

The following content is provided under a Creative Commons license. Your support will help MIT OpenCourseWare continue to offer high quality educational resources for free. To make a donation or view additional materials from hundreds of MIT courses, visit MIT OpenCourseWare at [ocw.mit.edu](http://ocw.mit.edu).

**PROFESSOR:** Over the past several lectures, we've developed a representation for linear time-invariant systems. A particularly important set of systems, which are linear and time-invariant, are those that are represented by linear constant-coefficient differential equations in continuous time or linear constant-coefficient difference equations in discrete time.

For example, electrical circuits that are built, let's say, out of resistors, inductors, and capacitors, perhaps with op-amps, correspond to systems described by differential equations. Mechanical systems with springs and dashpots, likewise, are described by differential equations. And in the discrete-time case, things such as moving average filters, digital filters, and most simple kinds of data smoothing are all linear constant-coefficient difference equations.

Now, presumably in a previous course, you've had some exposure to differential equations for continuous time, and their solution using notions like particular solution, and homogeneous solution, initial conditions, et cetera. Later on in the course, when we've developed the concept of the Fourier transform after that, the Laplace transform, we'll see some very efficient and useful ways of generating solutions, both for differential and difference equations.

At this point, however, I'd like to just introduce linear constant-coefficient differential equations and their discrete-time counterpart. And address, among other things, the issue of when they do and don't correspond to linear time-invariant systems.

Well, let's first consider what I refer to as an  $n$ th-order linear constant-coefficient differential equation, as I've indicated here. And what it consists of is a linear combination of derivatives of the system output,  $y(t)$ , equal to a linear combination

of derivatives of the system input  $x(t)$ .

And it's referred to as a constant-coefficient equation, of course, because the coefficients are constant. In other words, not assumed to be time-varying. And it's referred to as linear because it corresponds to a linear combination of these derivatives, not because it corresponds to a linear system. And, in fact, as we'll see, or as I'll indicate, this equation may or may not, in fact, correspond to a linear system.

In the discrete-time case, the corresponding equation is a linear constant-coefficient difference equation. And that corresponds to, again, a linear combination of delayed versions of the output equal to a linear combination of delayed versions of the input. This equation is referred to an  $n$ th-order difference equation. The  $n$  referring to the number of delays of the output involved, just as an  $N$ th-order differential equation, the  $n$  or the order of the equation refers to the number of derivatives of the output.

Now, let's first begin with linear constant-coefficient differential equations. And the basic point of the solution for the differential equations is the fact that if we've generated some solution, which I refer to here as  $y_p(t)$ , some solution to the equation for a given input, then, in fact, we can add to that solution any other solution which satisfies what's referred to as the homogeneous equation.

So in fact, this differential equation by itself is not a unique specification of the system. If I have any solution, then I can add to that solution any other solution which satisfies the homogeneous equation, and the sum of those two will likewise be a solution. And that's very straightforward to verify. By simply substituting into the differential equation, the sum of a particular and the homogeneous solution, and what you'll see is that the homogeneous contribution, in fact, goes to zero by definition of what we mean by the homogeneous equation.

Now, the homogeneous solution for a linear constant-coefficient differential equation is of the form that I indicate at the bottom. And it typically consists of a sum of  $N$  complex exponentials. And the constants are undetermined by the equation itself. And this form for the homogeneous solution, in essence, drops out of examining the

homogeneous equation, where if we assume that the form of the homogeneous solution is a complex exponential with some unspecified amplitude and unspecified exponent.

If we substitute this into our homogeneous equation, we end up with the equation that I've indicated here. The factor  $A$  and the  $e^{st}$  can in fact be canceled out, and we find that that equation is satisfied for  $N$  values of  $s$ . And that's true no matter what choice is made for these coefficients. And the essential consequence of all of that is that the homogeneous solution is of the form that I indicated previously. Namely it consists of a sum of  $N$  complex exponentials, where the coefficients, the  $N$  coefficients, attached to each of those complex exponentials is undetermined or unspecified.

So what this says is that in order to obtain the solution for a linear constant-coefficient differential equation, we need some kind of auxiliary information that tells us how to obtain these  $N$  undetermined constants. And there are a variety of ways of specifying this auxiliary information or auxiliary conditions.

For example, in addition to the differential equation, what I can tell you is the value of the output and  $N - 1$  of its derivatives, at some specified time,  $t_0$ . And so the differential equation together with the auxiliary information, the initial conditions, then lets you determine the total solution, which namely lets you determine these previously-unspecified coefficients in the homogeneous solution.

Now, depending on how the auxiliary information is stated or what auxiliary information is available, the system may or may not correspond to a linear system, and may or may not correspond to a linear time-invariant system. One essential condition for it to correspond to a linear system is that the initial conditions must be 0.

And one can see the reason for that, if we refer back to the previous lecture in which we saw that for a linear system, if we put 0 in, we get 0 out. So if  $x(t)$ , the input, is 0, the output must be 0. And so that, in essence, tells us that, at least for the system to be linear, these initial conditions must be 0.

Now, beyond that, if we want the system to be causal and linear and time-invariant, then what's required on the initial conditions is that they be consistent with what's referred to as *initial rest*. Initial rest says that the output must be 0 up until the time that the input becomes non-zero. And we can see, of course, that that's consistent with the notion of causality, as we talked about in the previous lecture.

And it's relatively straightforward to see that if the system is causal and linear and time-invariant, that will require initial rest. It's somewhat more difficult to see that if we specify initial rest, then that, in fact, is sufficient to determine that the system is both causal and linear and time invariant. But the essential point then is that it requires initial rest for both linearity and causality.

OK, well, let's look at an example, and let's take the example of a first-order differential equation, as I've indicated here. So we have a first-order differential equation,  $dy(t) / dt + ay(t)$  is the input  $x(t)$ . And let's first look at what the homogeneous solution of this equation is. And so, we consider the homogeneous equation, namely the equation that specifies solutions, which would correspond to 0 input.

We, in essence, guess, or impose a solution of the form. The homogeneous solution is an amplitude factor times a complex exponential. Substituting this into the homogeneous equation, we then get the equation that I've indicated here. What you can see is that in this equation, I can cancel out the amplitude factor and this complex exponential.

So let's just cancel those out on both sides of the equation. And what we're left with is an equation that specifies what the complex exponent must be. In particular, for the homogeneous solution,  $s$  must be equal to  $-a$ , and so finally, our homogeneous solution is as I've indicated here.

Now, let's look at the solution for a specific input. Let's consider, for example, an input which is a scaled unit step. And although I won't work out the solution in detail and perhaps using what you've worked on previously, you know how to carry out

the solution for that. A solution with a step input is what I've indicated here: a scalar,  $1$  minus an exponential, times a unit step. And you can verify that simply by substituting into the differential equation.

Now, we know that there's a family of solutions. In other words, any solution with a homogeneous solution added to it, is, again, a solution. And so if we consider the solution that I just indicated, we generate the entire family of solutions by adding a homogeneous solution to it, and so this then corresponds to the entire family of solutions, where the constant  $A$  is unspecified so far, and needs to be specified through some type of auxiliary conditions.

Now, a class of auxiliary conditions is the condition of initial rest, which as I indicated before, is equivalent to the statement that the system is causal and linear and time-invariant. And in that case, for the initial rest condition, we would then require in this equation above that this constant be equal to  $0$ . And so finally, the response to a scaled step-- if the system is to correspond to a causal linear time-invariant system, is then just this term, namely, a constant times  $1$  minus an exponential, times the step.

Now, if the system is a linear time-invariant system, it can, as we know, be described through its impulse response. And as you've worked out previously in the video course manual, for a linear time-invariant system, the impulse response is the derivative of the step response. And just to quickly remind you of where that result comes from.

In essence, we can consider two linear time-invariant systems in cascade, one a differentiator, the other the system that we're talking about described by the differential equation. And a step in here then generates an impulse into our system, and out comes the impulse response. Well, just using the fact that these are both linear time-invariant systems, and they can be cascaded in either order, then means that if we have the step response to our system, and that goes through the differentiator, what must come out, again, is the impulse response.

So differentiating the step response, we get the impulse response. Here again, is

the step response as we just worked it out, this time for a unit step. If we differentiate, we have then, since the step response is the product of two terms, the derivative of a product is the sum of the derivatives. And carrying that algebra through, and using the fact that the derivative of the step is an impulse, finally we come down to this statement for the impulse response.

And then recognizing that this is a time function times an impulse. And we know that if a time function times an impulse takes on the value at the time that the impulse occurs, then this term is simply 0. And the impulse response then finally is an exponential of this form. And this is the decaying exponential for  $a$ -positive, it's a growing exponential for  $a$ -negative.

And recall that, as we talked about previously, a linear time-invariant system is stable if its impulse response is absolutely integrable. For this particular case, this impulse response is absolutely integrable provided that the exponential factor  $a$  is greater than 0.

Okay, so what we've seen then is the impulse response for a system described by a linear constant-coefficient differential equation, where in addition, we would impose causality, linearity, and time-invariance, essentially, through the initial conditions of initial rest.

Now, pretty much the same kinds of things happen with difference equations as we've gone through with differential equations. In particular, again, let me remind you of the form of an  $N$ th order linear constant coefficient difference equation. It's as I indicate here. And, again, a linear combination, this time of delayed versions of the output equal to a linear combination of delayed versions of the input.

Once again, difference equation is not a complete specification of the system because we can add to the response any homogeneous solution. In other words, any solution that satisfies the homogeneous equation and the sum of those will also satisfy the original difference equation. So if we have a particular response that satisfies the difference equation, then adding to that any response that is a solution to the homogeneous equation will also be a solution to the total equation.

The homogeneous solution, again, is of the form of a linear combination of exponentials. Here, we have the homogeneous equation. As with differential equations, we can guess or impose solutions of the form  $A$  times an exponential. When we substitute this into the homogeneous equation, we then end up with the equation that I've indicated here. We recognize again that  $A$ , the amplitude, and  $z^n$ , this exponential factor, cancel out.

And so this equation is satisfied for any values of  $z$  that satisfy this equation. And there are  $N$  roots,  $z_1$  through  $z_N$ . And so finally, the form for the homogeneous solution is a linear combination of capital  $N$  exponentials, where capital  $N$  is the order of the equation. With each of those exponentials, the amplitude factor is undetermined and needs to be determined in some way through the imposition of appropriate initial conditions or boundary conditions.

So the general form then for the solution to the difference equation is a sum of exponentials plus any particular solution. It's through auxiliary conditions that we determine these coefficients. We have  $N$  undetermined coefficients and, so we require  $N$  auxiliary conditions.

For example, some set of values of the output at  $N$  distinct instance of time. Now this was the same as with differential equations. In the case of differential equations, we talked about specifying the value of the output and its derivatives. And there, we indicated that for linearity, what we required-- for linearity, what we required is that the auxiliary conditions be 0. And the same thing applies here for the same reason. Namely, if the system is to be linear, then the response, if there's no input, must be equal to 0.

In addition, what we may want to impose on the system is that it be causal, and in addition to linear time-invariant, and what that requires, again, is that the auxiliary conditions be consistent with initial rest, namely, that if the input is 0 prior to some time, then the output is 0 prior to the same time.

So we've seen a very direct parallel so far between differential equations and

difference equations. In fact, one difference between them that, in some sense, makes difference equations easier to deal with in some situations, is that in contrast to a differential equation, a difference equation, if we assume causality, in fact is an explicit input-output relationship for the system.

Now, let me show you what I mean. Let's consider the  $n$ th-order difference equation, as I've indicated here. And let's assume that we're imposing causality so that the output can only depend on prior values of the input, and therefore, aren't prior values of the output. Well, we can simply rearrange this equation solving for  $y[n]$  the leading term, with  $k = 0$ . Taking all of the other terms over to the right side of the equation, and we then have a recursive equation, namely, an equation that expresses the output in terms of prior values of the input, which is this term, and prior values of the output.

And so if, in fact, we have this equation running, then once it started, we know how to compute the output for the next time instant. Well, how do we get it started? The way we get it started, of course, is through the appropriate set of initial conditions or boundary conditions. And, if for example, we assume initial rest corresponding to a causal linear time-invariant system, then if the input is 0 up until some time, the output must be 0 up until that time. And that, in essence, helps us get the equation started.

Well, let's look at this specifically in the context of a first-order difference equation. So let's take a first-order difference equation, as I've indicated here. And so, we have an equation that tells us that  $y[n] - ay[n-1] = x[n]$ . Now, if we want this to correspond to a causal linear time-invariant system, we impose initial rest on it. We can rewrite the first-order difference equation by taking the term involving  $y[n-1]$  over to the right hand side of the equation.

Now, this gives us a recursive equation that expresses the output in terms of the input and past values of the output. And since we've imposed causality, and if we're talking about a linear time-invariant system, we can now inquire as to what the impulse response is. And we know, of course, the impulse response tells us

everything that we need to know about the system.

So let's choose an input, which is an impulse. So the impulse response is  $\delta[n]$  corresponding to the  $x[n]$  up here, plus a delta of-- this should be  $h[n-1]$ . And let me just correct that. This is  $h[n-1]$ .

So we have the impulse response as  $\delta[n]$  plus  $a$  times  $h[n-1]$ . Now, from initial rest, we know that since the input, namely an impulse, is 0 for  $n$  less than 0, the impulse response is likewise 0 for  $n$  less than 0.

And now, let's work out what  $h[0]$  is. Well,  $h$  of 0, with  $n = 0$ , is  $\delta[0]$  plus  $a$  times  $h[n-1]$ .  $h[n-1]$  is 0. And so,  $h[0]$  is equal to 1. Now that we have  $h[0]$ , we can figure out  $h[1]$  by running this recursive equation. So  $h[1]$  is  $\delta[1]$ , which is 0, plus  $a$  times  $h[0]$ , which we just figured out is 1.

So,  $h[1]$  is equal to  $a$ . And if we carry this through, we'll have  $h[2]$  equal to  $a^2$ , and this will continue on. And in fact, what we can recognize by looking at this and how we would expect these terms to build, we would see that the impulse response,  $h[n]$ , in fact, is of the form  $a^n$  times  $u[n]$ .

And we also can recognize then that this corresponds to a stable system, if and only if the impulse response, which is what we just figured out, if and only if the impulse response is absolutely summable. And what that will require is that the magnitude of  $a$  be less than 1. Now, we imposed causality, linearity, and time-invariance, and generated a solution recursively. And now, of course, if we want to generate the more general set of solutions to this difference equation, we can do that by adding all of the homogeneous solutions, namely, all the solutions that satisfy the homogeneous equation.

So here, we have the causal linear time-invariant impulse response. In fact, with an impulse input, all of the possible solutions are that impulse response plus the homogeneous solutions. The homogeneous solution is the solution that satisfies the homogeneous equation. That will, in general, be of the form an amplitude factor times an exponential factor, and if we substitute this into this equation, then we see

that the homogeneous equation is satisfied for any values of  $a$  and any values of  $z$  that satisfy this equation.

Again, as we did with differential equations, the factor  $A$  cancels out. And also, in fact, I can cancel out a  $z^n$  there and a  $z^n$  here. And so what we're left with is a statement that tells us then that the homogeneous solution is of the form  $a(z^n)$ , for any value of  $A$  and any value of  $z$  that satisfies this equation. And that value of  $z$ , in particular, is  $z$  equal to  $a$ .

So the homogeneous solution then is any exponential of this form with any amplitude factor. And so, the family of solutions, with an impulse input is the solution corresponding to the system being causal, linear, and time-invariant, plus the homogeneous term. If we impose causality, linearity, and time-invariance on the system, then of course, that additional exponential factor will be 0. In other words,  $A$  is equal to 0.

Now, we've seen the differential equations and difference equations in terms of the fact that there are families of solutions, and in order to get causality, linearity, and time-invariance requires imposing a particular set of initial conditions, namely, imposing initial rest on the system.

Let's now look at the difference equation and then later, the differential equation, interpreted in block diagram terms. Now, the difference equation, as I just simply repeated here, is  $y[n] = x[n] + ay[n-1]$ , where I've taken the delayed term over to the right hand side of the equation. So, in effect, what I'm imposing on this system is causality. I'm assuming that if I know the past history of the input and the output, I can determine the next value of the output.

Well, we can, in fact, draw a block diagram that represents that equation. The equations says, we take  $x[n]$  for any given value of  $n$ , say  $n_0$ , whatever value we're computing the output for. We take the input at that time, and add to it the factor  $a$  times the output value that we calculated last time.

So if we have  $x[n]$ , which is our input, and if we have  $y[n]$ , which is our output, we, in

fact, can get  $y$  of  $n$  by taking the last value of  $y[n]$ , indicated here by putting  $y[n]$  through a delay, multiplying that by the factor  $a$ , and then adding that result to the input.

And the result of doing that is  $y[n]$ . So the way this block diagram might be interpreted, for example, as an algorithm is to say that we take  $x[n]$ , add to it  $a$  times the previous value the output. That sum gives us the current value of the output, which we then put out of the system, and also put into a delay element, or basically, into a storage register, to use on the next iteration or recursion.

Now, how do we get this started? Well, we know that the difference equation requires initial conditions. And, in fact, the initial conditions correspond to what we store in the delay register when this block diagram or equation initially starts up. OK Now, let's look at this for the case of difference equations more generally.

So, what we've said is that we can calculate the output by having previous values of the input, previous values of the output, and forming the appropriate linear combination. So let's just build up the more general block diagram that would correspond to this. And what it says is that we want to have a mechanism for storing past values of the input, and a mechanism for storing past values of the output. And I've indicated that on this figure, so far, by a chain of delay elements, indicating that what the output of each delay is, is the input delayed by one time instant or interval.

And so what we see down this chain of delays are delayed replications of the input. And what we see on the other chain is delayed replications of the output. Now, the difference equation says that we want to take these, and multiply them by the appropriate coefficients, the coefficients in the difference equation, and so, we can do that as I've indicated here. So now, we have these delay elements, each multiplied by the appropriate coefficients on the input, and by appropriate coefficients on the output.

Those are then summed together, and so we now will sum these and will sum these. After we've summed these, we want to add those together. And there's a factor of  $1/a_0$  that comes in. And so that then generates our output. And so this,

in fact, then represents a block diagram, which is a general block diagram for implementing or representing a linear constant-coefficient difference equation.

Now, if you think about what it means in terms of, let's say, a computer algorithm or a piece of hardware, in fact, this block diagram is a recipe or algorithm for doing the implementation. But it's important to recognize, even at this point, that it's only one of many possible algorithms or implementations for this difference equation.

Just for example, I can consider that equation for that block diagram. And here, I've re-drawn it. So here, are once again. I have the same block diagram that we just saw. And I can recognize, for example, that this, in essence, corresponds to two linear time-invariant systems in cascade.

Now, that assumes, of course, that my initial conditions are such that the system is, in fact, linear. And that, in turn, requires that we're assuming initial rests, namely, before the input does anything other than 0. There are just 0 values stored in the registers. But assuming that it corresponds to a linear time-invariant system, this is a cascade of two linear time-invariant invariant systems.

We know that two linear time-invariant systems can be cascaded in either order. So, in particular, I can consider breaking this cascade here, and moving this block over to the other side. And so let's just do that. And when I do, I then have this combination of systems, and, of course, you can ask what advantage there is to doing that, and the advantage arises because of the fact that in this form, exactly what is stored in these delays is also stored in these delay registers.

In other words, it's this intermediate variable-- whatever it is-- down this chain of the delays and down this chain of delays, and so, in fact, I can collapse those delays into a single chain of delays. And the network that I'm left with is the network that I indicate on this view graph, where what I've done is to simply collapse that double chain of delays into a single change of delays. Now, one can ask, well, what's the advantage to doing that?

And one advantage, simply stated, is that when you think in terms of an

implementation of a difference equation, a delay corresponds to a storage register, a memory location, and by simply using the fact that we can interchange the order in which linear time-invariant systems are cascaded, we can reduce the amount of memory by a factor of 2.

Now, an essentially similar procedure can also be used for differential equations, in terms of implementation using block diagrams or the interpretation of implementations using block diagrams. And let me first do that-- rather than in general-- let me first do it in the context of a specific example.

So let's consider a linear constant-coefficient differential equation, as I've indicated here, and I have terms on the left side and terms on the right side. And with the differential equation, let's consider taking all the terms over to the right side of the equation, except for the highest derivative in the output.

Next, we integrate both sides of the equation so that when we're done, we end up with on the left side of the equation with  $y(t)$ . On the right side of the equation with the appropriate number of integrations. And so the integral equation that we'll get for this example  $y(t)$ , the output, is  $x(t)$  plus  $b$ , the scale factor times the integral of the input, and minus  $a$ , that scale factor, times the integral of the output.

So to form the output in the block diagram terms, we form a linear combination of the input, a scaled integral of the input, and a scaled integral of the output, all of that added together. So we need, in addition to the input, we need the integral of the input. And so this box indicates an integrator. In addition to the output, we need the integral of the output. And now, to form  $y(t)$ , we multiply the integrated input by the scale factor,  $b$ . And add that to  $x(t)$ , and we take the integrated output, multiply it by  $-a$ , and add to that the result of the previous addition, and according to the integral equation, then that forms the output.

So just as we did with the difference equation, we've converted the differential equation to an integral equation, and we have a block diagram form very similar to what we had in the case of the difference equation. Now, the initial conditions, of course, are tied up in, again, how these integrators are initialized. Assuming that we

impose initial rest on the system, we can think of the overall system as a linear time-invariant system, and it's a cascade of one linear time-invariant system with a second.

So we can, in fact, break this, and consider interchanging the order in which these two systems are cascaded. And so I've indicated that down below. Here, I've simply taken the top block diagram, interchanged the order in which the two systems are cascaded. And here, again, we can ask what the advantages to this, as opposed to the previous one. And what you can see, just as we saw with the difference equation, is that now, the integrators-- both integrators-- are integrating the same thing.

In particular, the input to this integrator and the input to this integrator are identical. So in fact, rather than using this one, we can simply tap off from here. We can, in fact, remove this integrator, break this connection, and tap in at this point. And so what we've done then, by interchanging the order in which the systems are cascaded, is reduced the implementation to the implementation with a single integrator. Very much similar to what we talked about in the case of the difference equation.

Now, let's just, again, with the integral equation or the differential equation, look at this somewhat more generally. Again, if we take the differential equation, the general differential equation, integrate it a sufficient number of times to convert it to an integral equation. We would then have this cascade of systems. And again, if we assume initial rest, so that these are both linear time-invariant systems, we can interchange the order in which they're cascaded. Namely, take the second system, and move it to precede the first system.

And then what we recognize is that the input to this chain of integrators and this chain of integrators is exactly the same. And so, in fact we can collapse these together using only one chain of integrators. And the system that we're left with then is a system that looks as I've indicated here. So we have now just a single chains of integrators instead of the two sets of integrators.

So we've seen that the situation is very similar here as it was in the case of the difference equation. Again, why do we want to cut the number of integrators in half? Well, one reason is because integrators, in effect, represent hardware. And if we have half as many integrators, then we're using half as much hardware.

Well, let me just conclude by summarizing a number of points. I indicated at the beginning that linear constant-coefficient differential equations and difference equations will play an important role as linear time-invariant systems throughout this course and throughout this set of lectures. I also stressed the fact that differential or difference equations, by themselves, are not a complete specification of the system because of the fact that we can add to any solution a homogeneous solution.

How do we specify the appropriate initial conditions to ensure-- how do we specify the appropriate initial conditions to ensure that the system is linear and time-invariant? Well, the auxiliary information, namely, the initial conditions associated with the system being causal, linear, and time-invariant are the conditions of initial rest. And, in fact, for most of the course, what we'll be interested in are systems that are in fact, causal, linear, and time-invariant. And so we will, in fact, be assuming initial rest conditions.

Now, as I also indicated, there are a variety of efficient procedures for solving differential and difference equations that we haven't yet addressed. And beginning with the next set of lectures, we'll be talking about the Fourier Transform and much later in the course, what's referred to as the Laplace Transform for continuous time and the Z-transform for discrete time. And what we'll see is that with the Fourier Transform and later with the Laplace and Z-transform, we'll have a number of efficient and very useful ways of generating the solution for differential and difference equations under the assumption that the system is causal, linear, and time-invariant.

Also, we'll see in addition to the block diagram implementations of these systems that we've talked about so far, we'll see a number of other useful implementations that exploit a variety of properties associated with Fourier and Laplace Transforms.

Thank you.