aliasing. So we can have time aliasing with frequency domain sampling just as we can have frequency aliasing with time domain sampling.

Finally, let me just indicate very quickly that, although we're not going through this in any detail, the same basic idea applies in discrete time. Namely, if we have a discrete time signal and if the discrete time signal is a finite length, if we sample its Fourier transform, the time function associated with those samples is a periodic replication. And we can now extract, from this periodic signal, the original signal by multiplying by an appropriate time window, the product of that giving us the reconstructed time function as I indicate below.

So we've now seen a little bit of the notion of frequency domain sampling, as well as time domain sampling. And let me stress that, although I haven't gone into this in a lot of detail, it's important. It's used very often. It's naturally important to understand it. But, in fact, there is so much duality between the time domain and frequency domain, that a thorough understanding of time domain sampling just naturally leads to a thorough understanding of frequency domain sampling.

Now we've talked a lot about sampling. And this now concludes our discussion of sampling. I've stressed many times in the lectures associated with this that sampling is a very important topic in the context of our whole discussion, in part because it forms such an important bridge between continuous time and discrete time ideas. And your picture now should kind of be a global one that sees how continuous time and discrete time fit together, not just analytically, but also practically.

Beginning in the next lecture, what I will introduce is the Laplace transform and, beyond that, the Z transform. And what those will correspond to are generalizations of the Fourier transform. So we now want to turn our attention back to some analytical tools, in particular developing some generalizations of the Fourier transform in both continuous time and discrete time. And what we'll see is that those generalizations provide us with considerably enhanced flexibility in dealing with and analyzing both signals and linear time invariant systems. Thank you.