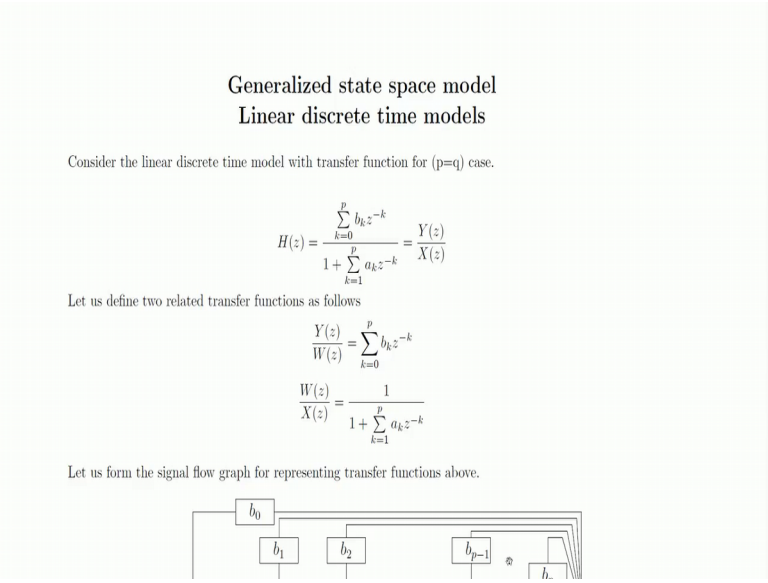
Mathematical Methods and Techniques in Signal Processing - I
Prof. Shayan Srinivasa Garani
Department of Electronic Systems Engineering
Indian Institute of Science, Bangalore

Lecture – 06
State space representation

So, in the last class we derived the transfer function for a linear discrete time model basically using an ARMA model. So, we will just go one step further into the discrete time model and derive the state space model here.

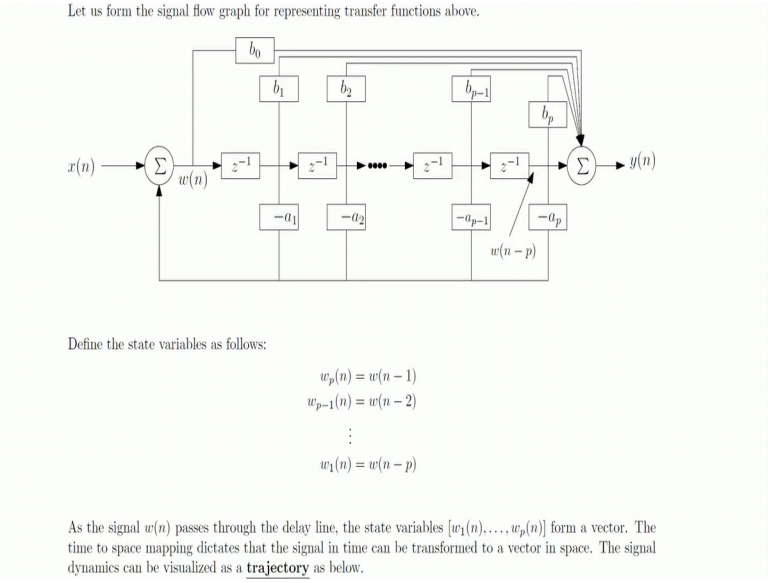
(Refer Slide Time: 00:35)



So, let us consider the case when p equals q, I mean this is without loss of generality, if p is lesser than q then we can take q states, if p is greater than q then we can take p states. So, let the transfer function H of z be given by Y of z by X of z and this is basically the numerator polynomial which is given by sigma k equals 0 to p, b suffix k is e power minus k divided by this denominator polynomial which is 1 plus sigma k equals 1 to p, a k z power minus k right.

And then we have the transfer functions which are related as follows. So, Y of z by W of z is basically this numerator polynomial which is in b and the denominator polynomial is W of z by X of z which is given by this polynomial.

(Refer Slide Time: 01:41)



So, the first one the foremost step is to basically form a signal flow graph which represents the above transfer functions. So, we start with the input which is x of n right and this is basically it goes through a delay line and we have p equals q k so basically we have p number of delay elements. So, z power minus 1 is one delay element like this we have p such delay elements.

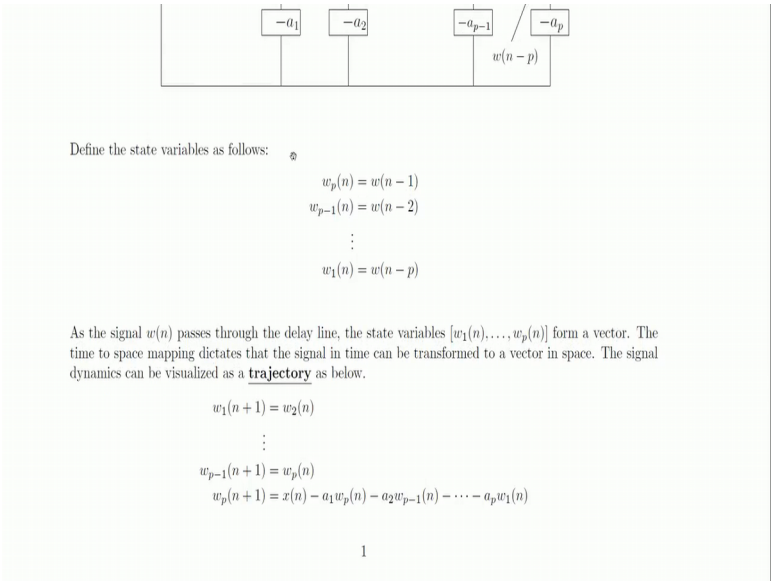
In the forward path we have all these bs of is which are connecting these outputs from the tap delay line. So, w of n is just at the output of the summer and this goes through this gain b naught and then goes to the output of the summer here y of n at the output summer. Then again from the output of the first delay line you have basically a feed forward path which is through the gain b 1 and then there is a feedback path which is through minus a 1 which is going back to the input. The feedback path leads into the input here and feed forward path goes to the output. So, like this we have b 2 which is at the output of the second tap delay line and it is connected to the output and minus a 2 is basically is a feedback path going back to the input and this is going on forward till we have the gains b p and minus a p which are basically the feed forward and the feedback paths which are connecting the output and the input.

Now, what we need to do is basically to formulate these state variables. So, the first observation that you need to make is the following. So, you have the signal w n here at

the output of the summer here and after p such delay elements this signal here is w of n minus p right this is straightforward. Now, let us start formulating the state variables.

Now, we will start with w p of n and we define this to be w of n minus 1. So, if you look at the output of the first delay element that is w n when it is filtered through the first delay you get w n minus 1 and that w n minus 1 I call it w p of n. And w p minus 1 n is basically w of n minus 2 so on and so forth and we have w 1 n which is w of n minus p this is a very important step. So, what we have done here is very simple. As the signal w n is passing through the delay line we are tapping the outputs of the delay line and we are formulating them as state variables. So, these are basically the hidden states of the system.

(Refer Slide Time: 05:01)

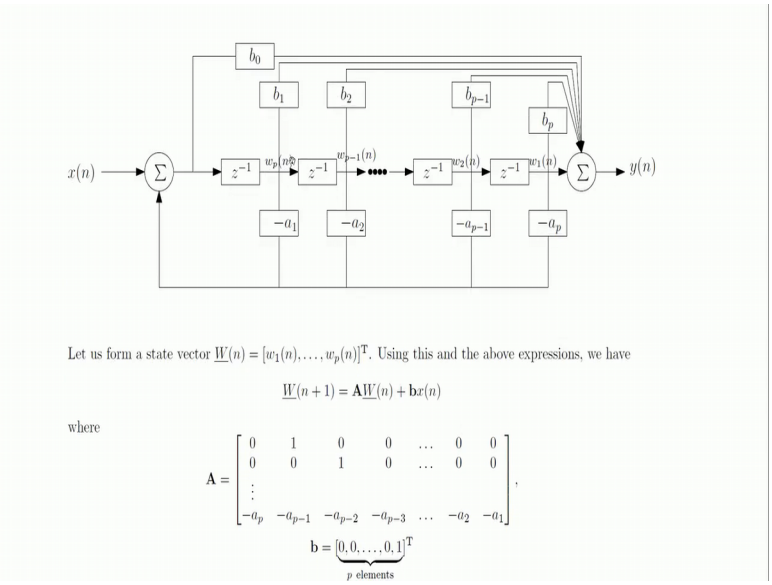


Now as a signal w n passes through the delay line the state variables w 1 n so on till w p n they form a vector. It is very important they form a vector. So, this is very important concept because a signal through a delay line can be transformed to a vector in a space. So, the conventional notion of what we think about as patterns in space can now be thought of using signals through a tab delay line, right. So, this is basically the time to space mapping which dictates that signal in time can be transformed to a vector in space and the signal dynamics can be visualized as a trajectory as below right.

So, we have w 1 of n plus 1 is w 2 n so on and so forth, w p minus 1 of n plus 1 is w p n right and w p n plus 1 is basically this point which is I say w p n is w n minus 1 right and

w p n plus 1 is basically your w n which is basically x of n minus all the terms that you have in the feedback path, which is basically x of n minus a 1. So, you have x of n, w n is the signal which is x of n minus a 1 times w p n minus a 2 times w p minus 1 n so on and so forth till minus a p times w 1 n.

(Refer Slide Time: 06:47)



Now, let us form a state vector which is w n as I told you which is a capital W n which is this vector which is w 1 and so on till w p n and we have these interim state variables which are described as follows right.

So, now, using this set of equations that we formulated here right we will try to connect them in the form of a matrix equation W of n plus 1 is some matrix A times W of n plus some vector b times x of n where this matrix A has this form and b is basically all 0s followed by 1 which is basically comprising of p elements, p element vector. It is very straightforward, I mean if you just look into these set of equations that that are here right. If you look at the set of equations here and rewrite them using the matrix form you get this equation which is very very simple.

(Refer Slide Time: 08:02)

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & & & & & & \\ -a_p & -a_{p-1} & -a_{p-2} & -a_{p-3} & \dots & -a_2 & -a_1 \end{bmatrix},$$

$$\mathbf{b} = \underbrace{[0, 0, \dots, 0, 1]}_{p \text{ elements}}^{\mathrm{T}}$$

Similarly, one can do the math for expressing the output $y(n)$ through a sequence of equations below:

$$y(n) = b_0 w(n) + \sum_{k=1}^{p} b_k w_{p+1-k}(n)$$

$$y(n) = b_0 w_p(n+1) + \sum_{k=1}^{p} b_k w_{p+1-k}(n)$$

$$y(n) = b_0[x(n) - a_1 w_p(n) - a_2 w_{p-1}(n) - \cdots - a_p w_1(n)] + b_1 w_p(n) + b_2 w_{p-1}(n) + \cdots + b_p w_1(n)$$

$$y(n) = \sum_{k=1}^{p} [b_k - b_0 a_k] w_{p+1-k}(n) + b_0 x(n)$$

$$y(n) = \mathbf{c}^{\mathrm{T}} \underset{\sim}{W}(n) + \mathbf{d}x(n)$$

where

$$\mathbf{c} = \begin{bmatrix} b_p - b_0 a_p \\ \vdots \\ b_1 - b_0 a_1 \end{bmatrix},$$

$$\mathbf{d} = b_0.$$

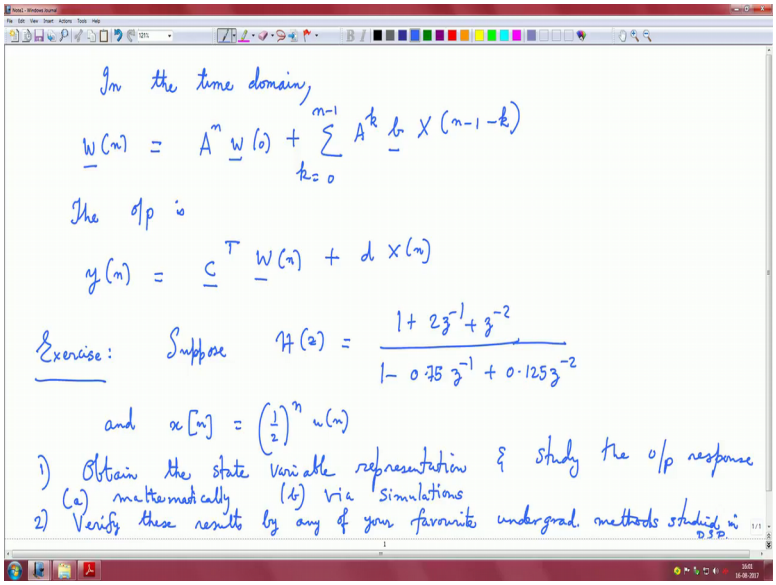
Now, we can do the math for expressing the output y of n through a sequence of equations as below. So, what we have is as follows. So, our y of n is basically b naught times w of n plus summation b k w p plus 1 minus k of n, I mean these subscripts you should just carefully observe as you see through the signal flow graph.

Now, we do a little bit of careful math here some simplification in true algebra which is b naught times w p of n plus 1. Recall w n is basically w p n plus 1 plus sum k equals 1 to p b k w p plus 1 minus k of n right. We have this equation. Now, we use w p of n plus 1 that we derived earlier I mean this is basically we have this equation here which relates x of n and the rest of the parameters. So, once we do that, y of n is basically b naught times x of n minus a 1 w p of n minus a 2 w p minus 1 and so on till a p w 1 n plus all the rest of the terms that you have from this summation.

So, we now have expanded w of n in terms of your coefficients bs of i, then x of i, x of n then your other coefficients which are a is. And then writing it in a compact form we have y of n is basically some r k equals 1 to p b k minus b naught a k times w p plus 1 minus k of n plus b naught x of n. So, this is this equation. Now, this is a scalar essentially b naught x of n is basically a scalar and if you think about the summation this can be thought about as an inner product between the state vector w n and some vector c with p elements that are described as follows.

So, this vector c is basically given by this form the first element is b p minus b naught a p and so on the last element is b 1 minus b naught a 1 right. So, now, we have all our variables that can describe the state space model because the important parameters are these cs and ds which link the state reps state variable state vector w of n and x of n to the output y of n and we have the matrix A and the vector b that link the state vector at time n and input to the state vector at time n plus 1. So, now, we have all the ingredients that can describe the state space model and this is the most general form that we have. And this representation is very useful for you to simulating discrete time systems given arbitrary inputs and transfer functions. To this derivation we are ready to go into the time domain equivalent of this formula.

(Refer Slide Time: 11:57)



So, in the time domain we can write the state vector w n in this form which is A power w n 0 plus sum k equals 0 to n minus 1 A power k times this vector b times x of n minus 1 minus k. And the output is y of n which is C transpose into W n because the vector the vector plus d times X n and you plug in W and from the earlier step and then you can simplify the rest of the equations. I hope this is clear to all of you.

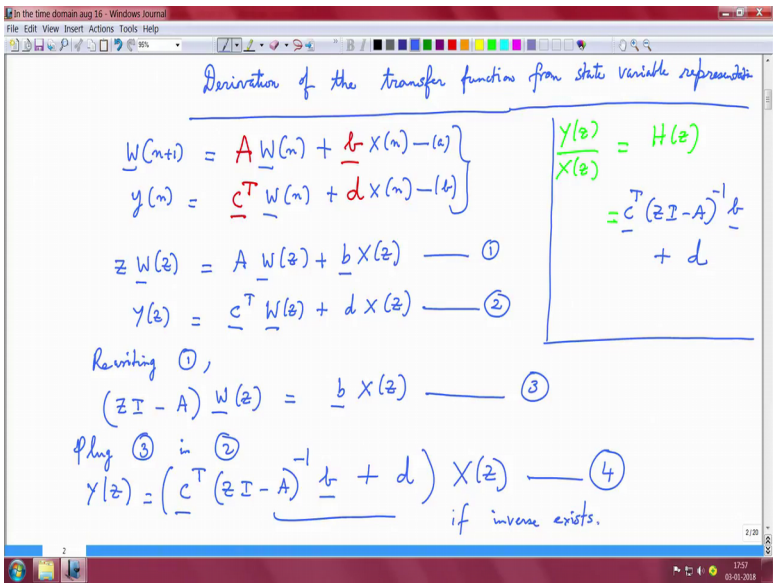
So, basically we should know what is the derivation, the derivation is a very important point because it tells you to relate the input and the output through a state vector and how do you define the state vector because state vector representation is not really unique I mean different people can have different ways in which they can define the states a

different take of points. So, therefore, each of your answers would be different. If one were to ask you to get the state way variable representation and derive the transfer function, but the transfer function has to be the same though the rest can be different.

So, with this I will give you a short exercise. Suppose I give you a transfer function H of z which is 1 plus 2 z power minus 1 plus z power minus 2 or 1 minus 0.75 z power minus 1 plus 0.125 z power minus 2. And suppose you have x of n is one-half power n, u of n. Obtain the state variable representation and study the output response a mathematically via simulations. Verify these results by any of your favorite undergraduate methods studied in DSP. You have done different methods for figuring out the output given the input right you have done many techniques. So, therefore, what I want you to do is obtain the state variable representation for this transfer function, study the output response mathematically, then once you have the state parameters write a short MATLAB code where you can simulate the system. What you do in simulation should be exactly what you do mathematically right and then verify these results by any of the methods you have studied in your under graduate DSP course. Clear so far.

Based upon the state variable representation we are ready to derive the transfer function from state variable representation.

(Refer Slide Time: 17:03)



So, what we have to do is basically start with the equations we had earlier W of n plus 1 is basically some matrix A, I think I probably would want to use a different color for this

matrix A times W n plus some vector. So, let me also use a different color for this vector plus b times x of n x of n is s a scalar, but we could have a vector also for this. Then we have the output y of n is basically C transpose W n plus d times x of n. Now, we have these pairs of equations right.

Now, we have to derive the transfer function. So, let us apply the Z transforms to equations a and b right. So, Z times W of Z is basically A times W of Z plus b times X of Z. So, this is a vector and then y of Z is C transpose W Z plus d times X of Z.

Now, I can rewrite this equation let me call this equation 1, let me call this equation 2. So, I can rewrite equation 1 in this form ZI minus A times W z is basically b times X of Z right. Remember A is a matrix we cannot write simply Z minus A right is an identity matrix there Z times identity matrix. So, this is what we have and therefore, let us call this equation 3.

Now, we can use equation 3 and we can plug it into 2, we land up with Y Z is basically C transpose times we have to replace this by W Z here which is basically ZI minus A inverse times b X of Z plus d times X of Z. So, this is going to be this. Now, we are ready to derive this transfer function basically we say Y Z by X of Z is equal to H of Z and this is given by into ZI minus A inverse b. So, this is the familiar form that most of you might be used to for the transfer function of the state variable representation, clear.

So, basically you can go with, if you have a linear time invariant model you can get the state space representation and your state space representation can be different from what somebody else may have depending upon how you have formulated your state variables. But your transfer function has to be the same. Well there should be an underscore here consistent with the previous equation. You may question the conditions under which this inverse exist I think I would just say that if inverse exists.

So, we will stop here for this module and then we will show the equivalence.