NPTEL

NPTEL ONLINE COURSE

CH5230: SYSTEM IDENTIFICATION

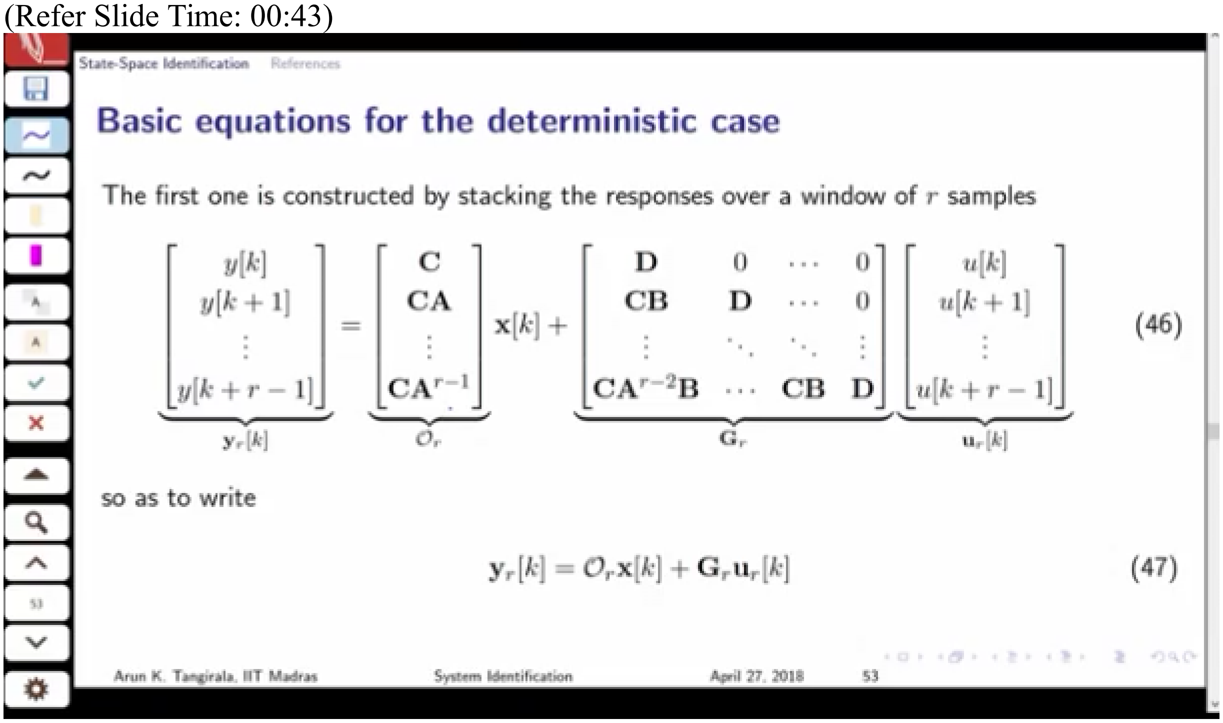
STATE-SPACE/SUBSPACE IDENTIFICATION
PART 6

ARUN K. TANGIRALA

DEPARTMENT OF CHEMICAL ENGINEERING,
IIT MADRAS

So let's start with the noise free case as usual, so the same way we started with the noise free case for the impulse response, now we are starting with the noise free case for the arbitrary input output data case, the idea is the same, we are going to stack, this time we are going to stack output data, like we stack impulse response data.

And what is our goal? Our goal is to get A, B, C, D,
(Refer Slide Time: 00:43)

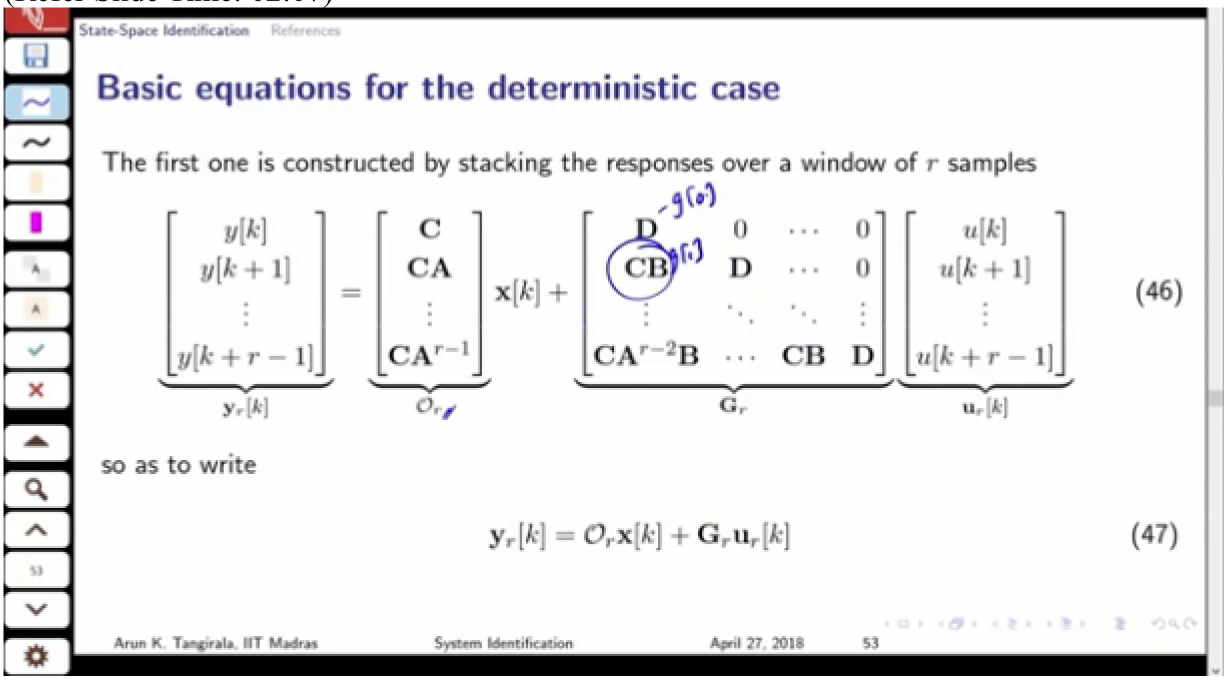


we never knew and when in LKG when we are learning A, B, C, D, determining A, B, C, D will be so difficult, right, life can present you a lot of surprises.

So what we do is we stack up to R, so we assume that you are standing at K from Y(k) we go up to Y(K + R-1) and all I have done is I have written the output equations at each of those instance, so you will recognize that this is nothing but OR, unsurprisingly that has to be OR.

And this matrix that we have here, why do I call it GR? What do you recognize that matrix as? Based on the, I'm tempted to give you a clue, but let's see if you can, of what? Is it a Hankel matrix exactly? So it's not a Hankel matrix, because Hankel matrix has a structure, but it's a matrix of something, correct, so it's a matrix of impulse response coefficients, right, you recognize these are, this is G(1), this is G(0),

(Refer Slide Time: 02:07)



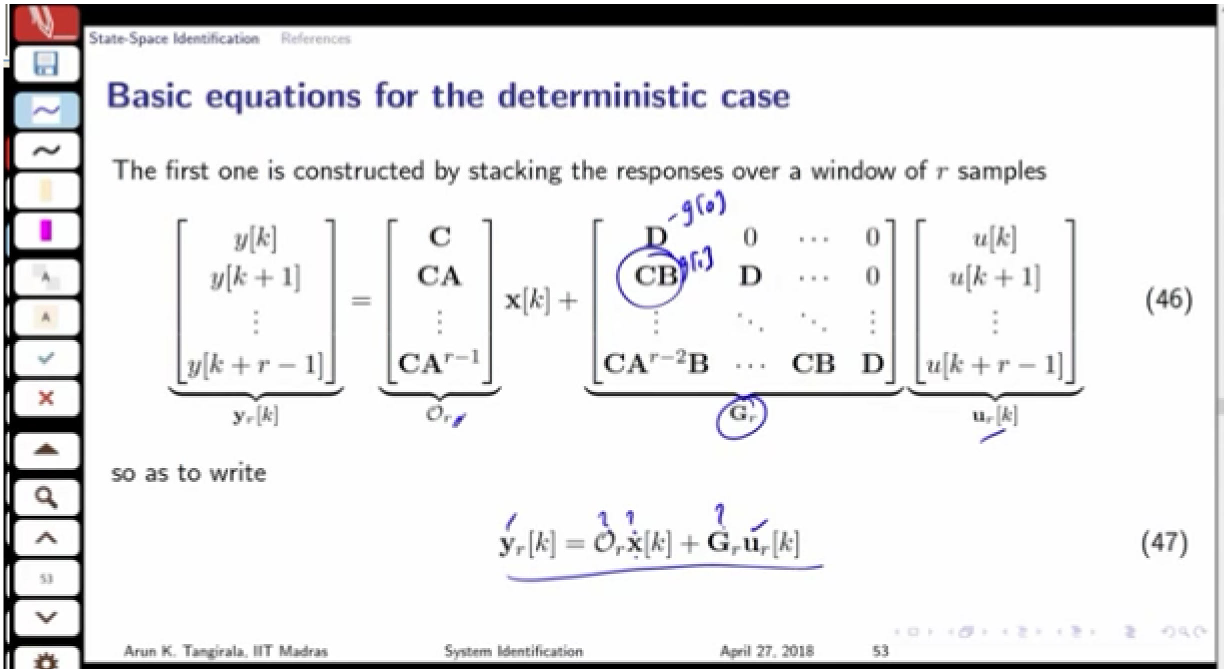
in fact what we could have done is from the arbitrary input output data, what is a simple way of estimating A, B, C, D, first estimate the impulse response coefficients through FIR modeling, and simply apply Kung's method, that's it, there is no, you know, there is no sweat broken there we could have done that, think as to why we don't want to do that, I mean you can do it, it will give you some descent estimates, but here we are not doing that, implicitly the impulse response coefficients are being estimated, right.

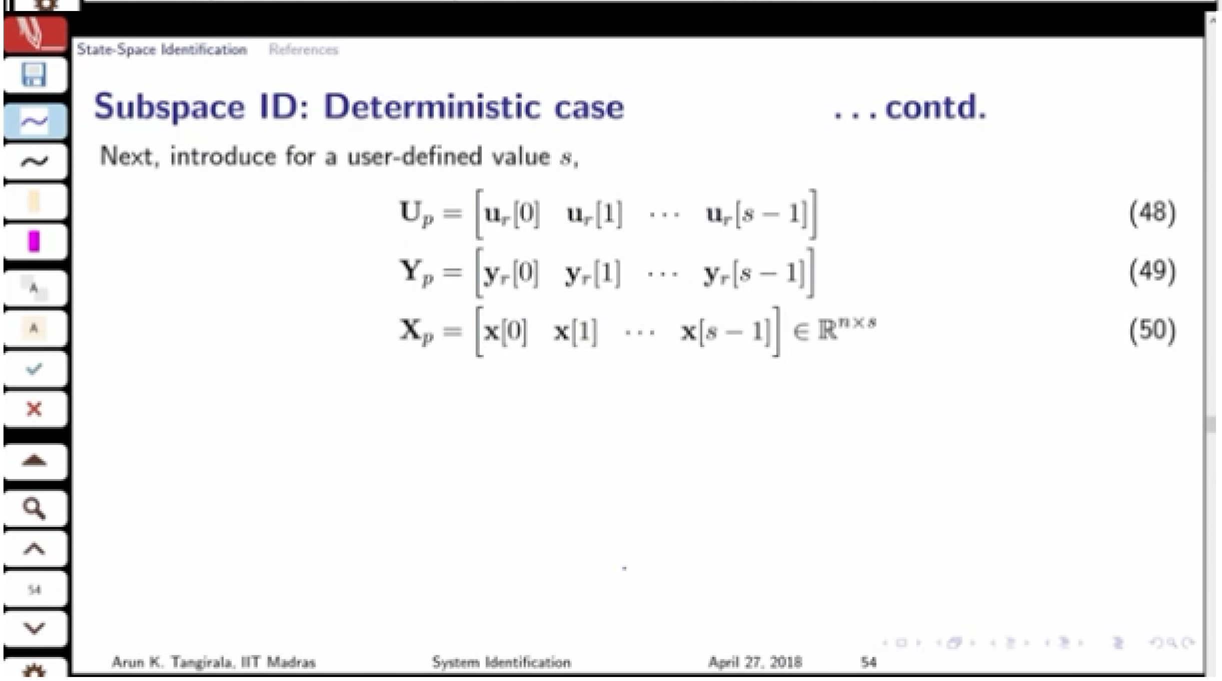
So this is nothing but the matrix of impulse response coefficients, so essentially what you are doing is, you are writing the convolution equation here, these GR is a matrix of impulse response coefficient, this is a stacked UR, so until this equation there is no issue at all. Only problem with subspace identification algorithm is that you run out of symbols pretty soon, okay.

The lower case, upper case, bold faced, simple faced everything, italics including Greek symbols and that was my experience also right in the chapter in the book. Anyway, so you can see that there are quite a bit of number of quantities involved here, so this part is clear, very good, so let's proceed now.

Now if you look at this equation, what is given to me in 47? I'm given Y, what else is given to me? U is given, so I'm supposed to estimate O, X, G, it's quite a bit, in input output identification this O, X is not available, is not present at all, you just simply write the convolution equation life is happier, okay.

(Refer Slide Time: 03:53)

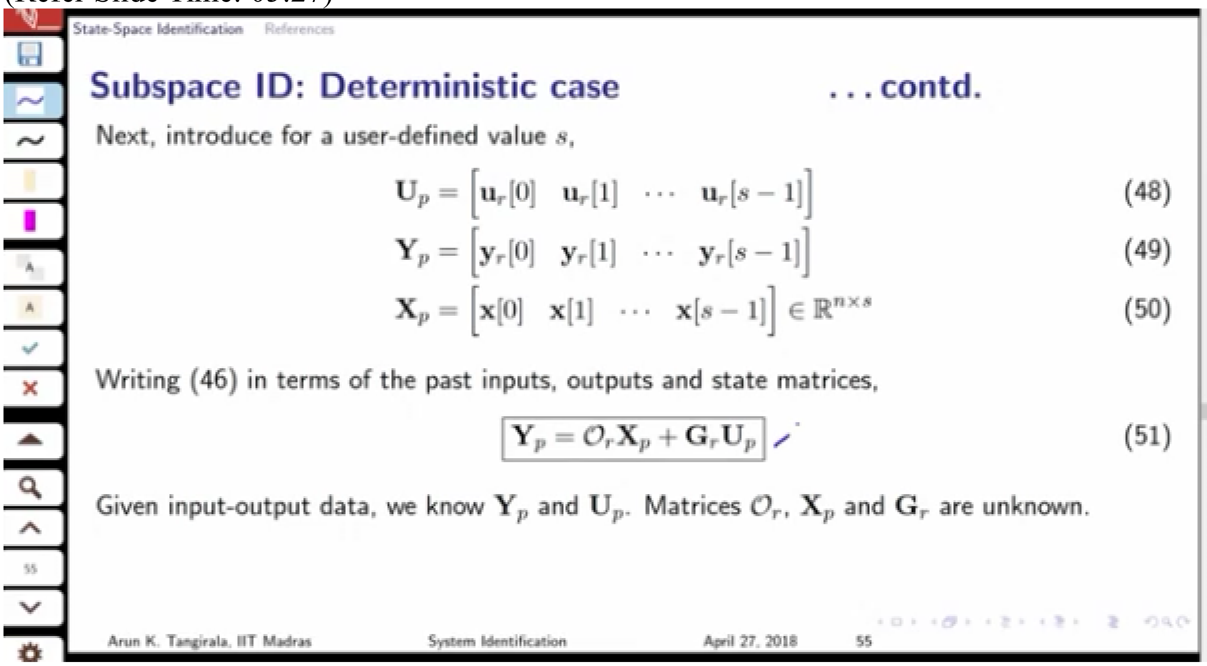




Now what you do is you construct matrices, because you are not happy with vectors alone, because after all if you're exist you're exhausting symbols you should also exhaust one dimensional things, fortunately no tensors come in, we'd stop with matrices there, okay, so you construct this matrices and this P here denotes past for some reason will worry about that a bit later, so if the user chooses first R like you choose L in Hankel method and you choose M, what are you choosing here? R and S at least easy to remember alphabetically consecutive, there L and M are consecutive, R and S are consecutive here, so R and S are user choices.

Again in MATLAB you must have never specified R and S, because the routine automatically makes this choices for you, but you should know that is making, sorry this choice is for you, else you know then you're blind user, alright, so choose R construct vector use and then choose S construct matrices U, Y and X of course, you won't be constructing X, but I'm just saying symbolically if you construct this matrix, then you can rewrite the same equation 47 in terms of the matrices here.

(Refer Slide Time: 05:27)

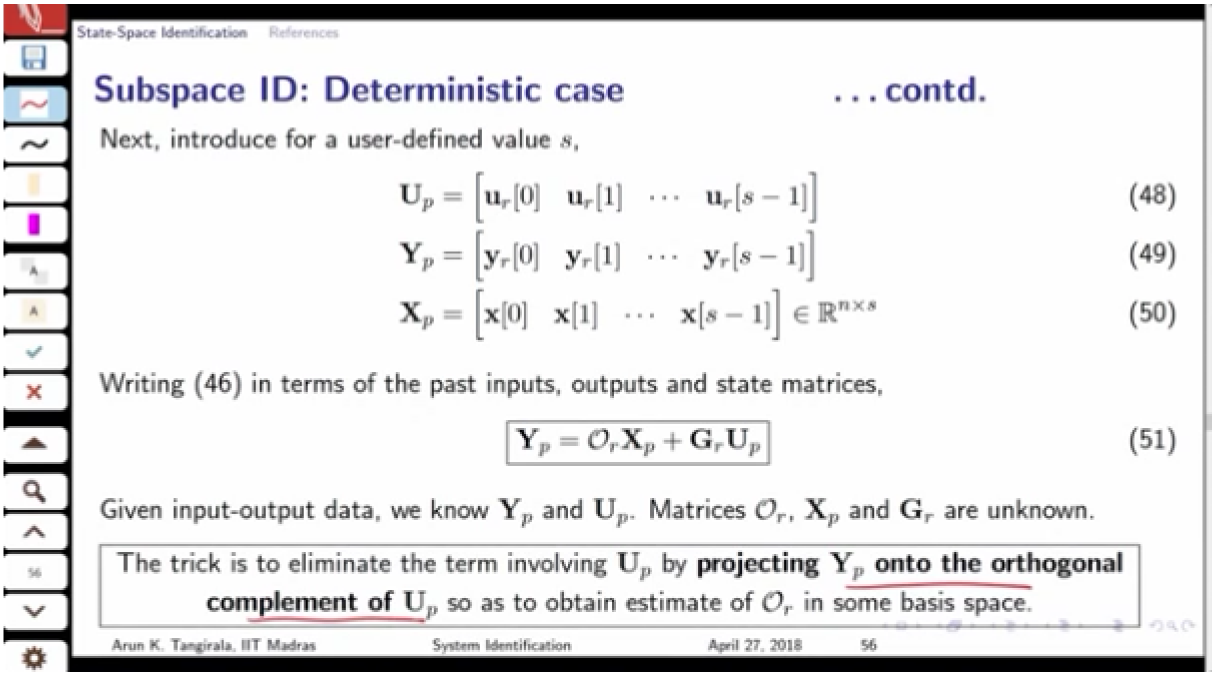


The idea is to use the full data that's available to you, and so that you can good estimates. Again here I know Y and I know U, and what do I need to find out? Many things, I need to first find this OR, once I get the OR then I'll get my A and C and then I can get my B and D through linear regression.

What about G? Don't I want to find G, I also want to estimate impulse response coefficients right, but why do I need to? What is my objective? A, B, C, D right, so I may not have to estimate G that is the idea here, although G is unknown, it is not the unknown of interest to me, therefore I may want to eliminate this term somehow that is the key idea in subspace ID, because it is a unknown that is not of interest to me, but still needs to be implicitly calculated in order to estimate OR, what do I do? I take a smarter root and that is the beautiful idea in subspace ID where this term is killed completely.

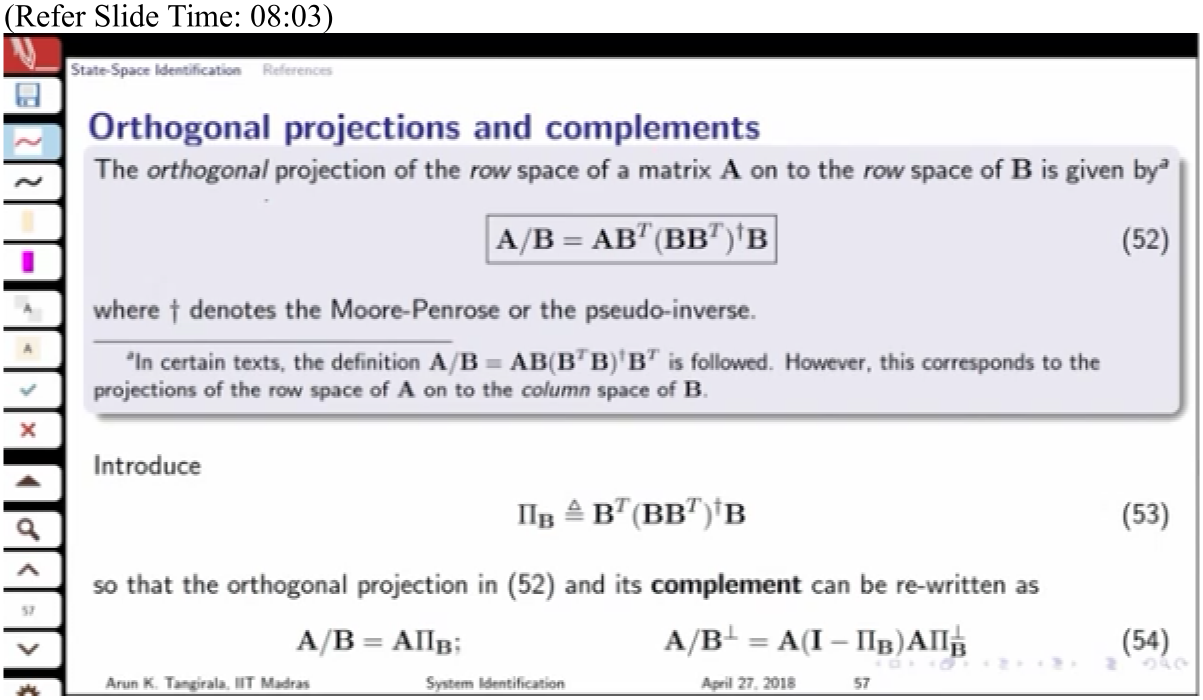
How do you think I can kill it? That is through linear algebra methods, so you imagine that you are multiplying both sides of this equation with something, so that selectively that term is killed out, that term is zero it out, what would be that term? Sorry, G inverse will not zero it out, so some kind of matrix that is orthogonal to UP, right, and that is what we do by projecting YP on to the orthogonal compliment of UP,

(Refer Slide Time: 07:35)



so you multiply both sides with the orthogonal compliment of UP such that the product of UP and by definition if he say that there is some matrix is an orthogonal compliment of another matrix A, then the product of those two is 0, so very briefly talked about orthogonal projections not fully, let's talk about the concept and the numerical implementation, so the orthogonal projection of the row space of a matrix A on to the row space of another matrix B,

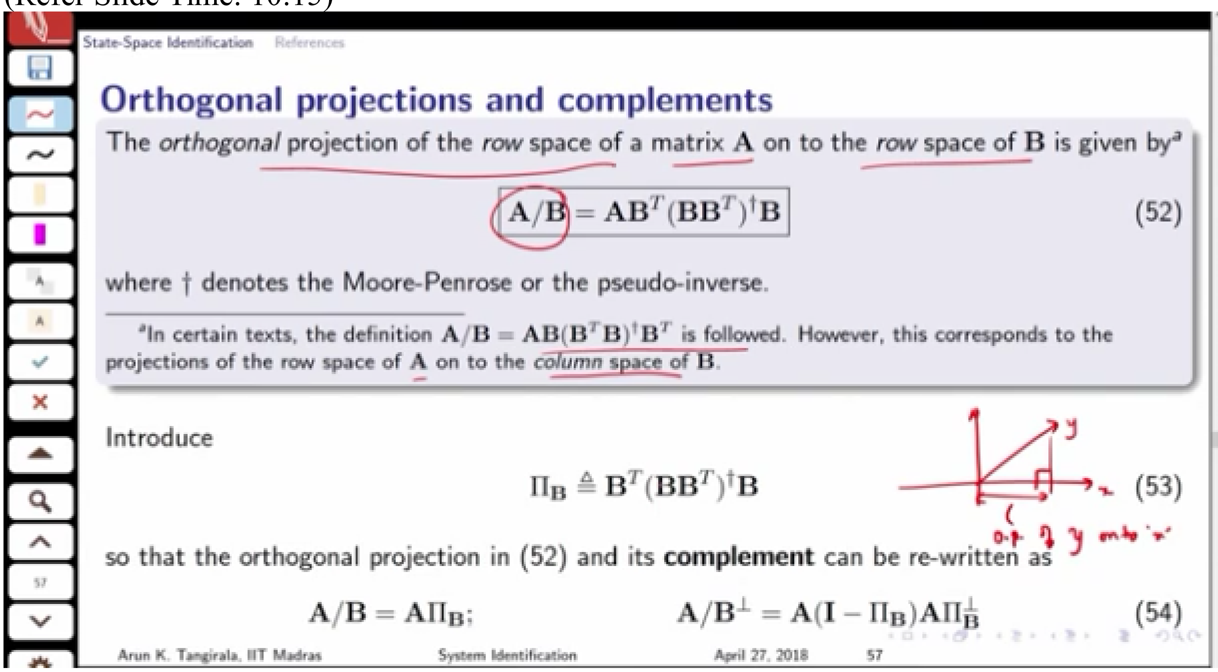
(Refer Slide Time: 08:03)



now you have to be careful whether you are projecting the row space or the column space, if you're post multiplying then it is column space and then you know and so on, so keep in mind, in certain texts the definition, this definition is followed, okay, but this corresponds to the projections of row space of A onto the column space of B, remember, alright.

So you want to project the row space of a matrix A on to the row space of B, and this is denoted as A forward slash B, what is this projection business? You have learnt in least squares right, so if you recall least squares, what is it that we are trying to do? Suppose the vector Y lives in a two dimensional space, there is a vector Y that lives in a 2 dimensional space, and my regress are is only the X axis, then the least square estimate is nothing of Y, is nothing but the projection of this two dimensional thing vector on to the one dimensional thing, so this is called the orthogonal projection of Y on to X, this is the orthogonal projection of this vector Y on to X. Why is it called orthogonal? Because the residual is orthogonal, okay, the residual is orthogonal, I can take an oblique projection also which we will work talk about shortly, but that oblique projection would mean that the error is making it a certain angle, anyway so the orthogonal projection here of our matrix A, so it's a bit more involved here, we have just talked about vector spaces but here we are talking of row spaces of a matrix and so on is given by this expression,

(Refer Slide Time: 10:15)

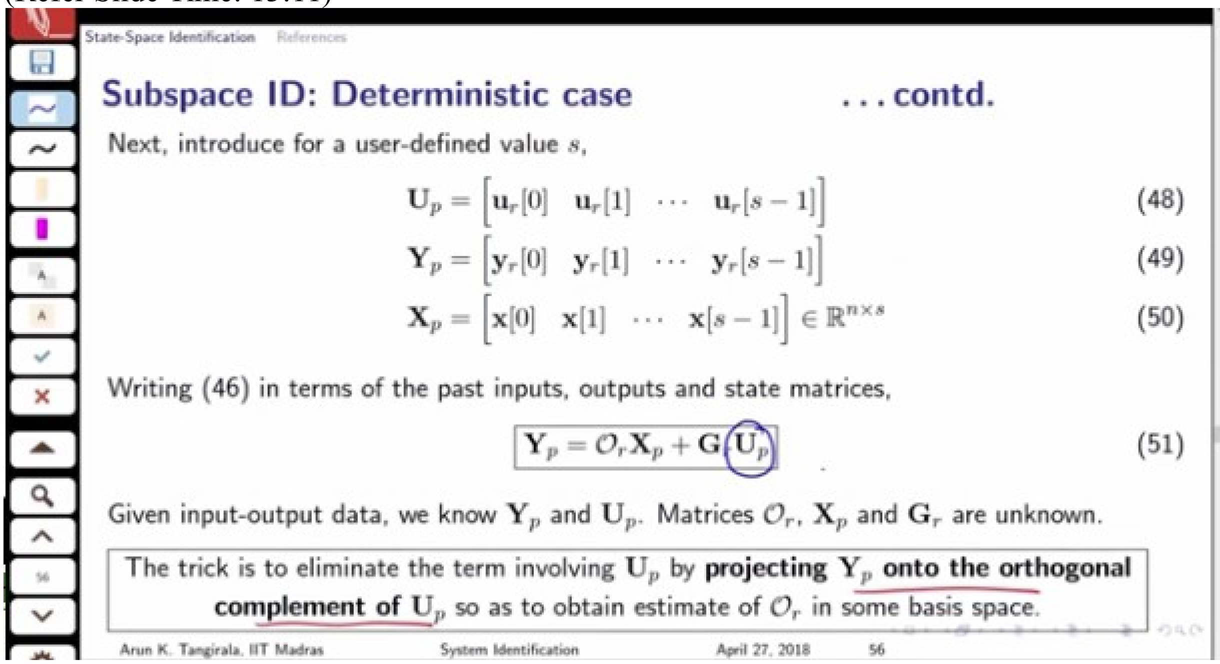


and the quantity here with which your post multiplying A we can denote that as pi B, you can use any other symbol but we have use this symbol.

And therefore we can write the orthogonal projection as A times pi B, and the orthogonal compliment is I – pi B, alright, the compliment can be written as in fact sorry there is an equality missing here, it's not A times I-pi B times A pi B perpendicular that would be deadly, so the pi B perpendicular is nothing but, what is pi B perpendicular? I- pi B, so that what do I get? A times pi B + A times pi B perpendicular is what is it that you get? A, that's why it's called a orthogonal compliment, whatever is left out in this projection is contained in this, that's why it's called the compliment. And what is pi B times pi B perpendicular? 0, is it correct or no? It's not, pi B + pi B perpendicular is I, okay, sorry.

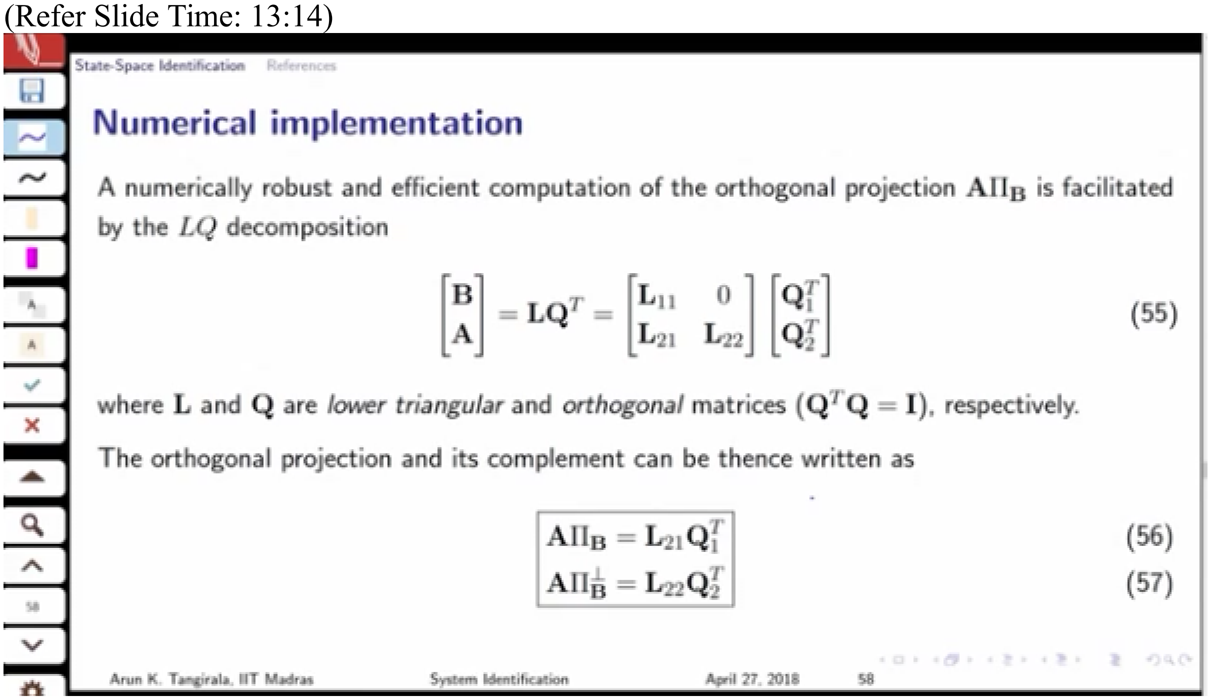
So now let us look at B times pi B perpendicular? What is B times pi B perpendicular? So this implies B times pi B perpendicular, what is it? First quantity is B, then what you get? –B so 0, so the projection of B on to pi B perpendicular is 0, now you get the idea why? So why do you think we are talking about this now? What is the idea with respect to our original problem? Post multiply in this case with which perpendicular? Pi, what is B for us here? So we just have to project Y on to the orthogonal compliment of UP, so if I multiply both sides with UP perpendicular, then the second term is eliminated without knowing G so that's a nice thing, I don't care what G is, it is going to be 0,

(Refer Slide Time: 13:11)



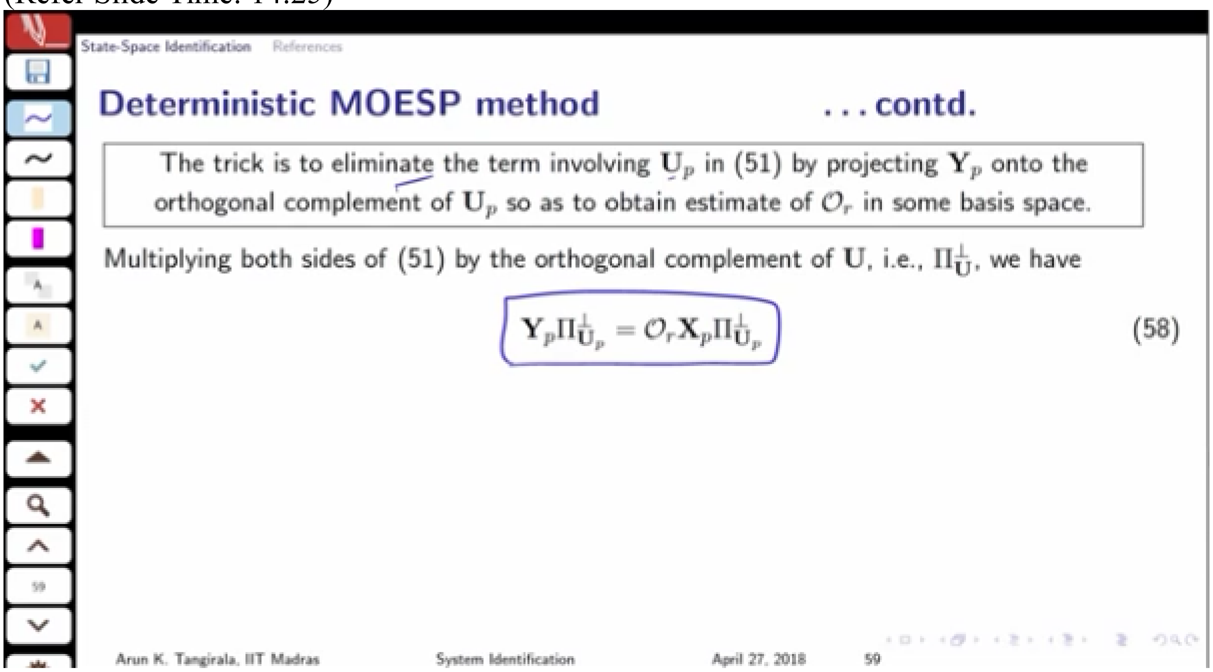
so that is the idea, by the way the orthogonal compliment calculations,

(Refer Slide Time: 13:14)



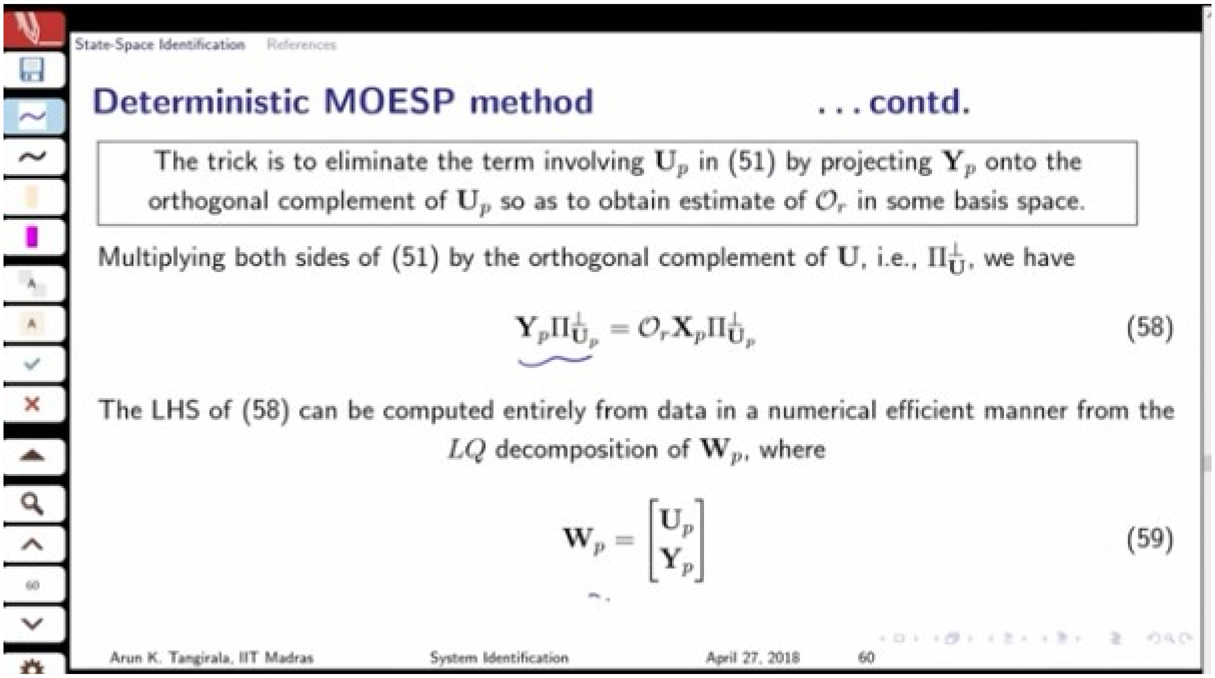
this is theory, but numerically how do you implement this? A numerical implementation is done through what is known as LQ factorization, so always remember even in least squares we have a solution, and then there is a numerical implementation, if you recall least squares implementation is numerically best implemented through either QR factorization or SVD.

Why do we have a different theory for numerical implementation? Because I've to worry about rounding of errors, robustness and so on, so likewise here computation of orthogonal compliment is done through LQ factorizations, so if I want to compute A pi B and A pi B perpendicular, I construct this matrix do an LQ factorization which you must have learnt in a numerical linear algebra course, and then you can compute the orthogonal projection and it's compliment using this, that is numerical implementation part, right, so that we now already know what to do, the trick is to eliminate the term involving UP by multiplying both sides with this, excellent. So now what am I going to do? What do you think I should do?
(Refer Slide Time: 14:25)



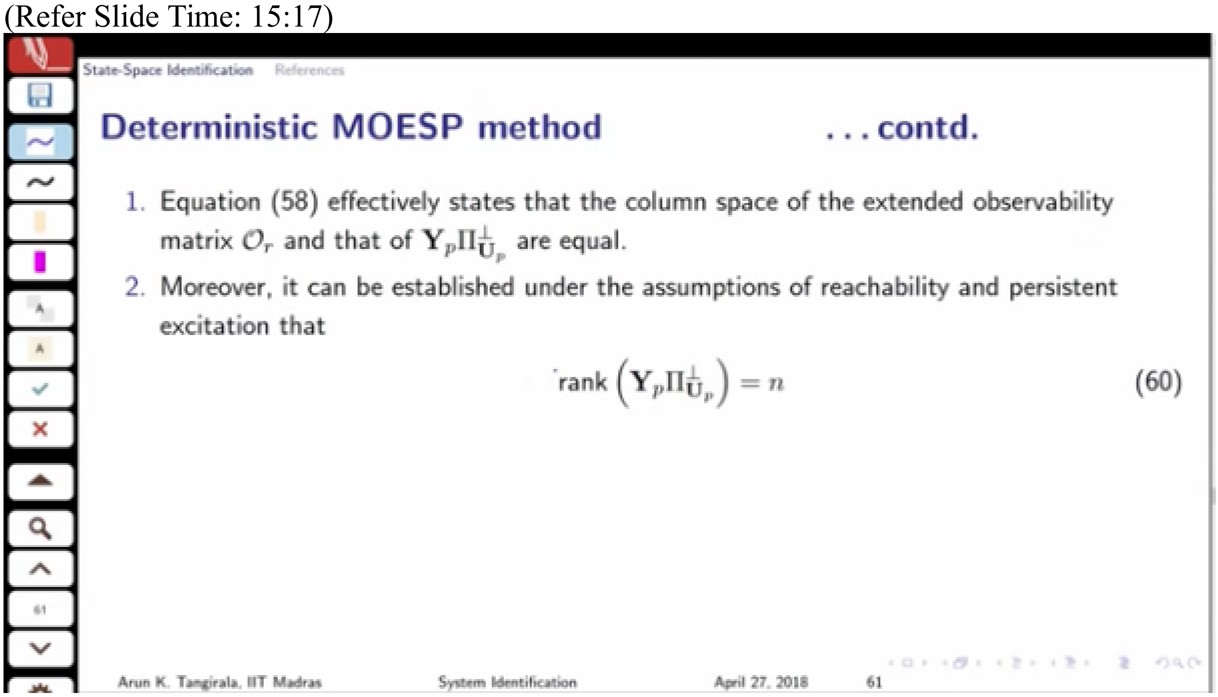
I've gotten rid of one term which had a unknown, of interest to me, now I want to get to this.

Do I know the left hand side completely? Yes, so I know this two, I don't know anything on the right hand side but I know that the right hand side is some OR times T, some matrix, okay, I know that it is OR's times some matrix, so I'm going to use that, what we do is first compute this numerically, this is just through LQ factorization, same thing, this is B,
(Refer Slide Time: 15:00)



this is A as per our previous notation, you compute this left hand side and then the first thing to be shown is that the left hand side has the properties of an extended observability matrix,

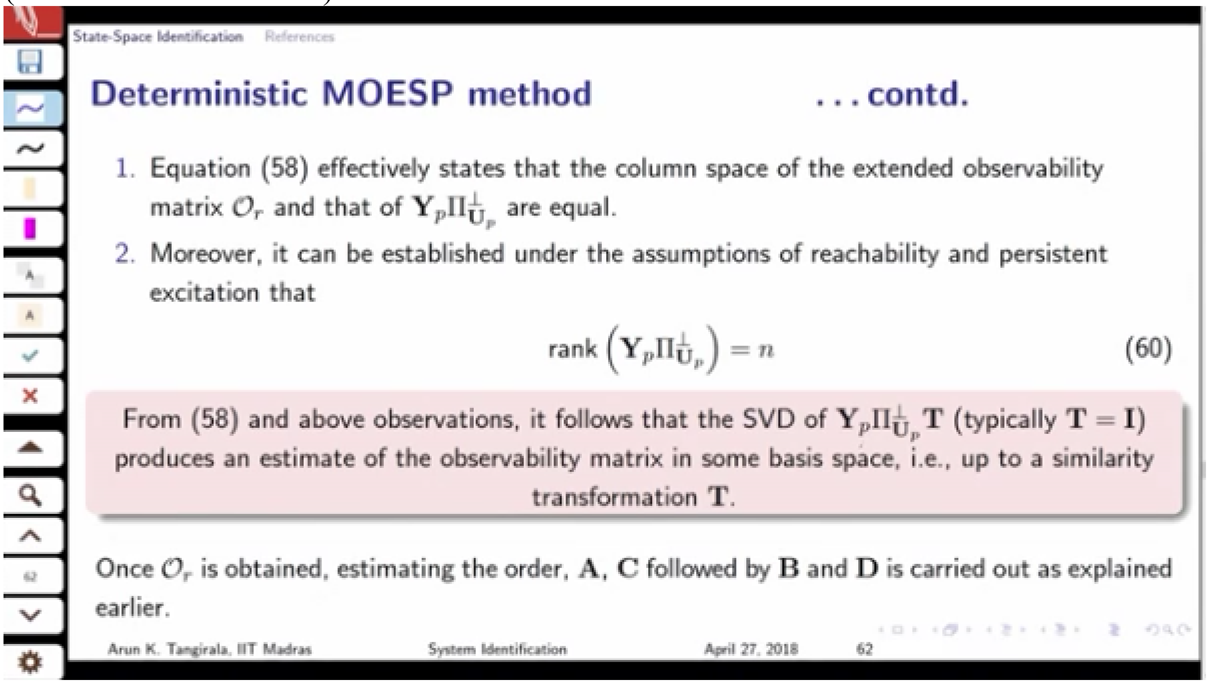
(Refer Slide Time: 15:17)



so one thing you should remember is YP times pi UP perpendicular is not going to be exactly OR, you will get OR times some matrix, that is what exactly amounts to saying, I will get a state space model in some basis, okay, so it has been shown that rank of YP pi UP perpendicular is N, and also that the column space of the extended observability matrix and YP UP perpendicular are equal, I'm avoiding those proofs you can refer to the text.

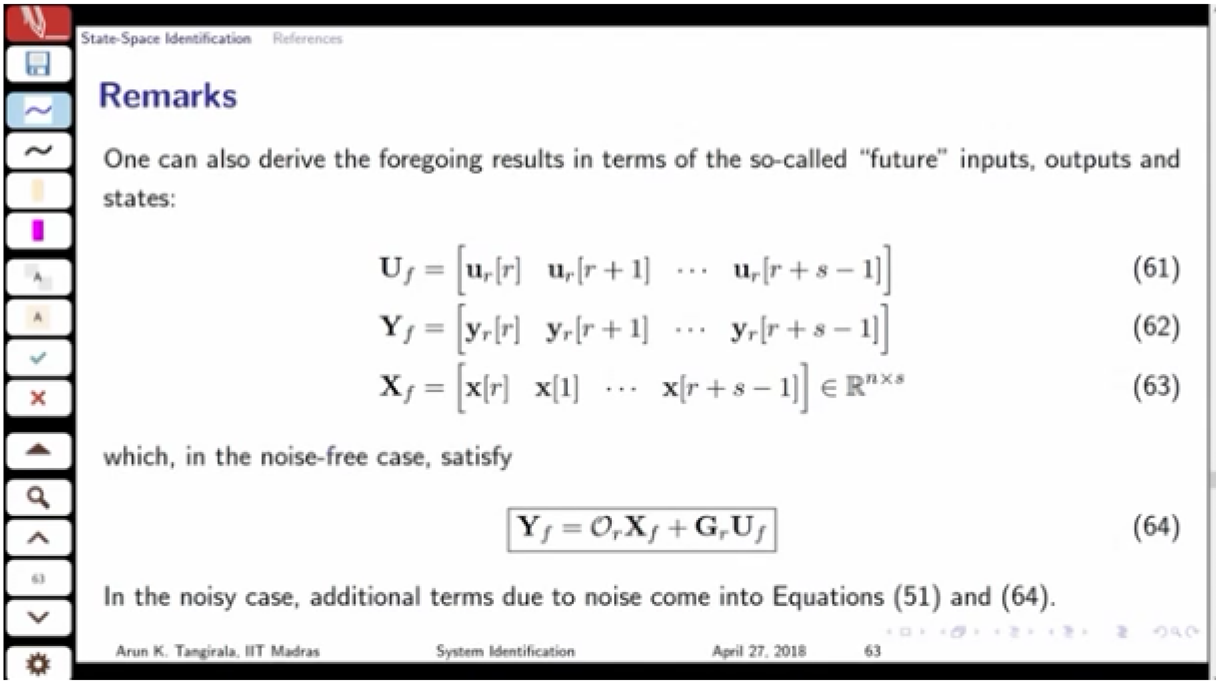
So the idea is simply compute SVD of this,

(Refer Slide Time: 16:00)



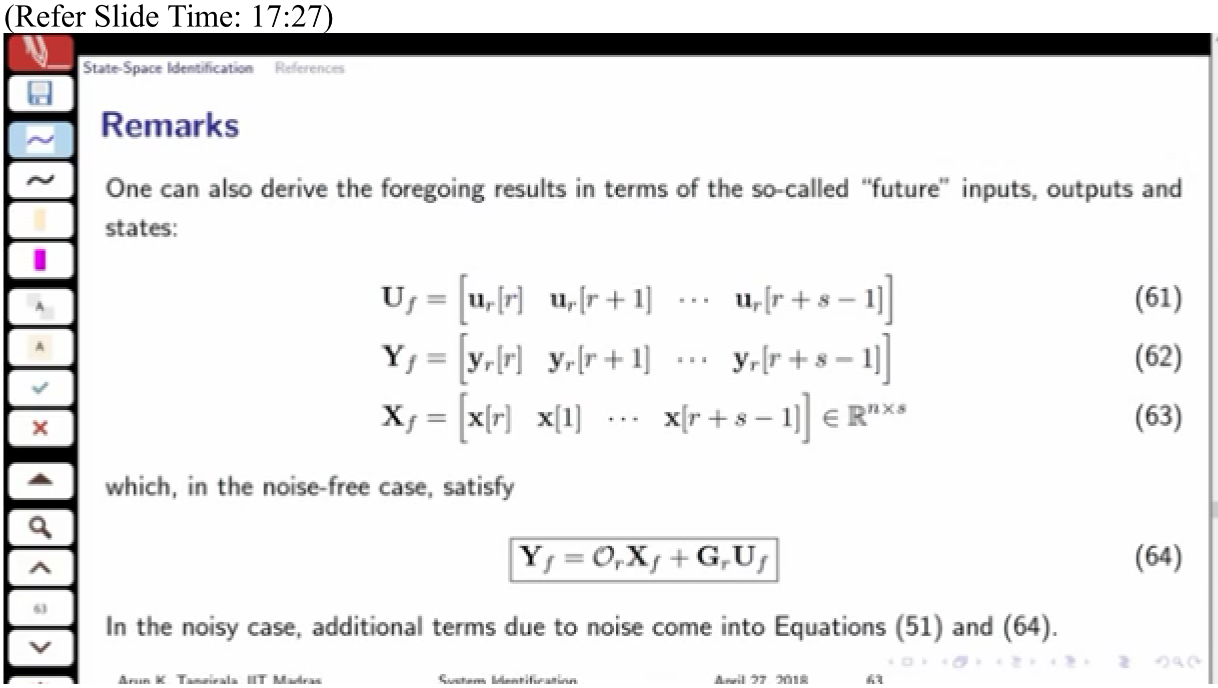
of course T is just to show that there is an ambiguity, you can multiply both sides with T and still you will get the answer and you will get some observability matrix, so all you do is step 1 construct this matrices Y and U, step 2 implement this, compute this orthogonal compliment through LQ factorization, and step 3 do an SVD of that, so you see the idea is more or less the same as Ho and Kalman's method, the only difference is now we are dealing with both input and output, there I didn't take the input into account at all, why is that? I've already taken into that account by writing a relation between G's and state space matrices, there the input is only one at time zero and zero later on, so the input doesn't appear at all, here the input comes into play because we are dealing with arbitrary input output data, that's all.

(Refer Slide Time: 17:07)



So you can also rewrite this in so called future states, and future inputs and so earlier we called this as past, so if you look at the matrices U, here we wrote as UR(0), UR(1) up to UR(S-1), right, you can also start at a different time and write it, so there it started off from 0,

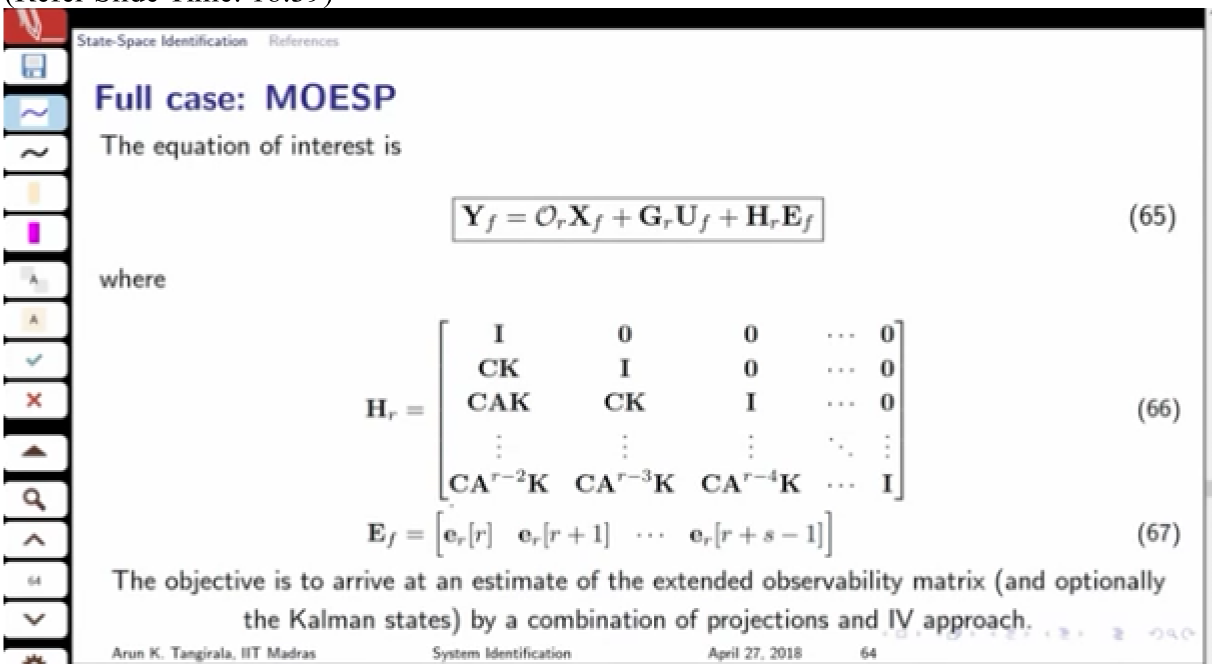
(Refer Slide Time: 17:27)



so what is done and if you read the subspace ideal literature the entire data that you have is considered to be divided into past and future and so on depending on where you start off, and there is a reason for doing that we'll not worry about it, but you can also essentially rewrite your algorithm, you don't have to start from 0, you can actually start from R and proceed up to R+S-1 the equation remains the same, it doesn't change, and you do the SVD and so on, so I'm not showing you the example for the MOESP deterministic is, straightaway we'll jump to the noisy case and then close the discussion on subspace ideal.

Now in the noisy case, how do you expect this equation 64 or similar one that you've seen before to change? What will be the difference? You've written this for the noise free case right? Yeah, so there will be an additional term where now the impulse response coefficients of the noise model comes into picture,

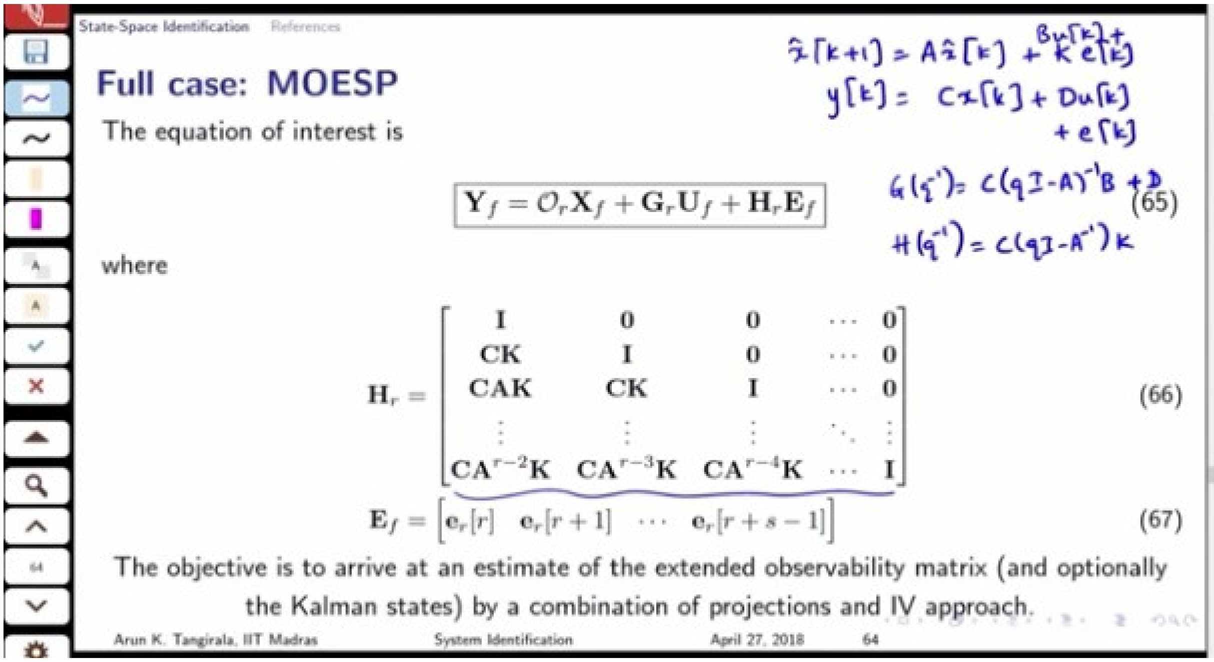
(Refer Slide Time: 18:39)



by the way the nice thing about the innovations form, you recall the innovations form that we had written, we had written innovations form X hat(K+1) A X hat(K), I'm avoiding, it's understood one step ahead predictions + some K prime will drop the prime, and we'll also drop the sub index on K and then we have Y(K) = CX(K) + DU(K) + E(K). I'm sorry, sorry, correct, thank you.

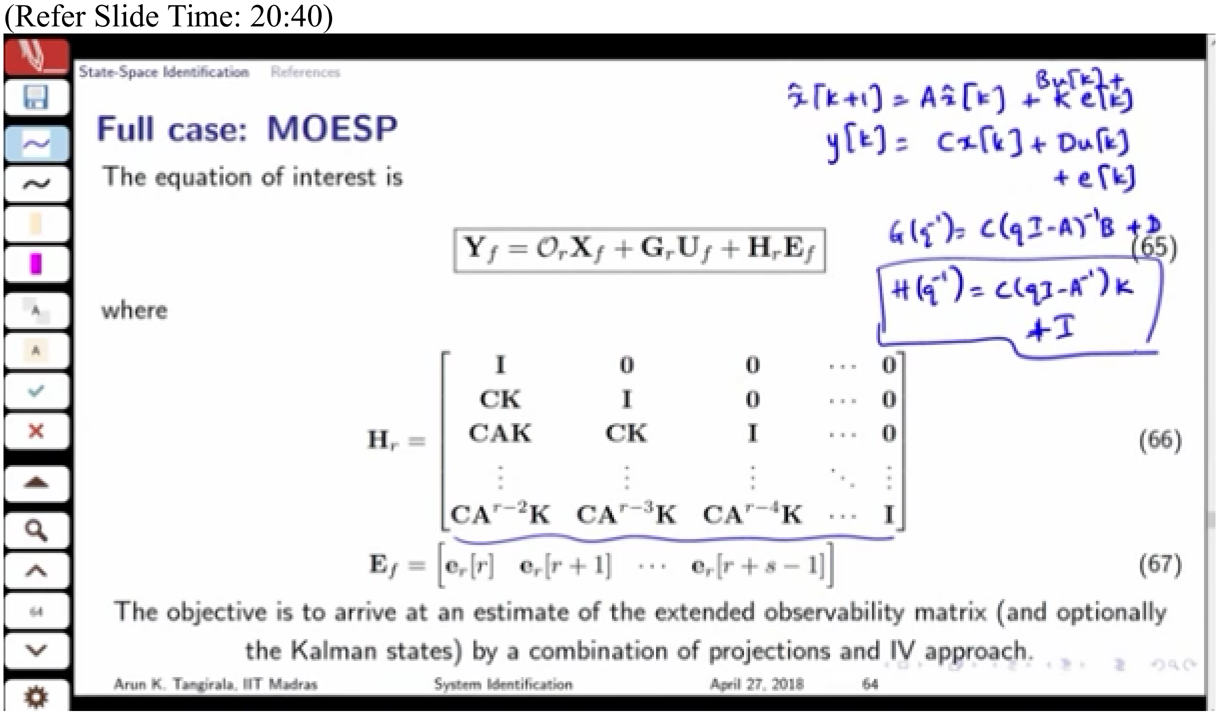
So now we know from this already that G(Q inverse) is simply C(QI-A) inverse B + D, the nice thing here is I can write the noise model H, why? Because already everything in terms of E(K), what would be H now? That's all, that's all? There is no identity? Plus identity?

(Refer Slide Time: 19:56)



All you have to do is what is the equivalent of D here for noise, do you think of noise is another input + identity, otherwise it will violate the assumption on time series models, correct, so we couldn't do this in the general state space description form, you remember we said I have to construct V(K) if it a time series model and then get my H. Whereas with the innovations form I can write the noise model straightaway and that's a beauty.

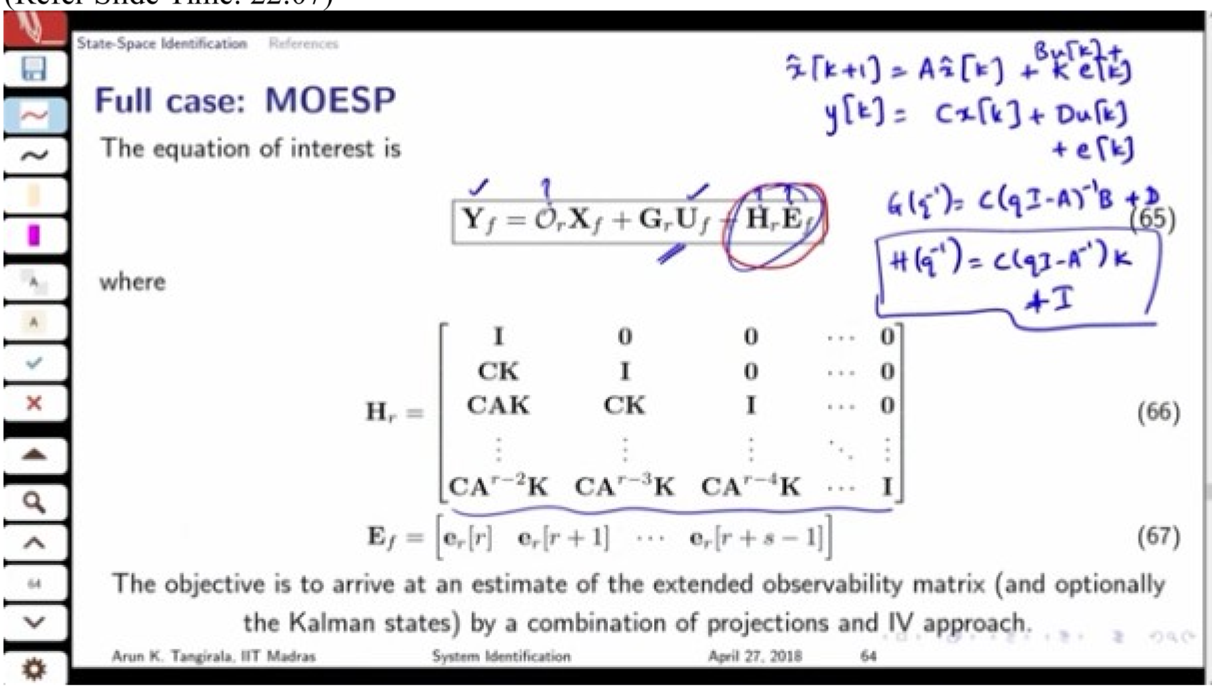
So earlier we had construct GR and we said that is a matrix of impulse response coefficients of G,
(Refer Slide Time: 20:40)



that things don't change, all you have, the way you look at it is you have an additional input called noise, earlier we didn't have this term, this term was missing, correct, now I have an additional input called noise which, whose the relation between Y and noise is determined by this impulse response coefficients, look at the GR and HR they have the same structure, but there is a difference, there are two differences, what are the two differences? Diagonal matrix is all identity whereas we had D there, and instead of B you have K, that's all, otherwise it doesn't make any difference.

Now what do I have to do? To get, again the story is the same, I'm given Y, sorry, and I'm given U and I want to get OR, but I've also not given H, I'm not given E, that is a challenge now, that is the major difference between the noise free case and the noisy case, in the noise free case I was able to kill this term, right, but now with the noisy case how can I kill this term, I don't have E, right that is going to be the challenge,

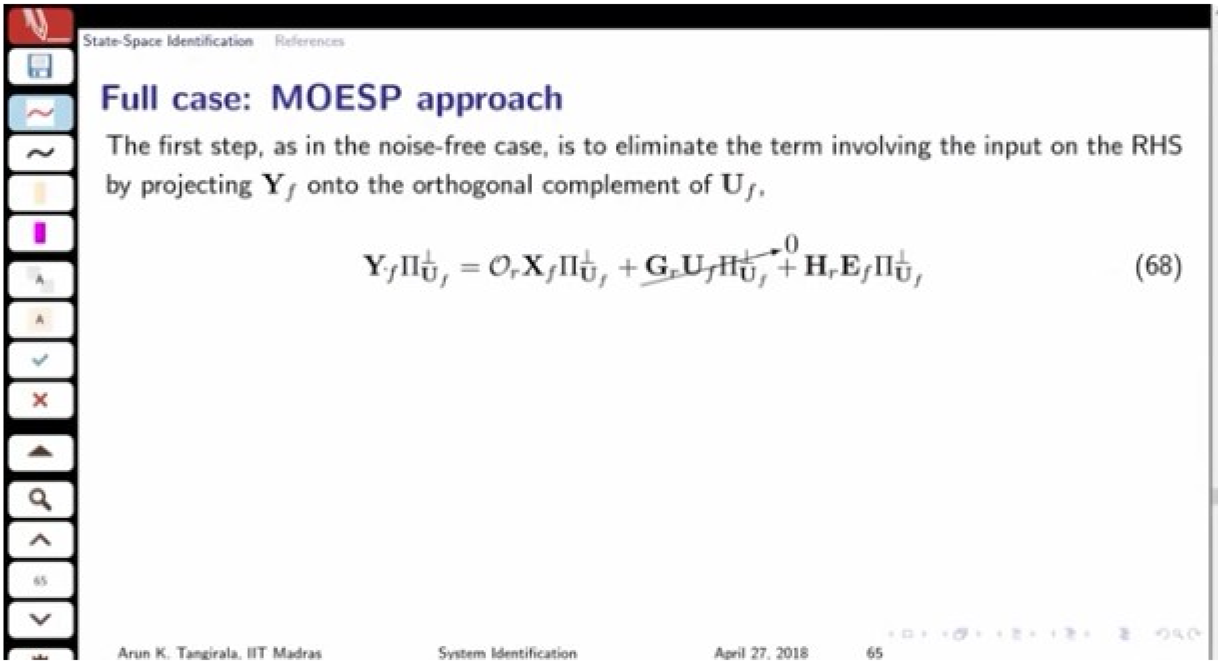
(Refer Slide Time: 22:07)



so let's see how we do that, that is the only step that is remaining and then we are done.

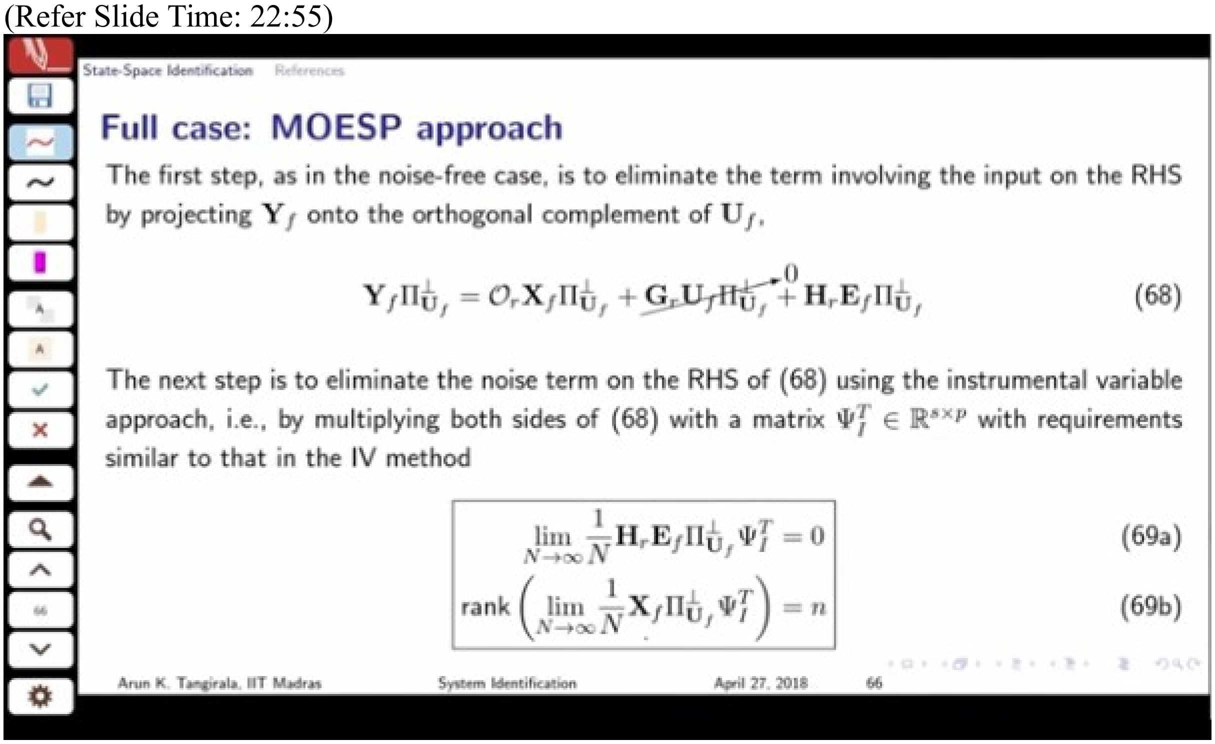
So as usual we kill the, we zero out the second term by projecting Y on to the orthogonal compliment of UP,

(Refer Slide Time: 22:22)



done, and then we are left with these two terms, how do I get rid of this term, any idea? How do I get rid of the noise term? So now we turn to an IV kind of approach, right, so I use instruments like we did in our IV approach for the input output models, i somehow seek instruments, some instruments called big sai,

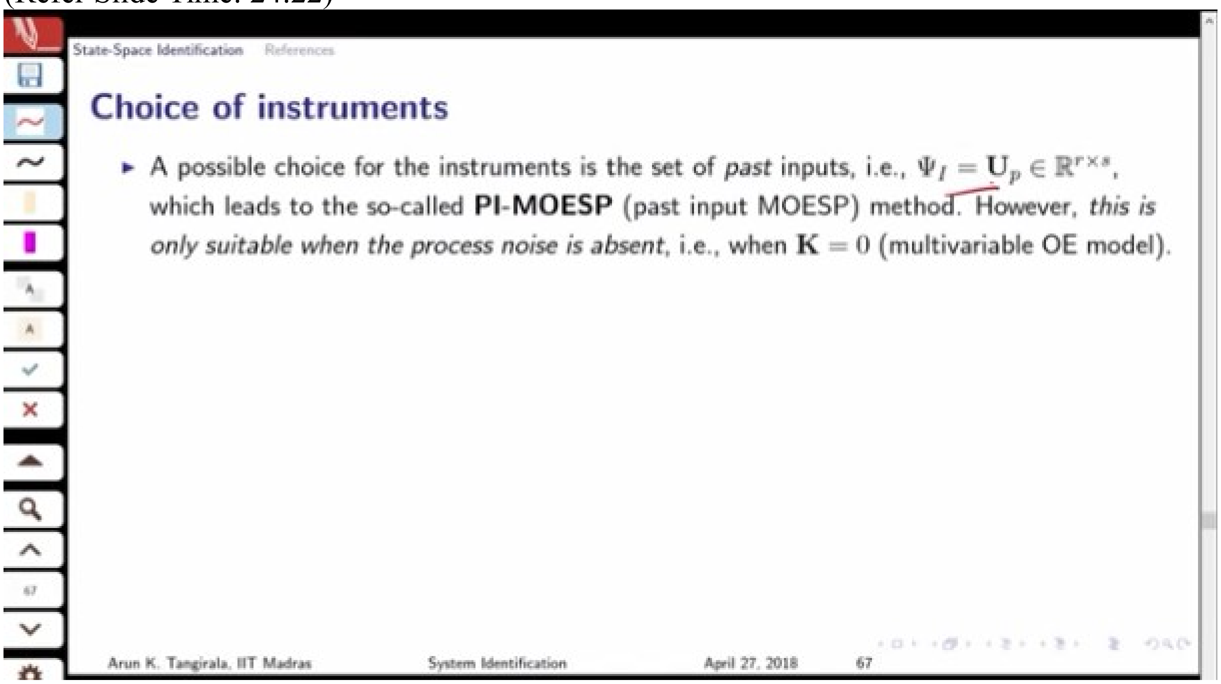
(Refer Slide Time: 22:55)



in some text books they use big Z, but that can be confusing because Z we use for input output data putting together and so on.

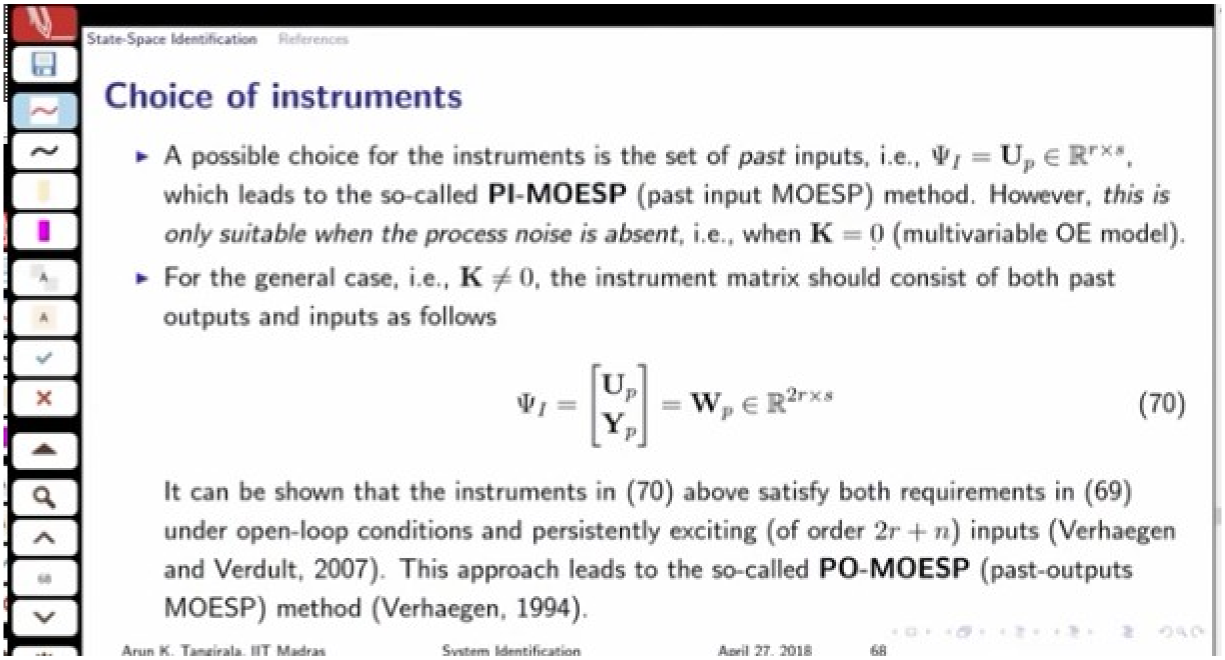
Anyway, so essentially I'm interested in this some instruments sai, same conditions that we have imposed in IV method, remember, it should be uncorrelated with the noise term and highly correlated with what I want to estimate, those are the exact conditions that we have imposed, so what is a natural choice of instruments? So of course you have to go into the structure of this matrices, I should choose the instrument such that this is satisfied, here we are writing, time domain version of the requirement but a theoretically requirement is it should be uncorrelated.

What does E consist of? Matrix of errors, and pi UF consist of orthogonal compliment of UF, right, so what is a natural choice of sai? The past inputs, if I use past inputs then I can expect this to go to 0, so we can also expect this to be a Franken, expect of course proofs are available, I'm just saying intuitively natural choice of sai is to use a past inputs, this is the idea you can call the past input MOESP,
(Refer Slide Time: 24:22)



so now you see the subscripts you have to notice carefully, this F here is future, because we have done a partitioning of data into past and future, now you understand why we are doing that, the instruments are now going to be based on past inputs, alright, and it can be shown that with this choice of instruments these two conditions are satisfied.

But generally this choice of instruments is recommended again I'm avoiding the proof, when the process noise is absent, okay, that is when K is 0, that means for output error models it is ideally suited. What do we mean by only suitable? I'll talk about it shortly, in the general case when the process noise is also present,
(Refer Slide Time: 25:19)

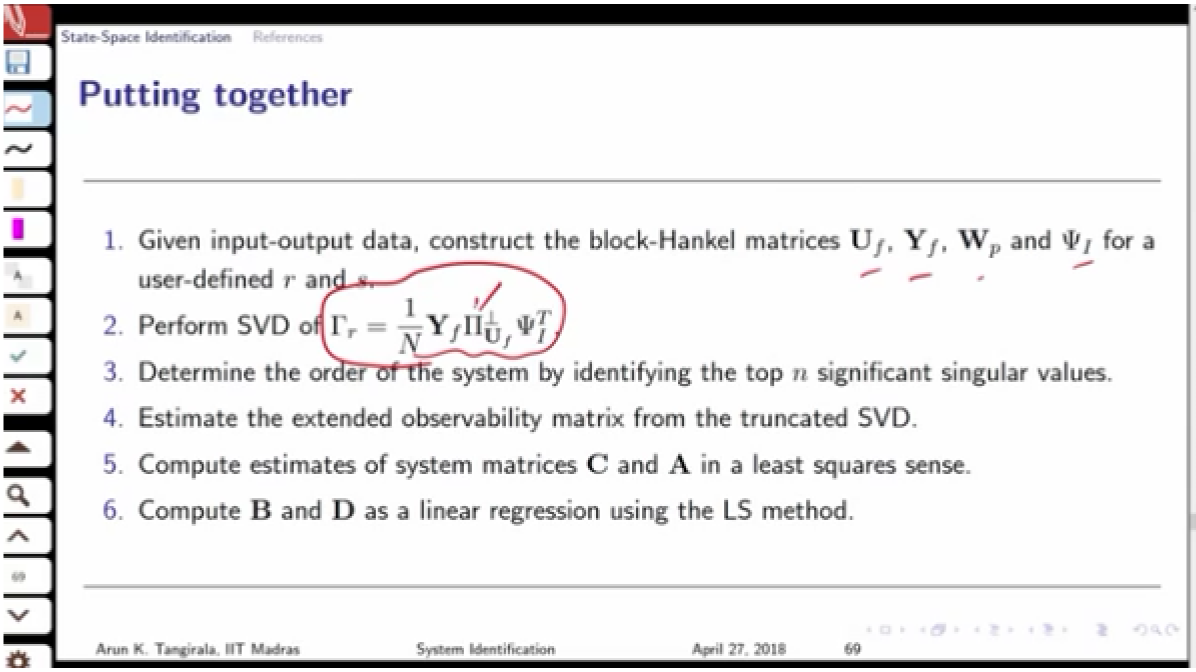


in the first case we choose the instruments to be U, and you can choose that to deal with measurement noise cases, the measurement noise could be white colored and so on, but whenever the process noise is present it is recommended to use these instruments, of course you will have to use a dimensions carefully here, although I write here UP YP, you will have to be careful with the dimension so that it matches things, and remember the dimension of here sai F is 2R/S, the dimension of UP is R/S, and I'm multiplying here with transpose, so the sai transpose is the same in both this cases, the sai transpose has the same number of rows S, it's only the columns that are different.

I have fewer columns here in the first case, and more number of columns in the second case, but that doesn't matter, the matrix multiplication comparability will be maintained that is not an issue, anyway, so it can be shown that again this choice of instruments where I'm choosing past outputs and past inputs as the instruments, they'll satisfy the requirements under open loop conditions provided the inputs are persistently exciting, which is something that we'll briefly talk about after we are done with this. So this approach leads to what is known as the past output MOESP.

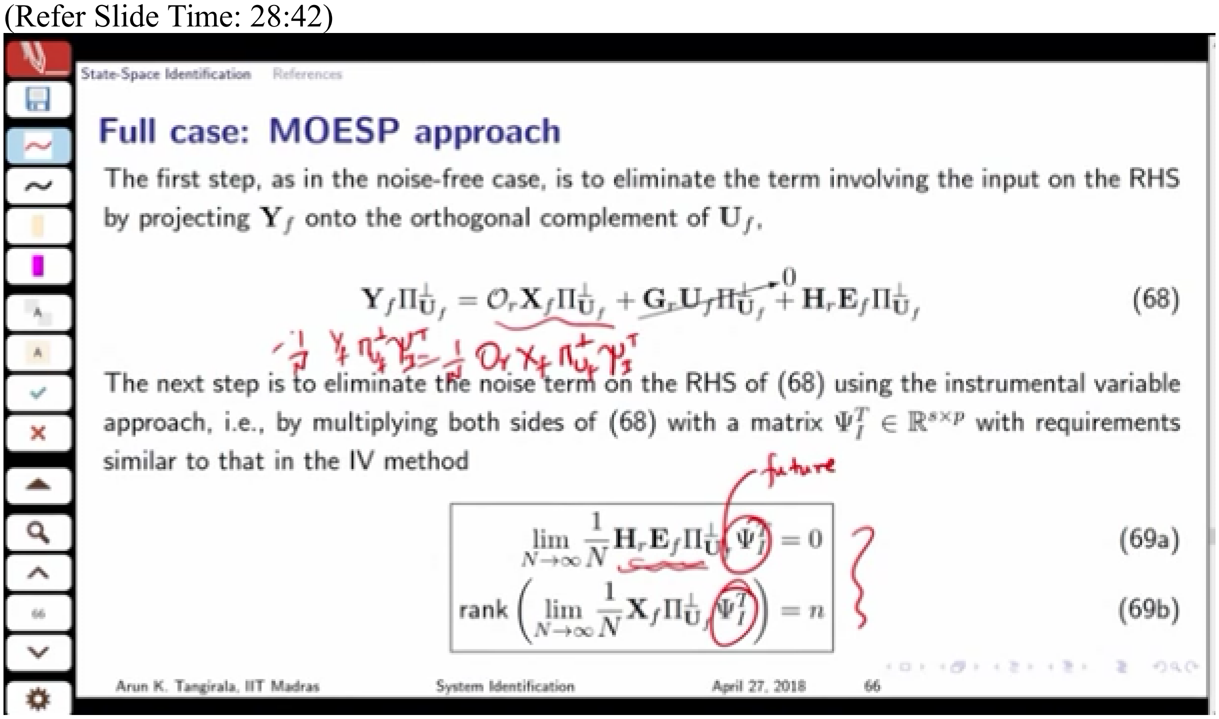
So you understand now the idea you put, you think of Ho and Kalman's method, and you think of subspace ID there are lot of similarities, in subspace ID we do a smart way of avoiding the impulse response calculations by zeroing out the term using projections for noise free case, for noisy case we have an additional step of using the, setting up the instruments and constructing the instruments and correlating the data with the instruments, so that's it, so this is the step putting together you construct these matrices and these instruments, perform SVD of this matrix. Now this matrix is important, important right, the first step multiplication is to zeroed the effects of inputs,

(Refer Slide Time: 27:44)



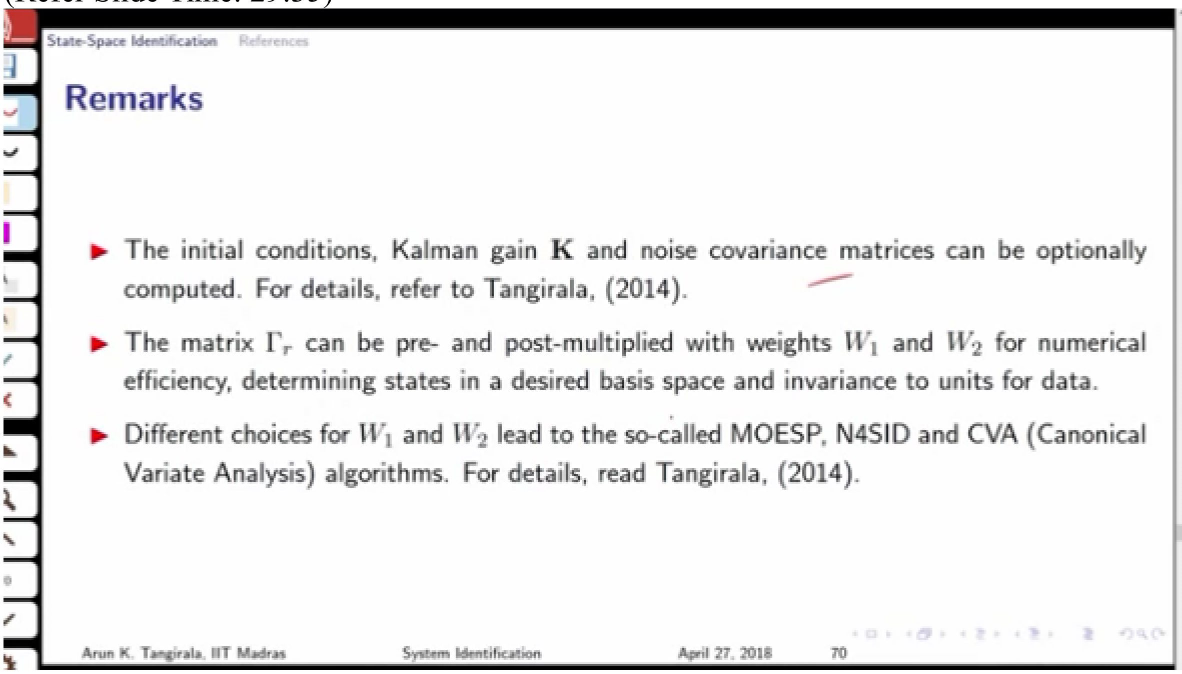
the second step multiplication is to killed the effects of noise, so I do an SVD, what will I be left out with? If I do that, if you go back to this equation, when I multiply both sides with sai transpose this term will come OR XF, I don't worry about this term all I have to show is that the column space of the left hand side, sorry, is the same as a column space of OR, that's all I have to guarantee and that is shown, and that the rank of this is N, that is also shown, the rank of the left hand side is the same as a rank of, is the same as order of the minimal realization.

(Refer Slide Time: 28:42)



So all I do is now construct this left hand side matrix, because I know Y, I know U, I know my instruments, I can construct it, I do an SVD which will give me the rank and also the factors that I want, observability matrix essentially here notice the controllability matrix doesn't come into picture at all, why? Because I'm not estimating B here, I'm only first interested in estimating OR, so that is another difference between Ho and Kalman's method and this method here, alright, so do that than determine the order, extended observability matrix, estimate C and A like you do in Kung's method and then estimate B and D as a linear regression problem, that's it, right, so the initial conditions Kalman gain and noise covariance matrices can be also computed,

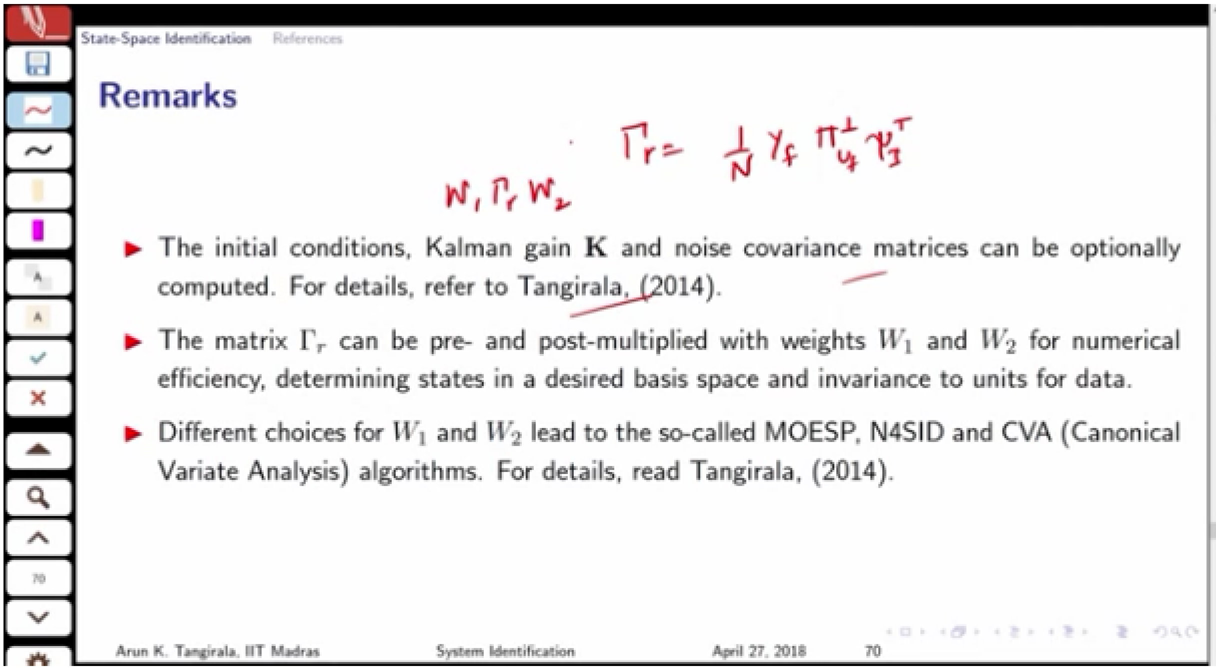
(Refer Slide Time: 29:35)



you can refer either to this book alright or much better books perhaps on subspace ID for details, and the other thing that you should remember is we are computing this gamma R which is 1/N, do you remember? YF pi UF perpendicular and sai I transpose, I can always pre-multiply and post multiply this with suitable matrices without changing the results, results in the sense without affecting the order analysis, the A and C will be affected.

The moment I pre–multiply and post multiply gamma R with some weighting matrices, what I am doing is essentially changing the basis in which I'm identifying the state, that is one thing and it can also be shown that this pre and post multiplication helps achieve,

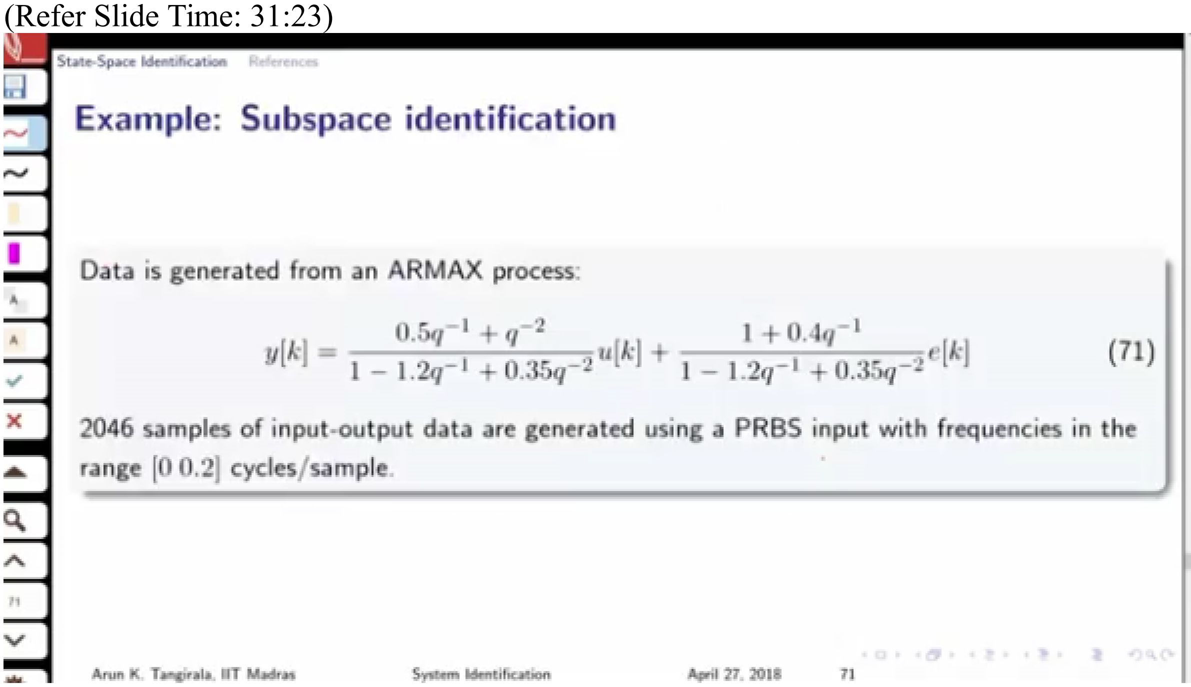
(Refer Slide Time: 30:37)



depending on how you choose your W1 and W2 you can achieve invariance to the choice of units of Y and U, see that is another thing that you have to always watch out for in data analysis, which is that the method can be sensitive to the choice of units for Y and U, always you should choose a method that is invariant to the choice of units.

What has been shown in the literature that CVA, the canonical variate analysis among the three has this invariance it chooses W1 and W2 in such a way that whatever units you choose for Y and U the results won't change, and that it is also robust, more robust than MOESP and N4SID, so that is the basic idea.

(Refer Slide Time: 31:23)



Online Editing and Post Production

Karthik
Ravichandran
Mohanarangan
Wilhelm Benjamin Anand
Sribalaji
Komathi
Vignesh
Mahesh Kumar

&

Web-Studio Team

NPTEL Co-ordinators

Prof. Andrew Thangaraj
Prof. Prathap Haridoss

IIT Madras Production

Funded by
Department of Higher Education
Ministry of Human Resource Development
Government of India

www.nptel.ac.in