# CH5230: System Identification

## Estimation of parametric model

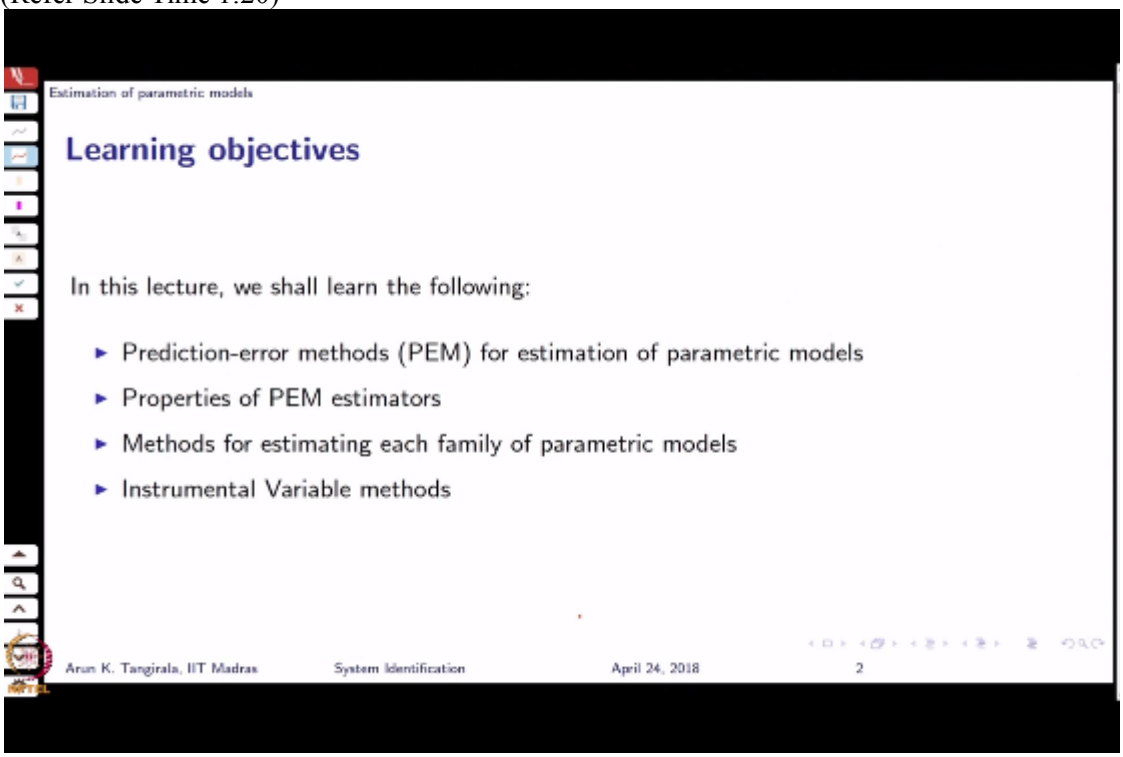
## Lecture 49 Part 1

So welcome to the lecture on estimation of parametric models. In the previous lectures, we have learnt different facets of estimating non parametric models. And that was a right stepping stone because as you know in system identification, one begins with non-parametric models, gathers quite a bit of

insights into the process characteristics, which then become useful for fitting parametric models. And you should recall that the reason, primary reason for turning to parametric models is parsimony. Right? Because non-parametric models simply involve a lot of unknowns to be estimated, of course, the advantage is that you make minimal assumptions. With the insights gathered from non-parametric models, we are in a much better position to estimate parsimonious parametric models, and we have already studied the parametric model family.

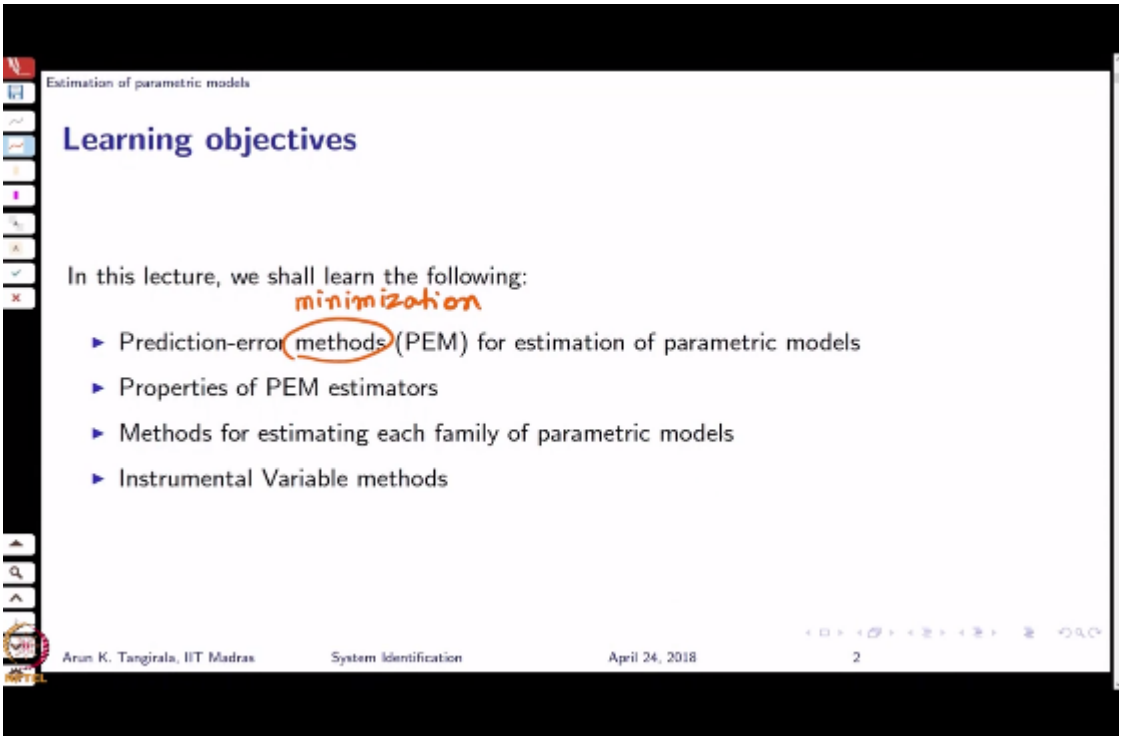
So the learning objectives of this lecture is to first study prediction error methods for estimation of parametric models. And then we'll discuss properties of this PEM estimators. In fact, PEM should ideally stand for Prediction-Error Minimization.

(Refer Slide Time 1:20)




So the methods here should denote, actually it should be, replaced with minimization. And we will also briefly study the methods for estimating each of the family of parametric models that we have learnt earlier, namely the ARX, ARMAX, OE, BJ models. And then we'll conclude this lecture with a discussion on the instrumental variable methods for which we got to peek into when we were discussing non-parametric model estimation.

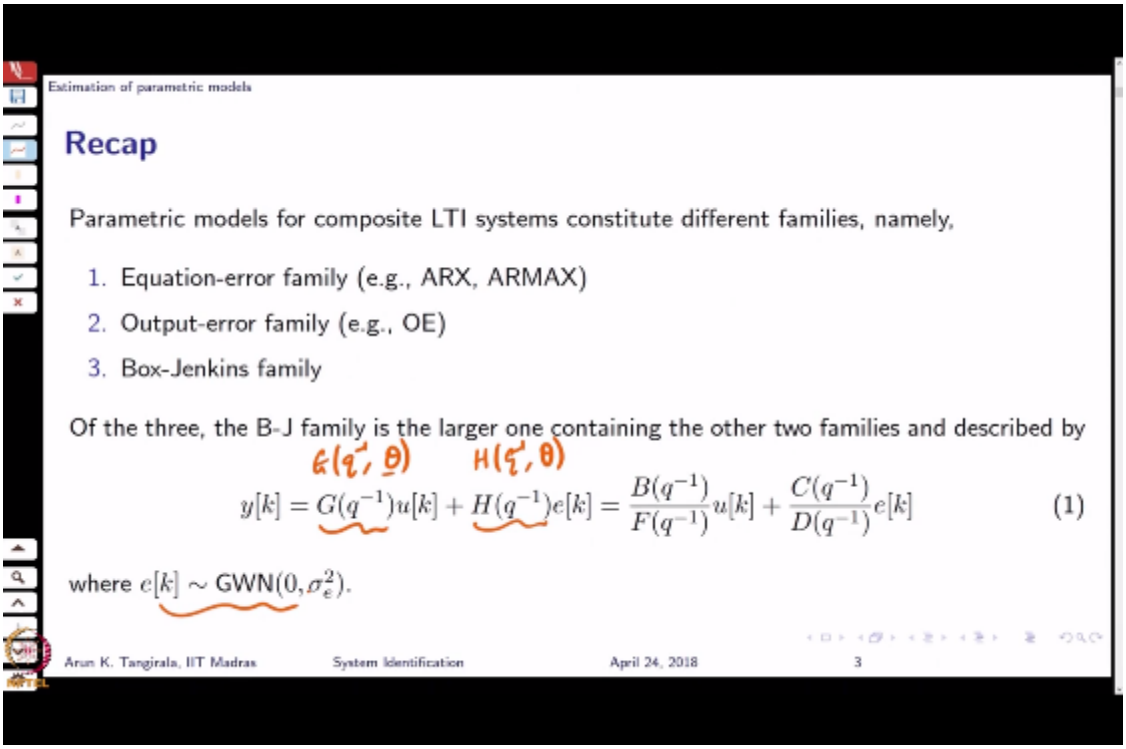
(Refer Slide Time 1:58)




So let's recap the different parametric models that we had looked at before for composite LTI systems. By composite, I mean, deterministic plus stochastic. So we looked at the Equation-error family, especially the ARX and ARMAX class of models. Then the Output-error family, which is somewhat contrasting to the equation-error family. And then the more generic Box-Jenkins family. Again, you should remember that these different families arise based on the kind of assumptions that we make on the plant and noise models, how we parametrize them.

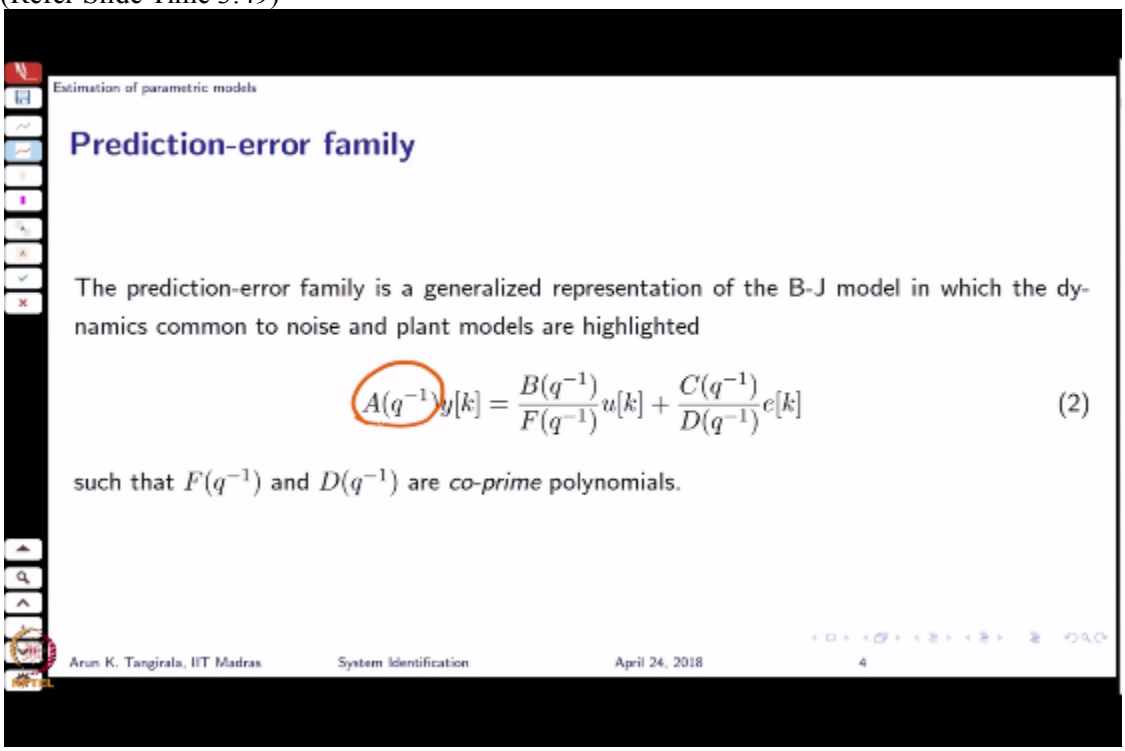
So, strictly speaking, again this G of q inverse should be written as G of q inverse, theta, where theta is a vector of parameters that we are estimating. Likewise, H of q inverse should be H of q inverse, theta. So as to explicitly state that we are now dealing with parametrized forms of this plant and noise models. And the B and F, as you know, are polynomials in shift operators and likewise are C and D for H. And e is our usual Gaussian white noise. Mean 0, variance sigma square e. The goal is given input output data.

(Refer Slide Time 3:16)




We would like to estimate these polynomials. We'll make the more formal statement a bit later. The more, I would say a comprehensive prediction-error family or the parametric family, is this so called PEM structure as Ljung calls it, where we have now factored out the common polynomials between denominated polynomials between the plant and noise models and collected them in A of q inverse.

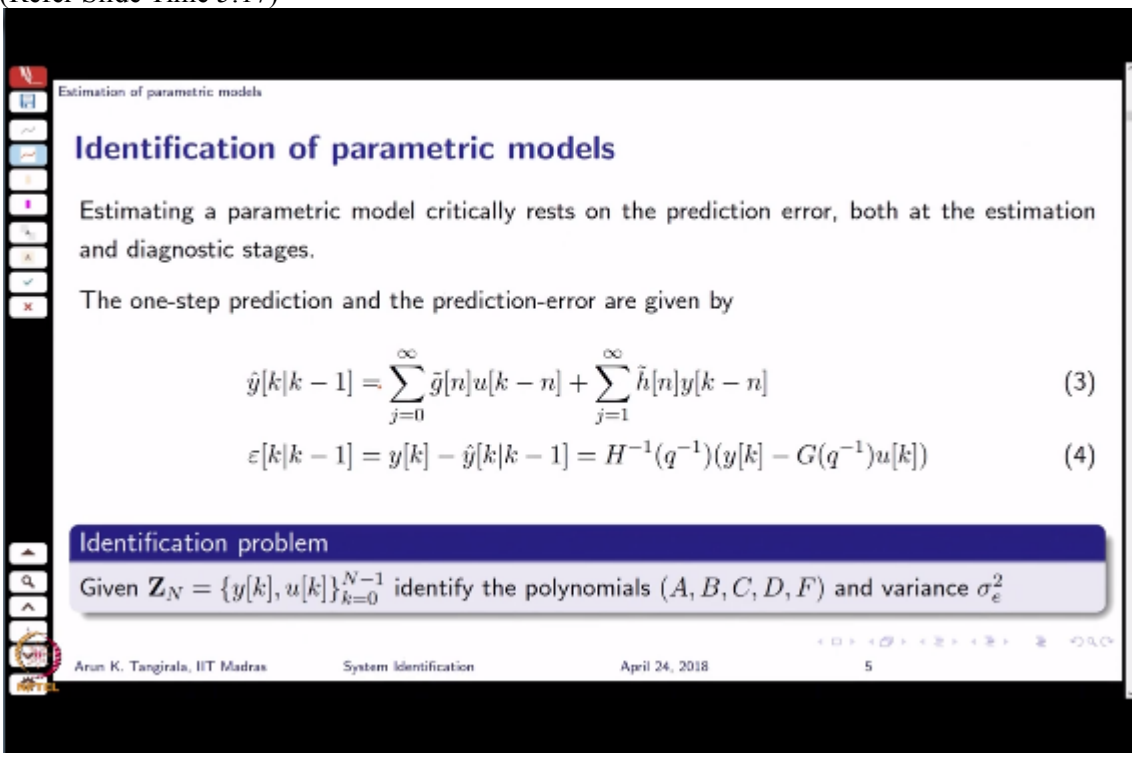
(Refer Slide Time 3:49)



So A of q inverse, as I had pointed out in one of the lectures earlier, this is a common, I would say, characteristics to both plant and noise models. The F and D are now unique. They carry unique signatures of the plant and noise models. The other way of stating is the same thing is that F and D are co-prime polynomials.

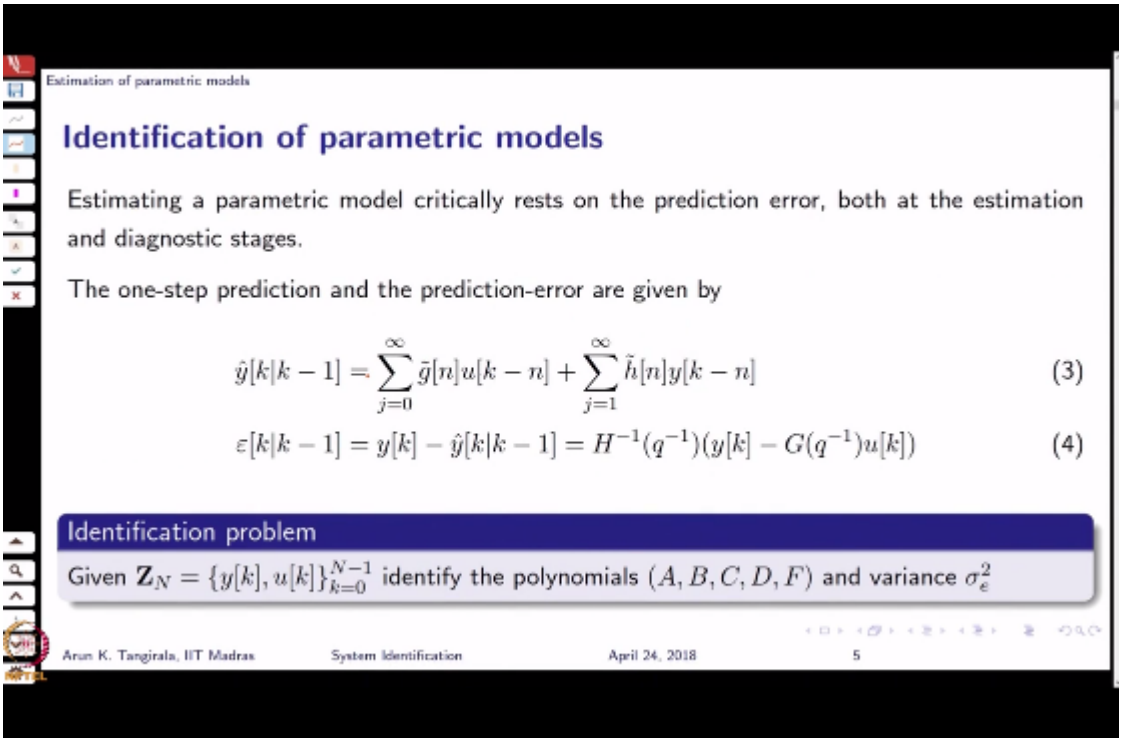
So I have also explained in one of the earlier lectures why we would like to factor out the common one, because that is also amounting to feeding in some prior knowledge. So, for example, if you go back to this structure, this Box-Jenkins structure, there may be commonality between F and D, which is not really highlighted. And if you don't supply that information, there will be additional number of parameters that would be estimated as against the other case, where I specify that there are some common factors to F and D, so that there is no repetition of parameter estimation. So, obviously specifying this common factor amounts to bumping in some additional information and which will reduce the variance of the parameter estimates.

(Refer Slide Time 5:17)



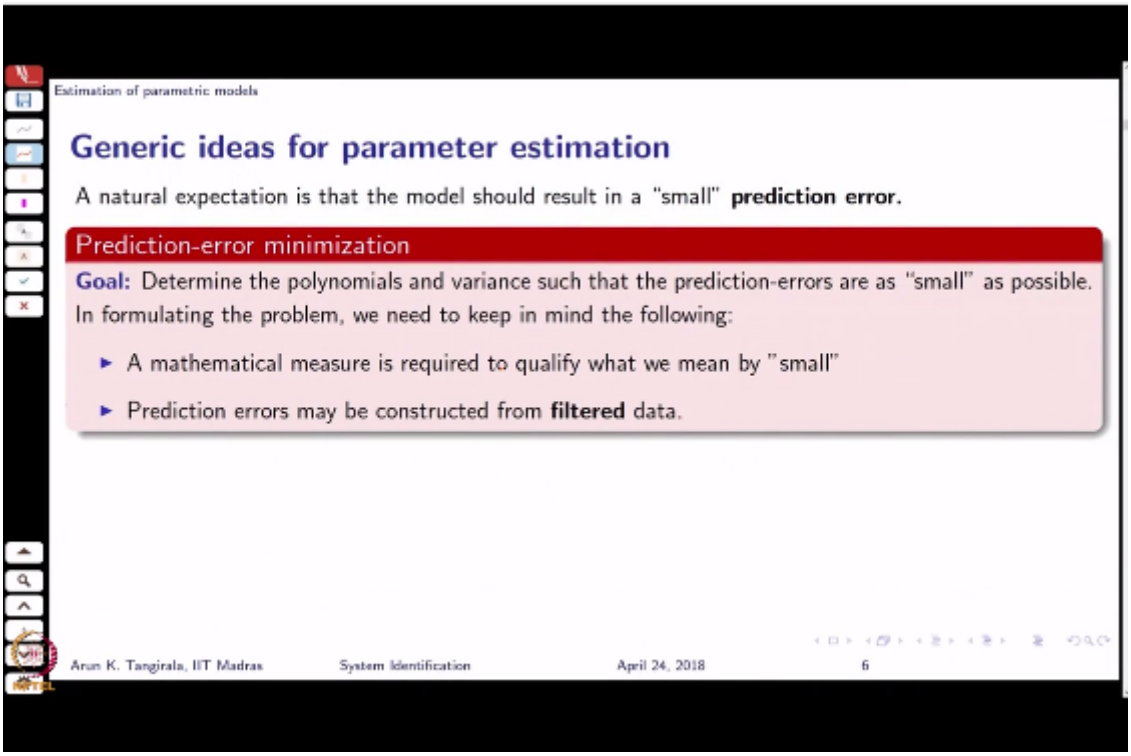

So we know very well, depending on the assumptions that you make on A, B, C, D, F, you generate ARX or ARMAX output-error, Box-Jenkins models, and so on.

(Refer Slide Time 5:29)



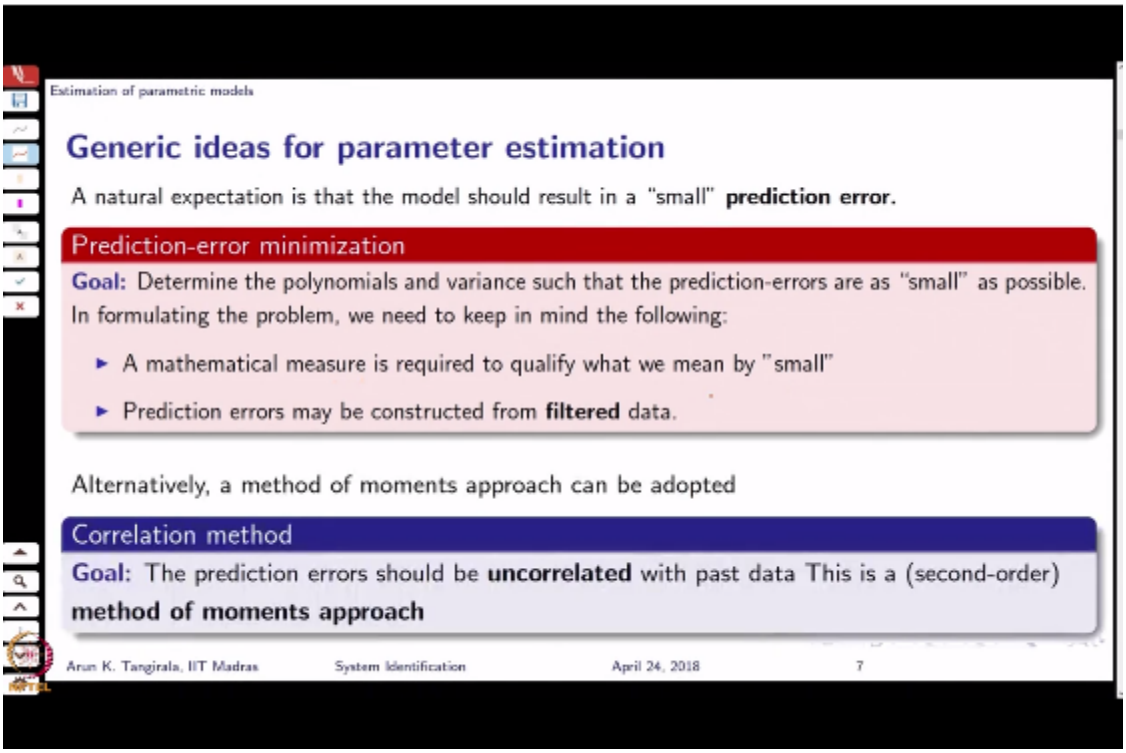

So the statement of identification of parametric models is as follows, given Z identify the polynomials A, B, C, D, F, and variance sigma square e. This is a very, the statement is a one liner, which means obviously there's a lot of work to be done. Usually the problem statements if they are big than the work done, that has to be done is less. So that's called the law of conservation of the problem statement and the solution together, the length of it. And generally estimating a parametric model critically rests on the notion of prediction error. And here we give the expression, we have derived this expression for the one-step prediction long ago. And consequently the one-step ahead prediction error, in terms of, of course, now this equation 3 and 4 are in terms of the impulse response coefficients. We had derived also in terms of G and H, which I'll show you later on.

(Refer Slide Time 6:36)




So there are broadly speaking two generic ideas for parametric model estimation. One class of methods, one idea pursues this so-called prediction error minimization, where the goal is to – or the idea is to determine the polynomials and the variance sigma square e, such that the prediction error is minimized, in some sense, it could be in at least square sense, in a one norm sense and so on. Of course, that means that we need a measure to quantify what we mean by small and also we can remember that the prediction errors may be constructed from filtered data. So we'll use that a bit later. We'll use that idea of it later. So broadly speaking now, this idea here says that I will identify the polynomials in such a way that the prediction errors are small in some mathematical sense.

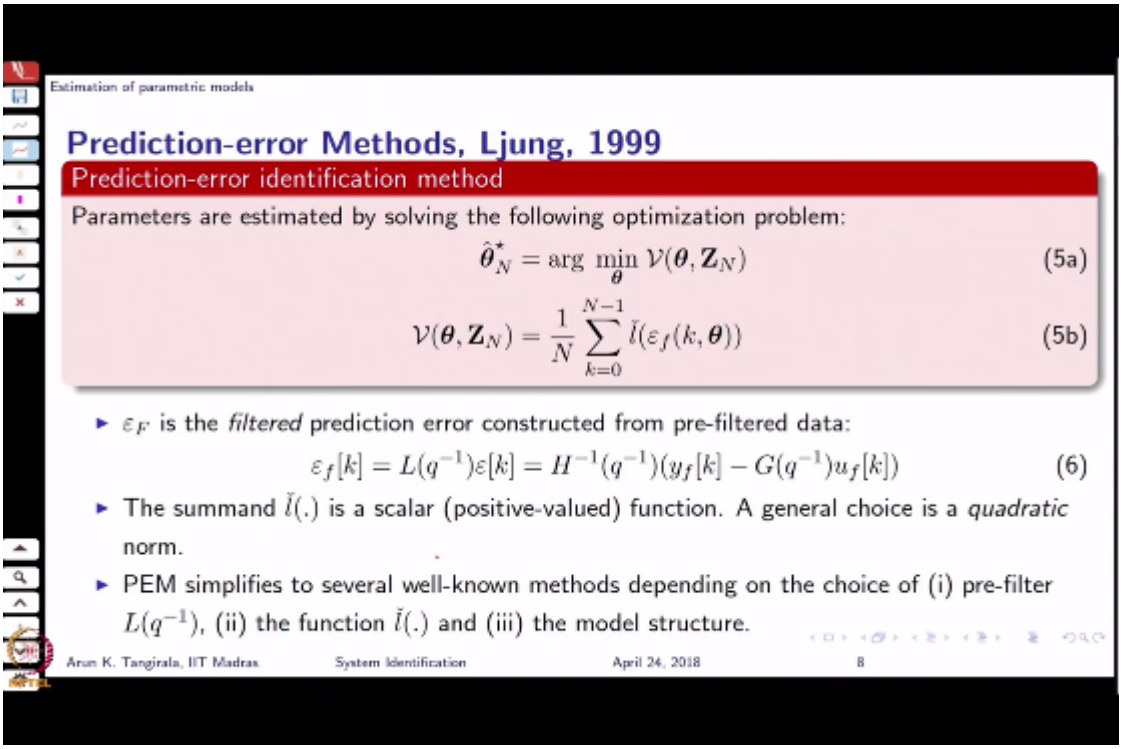
(Refer Slide Time 7:39)




The other class of methods that take birth from a different idea or call it correlation methods, where the requirement is that the prediction errors be uncorrelated with the past data. So this requirement is quite different. This is more on the method of moments idea, where we are looking at the second order moments. You could use other moments, too. You can say, you can impose a stronger requirement. You can say, for example, that I would like to estimate the polynomial such that the prediction errors are independent of the past data, which means I'm ruling out non linear relations and so on, but we don't need to in the linear world.

Now, the instrumental variable method that I briefly mentioned earlier belongs to this class of methods. Normally, one encounters the prediction error minimization methods in system identification. There are other ideas as well. You could look at those, but these are the, broadly speaking, these are the two methods that you would encounter. And many times the correlation method is used as an initialization method for kick-starting a prediction error minimization methods, which I had briefly mentioned also in one of the earlier lectures.

(Refer Slide Time 8:57)

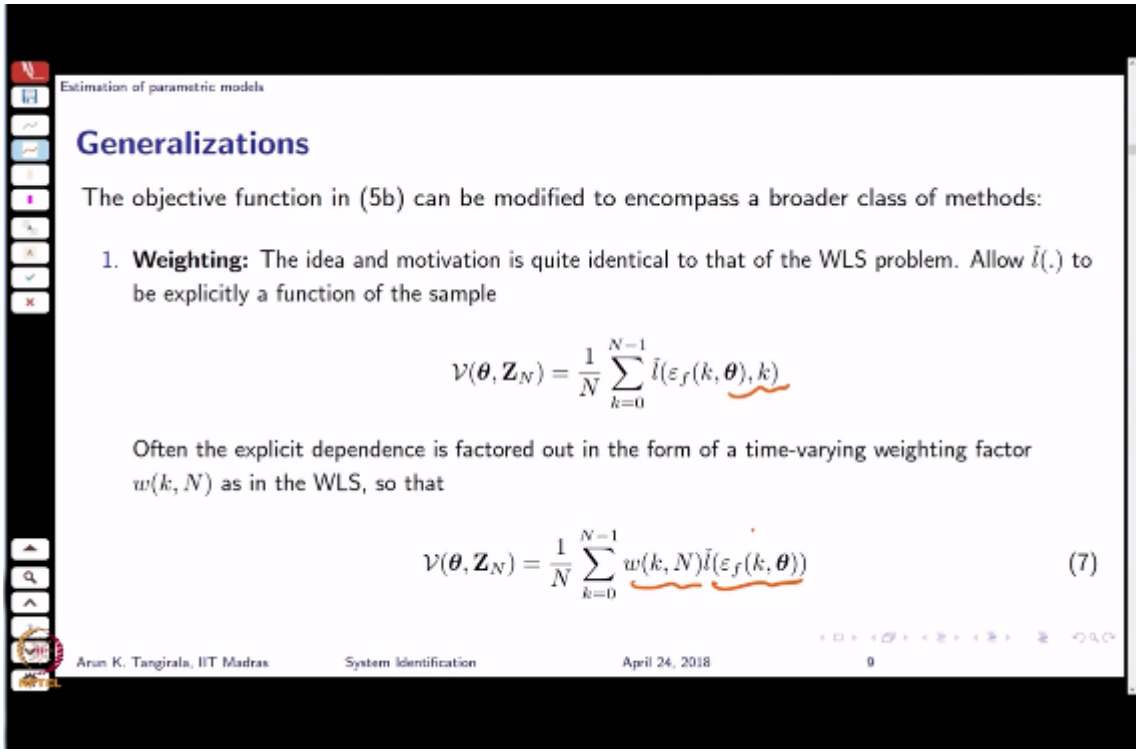



So let's spend some time on the prediction error methods because -- minimization methods when I say prediction error methods, you should remember that we mean prediction error minimization methods. Because these are very common, very popularly used and this was largely, these methods are largely advocated by Ljung in the ‘90s and even a bit earlier, where the parameters are estimated by solving this optimization problem. So if there is a cost function that we are minimizing and that cost function here is some function, okay, where this l inverse cap is some function of the filtered prediction error.

So it's a very fairly generic one, l inverse cap can be anything that you choose. And that will, of course, as we know from estimation theory, that will go on the quality of estimates and you could optionally choose to not have the filter, in which case the subscript on F on the prediction error will vanish. So there are several classes of methods that can be brought under this umbrella of prediction error minimization methods or PEM methods.

And the decision variable is of course theta. I didn't point out sigma square e, but the sigma square e can also be estimated through these methods. The general idea is to first estimate theta and then estimating sigma square e. You can do that or you can estimate them simultaneously. All right. So this epsilon f, I’m sorry, there is an upper case f here, you should have a subscript f, is the filtered prediction error constructed from pre-filtered data. We have talked about pre-filters earlier. In fact, we have shown that many of them -- model families, for example, how the output-error model can be viewed as ARX model on pre-filtered data, where the pre-filter is simply one over N. So the user may have pre-filtered the data for various reasons, maybe to remove noise, or maybe to focus on certain frequency ranges and so on. And that pre-filter is denoted by l, right? And PEM simplifies several well-known methods. That is what I meant earlier, depending on the choice of the pre-filter and the function and the model structure that you choose. Mostly depending on the pre-filter and the function.

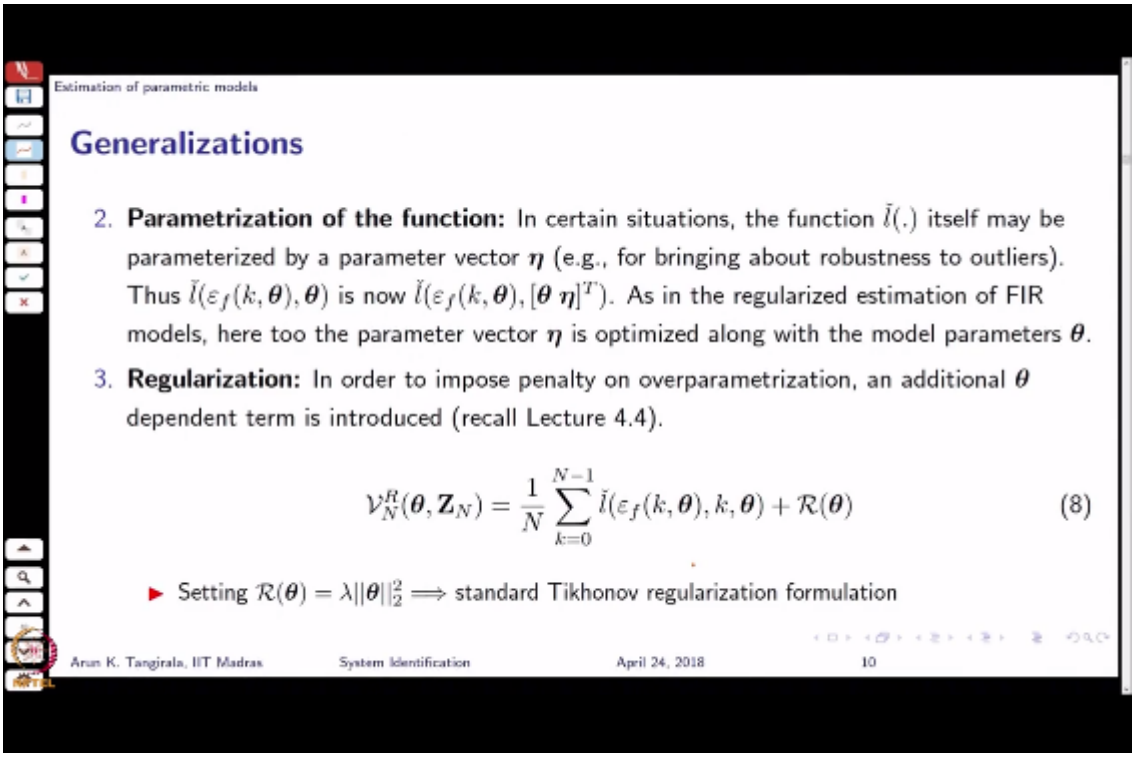
(Refer Slide Time 11:14)




So let's discuss some options that are available in PEM. PEM is a very broad approach, as you can see now, you are just minimizing some function of pre-filtered prediction error. You can, in addition, modify that objective function to include some kind of weighting, which means that I may not give same importance to all data points, I may use a different weighting. So, for example, in weighted least squares I do that, in which case l cap now, or the l inverse cap is not only a function of the epsilon f, but also explicitly a function of the observation that you're looking at. And that is what essentially this means. This explicit dependence can be factored out as in the weighted least squares case this way. So you could do a separability, you could invoke a separability assumption and think of this l inverse cap as a product of two functions. One is a weighting function and other is your usual l inverse cap. So this is the basic idea in weighted least squares, for example.

You must have guessed by now what should be the choice of l inverse cap to obtain least squares, what should it be? That's all, so it's squared to norm, right? So, in fact, since we had seen l inverse cap at a single observation, we don't generally talk about norms unless epsilon at each observation itself is a vector, that will happen usually for MIMO systems. So, instead of saying to norm, essentially it's a square. If l inverse cap is a square and there is no filtering, then the weighting is one, that it reduces to a least squares problem. All right, so let's move on.

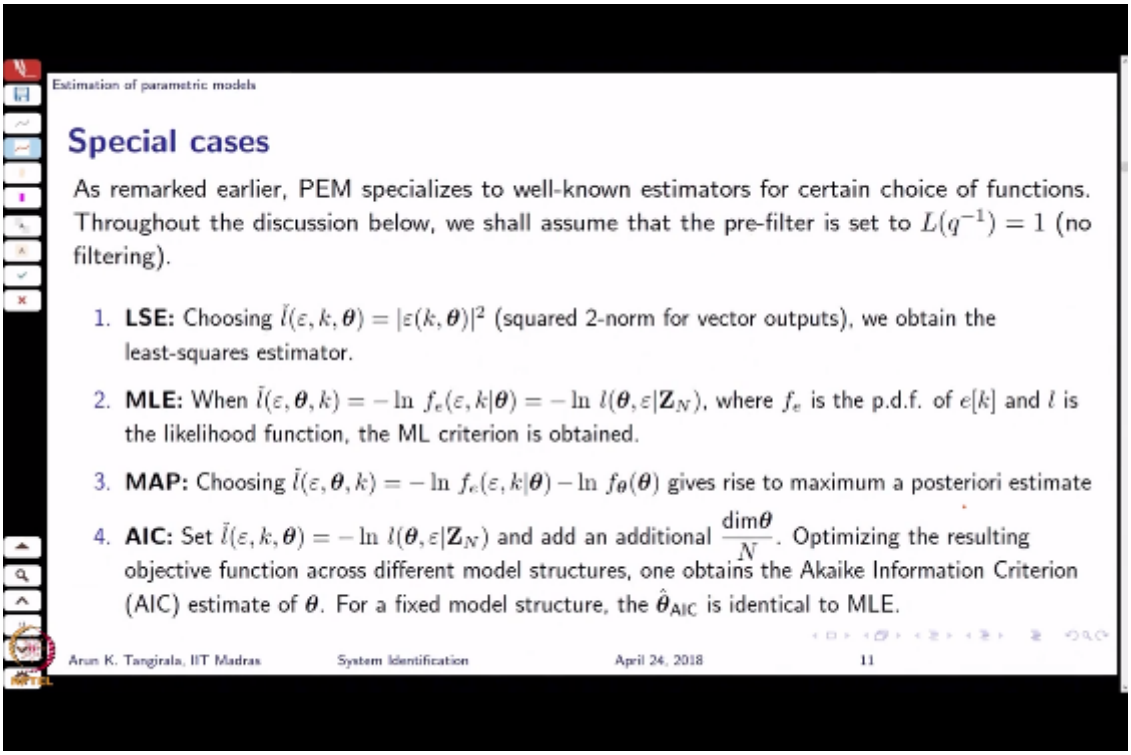
(Refer Slide Time 13:40)




The other option is to parametrize the function. In certain situations the function itself maybe parametrized by another vector eta. So you already have the parameters of the model, but the function itself could be now a function of some other outer set of parameters. For example, that you may have to bring about robustness to outliers and so on. So in other words, now the l cap is a function of this augmented parameter vector. And if you are going to apply some regularization, then the new parameter vector is also going to be optimized. So that is one option. These are all extensions of the PEM, or you can say embellishments that you can make.

The third embellishment that you can make to the PEM is regularization, like we did in least squares. So what you can do is, we can say that essentially the cost function now includes regularization. Right? Where if you choose regularization to be simply the square to norm of theta, then you run into the Tikhonov regularization, but we have talked about regularization at length earlier. So this is another embellishment that you can make to PEM.

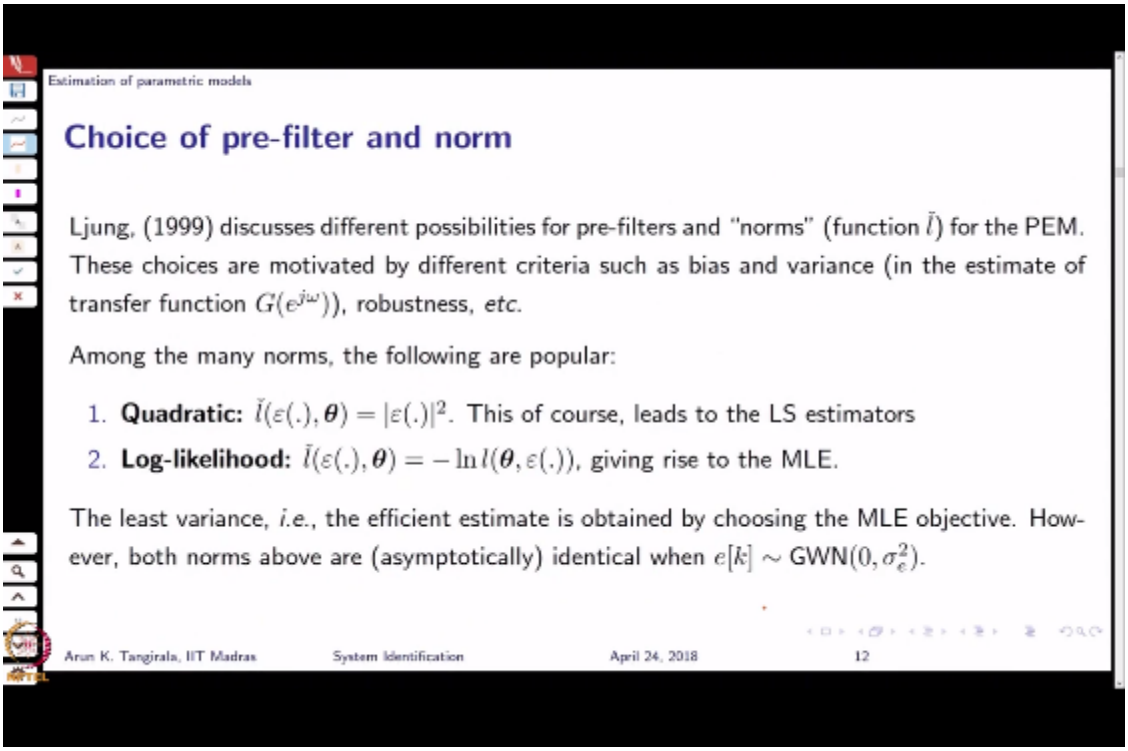
(Refer Slide Time 15:01)




Now we shall look at some special cases. So we have discussed three embellishments that we can make to PEM. Now we'll discuss some special cases that PEM simplifies to, depending on the choice of the pre-filter and the norm. So if you choose the norm to be simply the squared to norm for vector outputs, then we obtain the least-squares estimator, which we have just discussed. On the other hand, if you choose the pre-filter to be the negative logarithm of the p.d.f. or the likelihood, then you encounter the ML criteria, which is also very nice. So which means MLE is also now a special case of PEM, which is good. But we already know MLE and least squares are equal and if you assume the data to be jointly Gaussian distributed and so on. But what we have learnt just know is that MLE and least squares themselves are special cases of a broader family of methods, mainly prediction error minimization family.

And if you add, if you choose l to be this, then what estimated you get, you get a Bayesian estimator, specifically you get the maximum a posteriori estimate, the map estimate. And if you choose l inverse cap to be the negative log-likelihood and add additional dim theta by N as a regularization function, so f of theta is now going to be dim theta dimension. Dim theta is essentially the number of parameters in theta, by N. Then that's like a regularization kind of root. And when you choose to optimize this, you run into AIC estimates, Akaike information criterion. And you should remember that for a fixed model structure, the theta hat AIC is identical to MLE. Okay. Okay. So these are the four different specializations that PEM specializes to. But there are other special cases as well. We just considered the broad classes of estimators that we normally encounter to which PEM specializes.

(Refer Slide Time 17:17)




Now, as far as choice of pre-filter is concerned, so you may wonder, what is the kind of pre-filter that I should choose? That depends entirely on the application and different criteria such as bias and variance and so on. So for example, I may want to focus only on a certain frequency range, then I will, let's say some low pass frequency range, low frequency range. Then I'll filter the data or I'll choose a pre-filter whose response, whose bandwidth is, in the low frequency range. Then the bias in the, that region is also is reduced so that the model gives you a very good fit in that frequency range. But at the expense of variance, increase in variance, or you could also focus on minimizing the error variability in that frequency range in which case the bias may shoot up and so on. Or you could say that I'm interested in reducing the bias in a certain frequency range and I'm willing to sacrifice the bias on the other frequency, and that kind of compromise is also possible. So, all of this is essentially shaping the bias. It's called bias shaping in system identification.

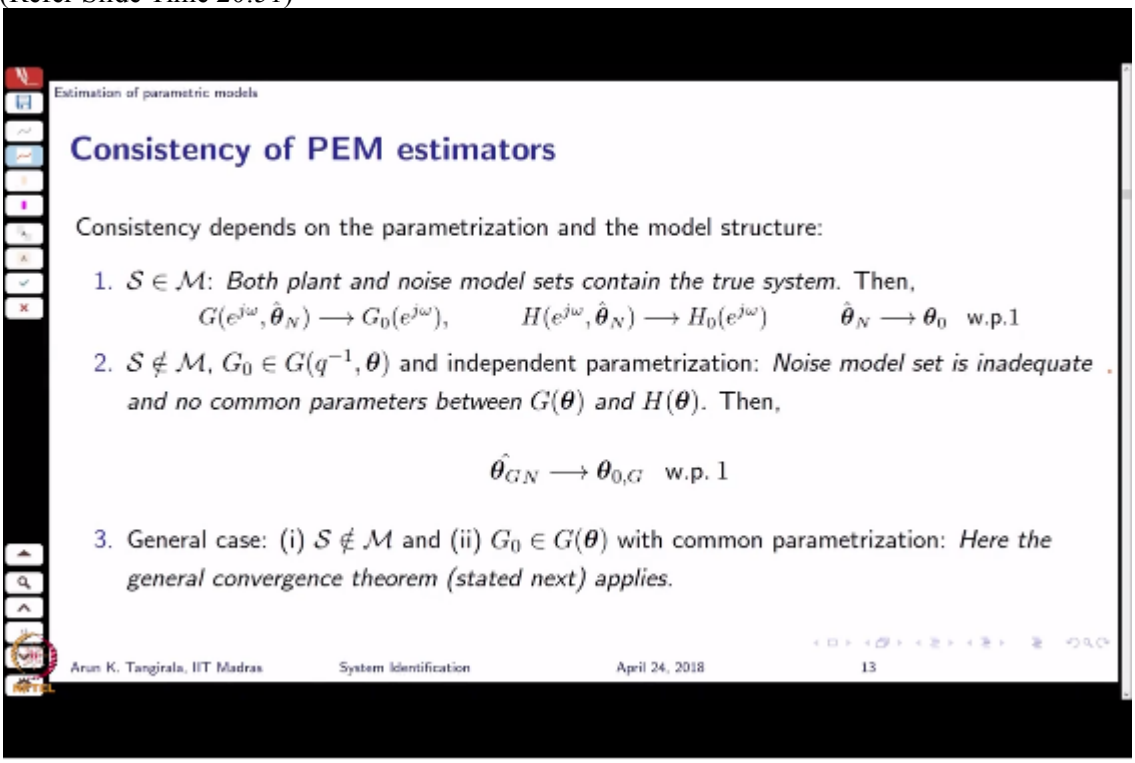
In fact, in control there is something called loop shaping. You know that estimation and control are duals of each other. Here we talked about bias shaping, they're in control, we talk of loop shaping where we minimize the sensitivity of the control loop over a certain frequency range, and by a familiar result, a very celebrated result in control in the form of body sensitivity integral, we know that if you try to minimize the sensitivity of the control loop over a certain frequency range elsewhere in the other frequency range, the sensitivity shoots up. So pretty much you would have a similar situation here. As for the choice of norms, the standard choices are quadratic and log-likelihood. These are the standard choices. And we know already one leads to the least squares, other leads to the maximum likelihood.

The best option is always to choose the log-likelihood. But if you know that the data comes from a joint Gaussian distribution, then what you can do is, you can simply choose a quadratic. And when you choose the quadratic, usually it's called a quadratic PEM. And this is what the PEM routine in MATLAB assumes. The OE method that you choose, that you use, or the BJ method, ARMAX and so on, essentially uses the quadratic PEM. In MATLAB system identification tool box, there are

specialized routines for each of the model structures. And there is a genetic routine called PEM, which can be applied to discrete time, estimating discrete time models of any structure as well as continuous time models. Okay? Because the PEM routine simply doesn't worry about -- doesn't necessarily cater to a particular model structure. Okay. So any questions?

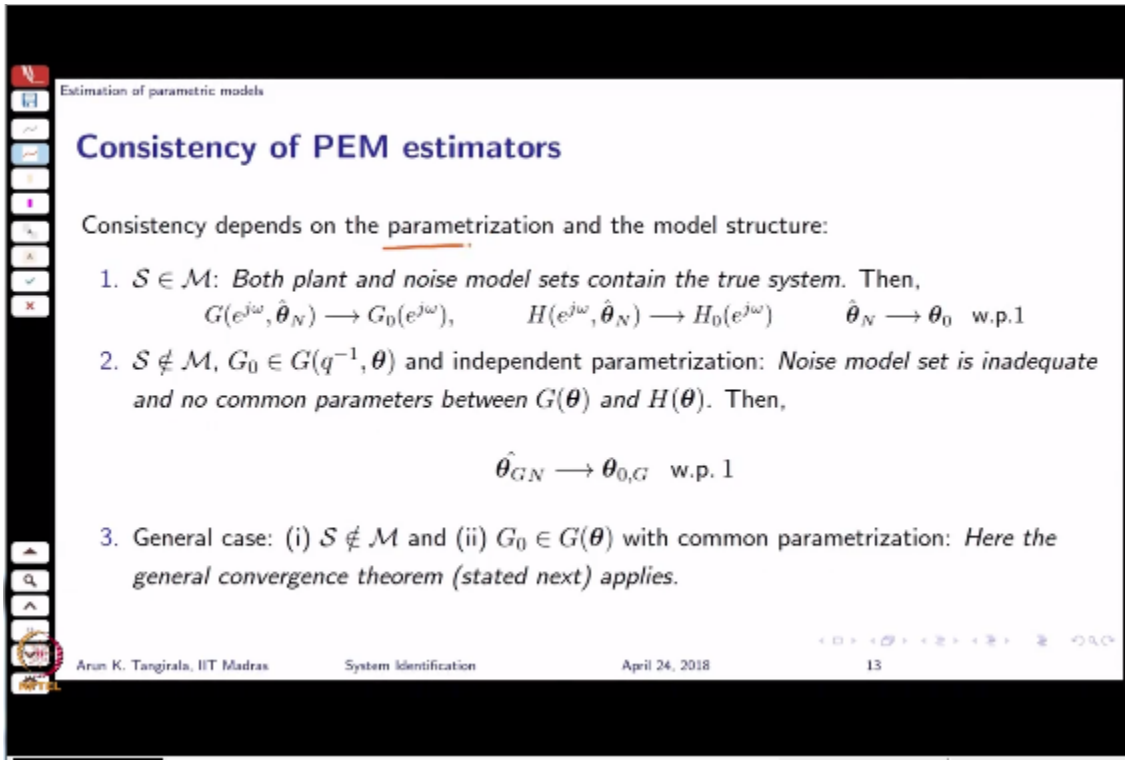
So now the most important discussion.

(Refer Slide Time 20:51)




So we have discussed the philosophy of PEM estimators, and of course, I have not talked about how you minimize objective function. Why? Because we've already discussed the nonlinear least squares, the maximum likelihood estimation and so on. Essentially you run into a nonlinear optimization problem and you simply invoke a numerical optimizer and typically a modified Gauss-Newton method is used in this nonlinear optimizers, which is much better than the standard Gauss-Newton method. And that is the case of solving it. What we are interested at this juncture is how good the estimates are, how good these PEM estimators are, what happens. Again, you can turn to nonlinear least squares and you can turn to MLE and get the answers. Fine. But what is now of interest to us is the broad PEM estimators, whether I use least squares MLE or use or whatever PEM specializes to, I'm not so worried about that. I'm worried about the consistency of general PEM estimator in the context of system identification. So if you turn to least squares or MLE and so on, they were devised earlier for parameter estimation and so on. But here, we have to answer different kind of questions, again, related to parameter estimation. But here we would like to talk in terms of models rather than parameters themselves. So that is the transition that we are making. We are moving from parameters to models. Now it turns out that the consistency of PEM estimators depends on two things: How you parameterize and what the model structure is.

(Refer Slide Time 22:40)

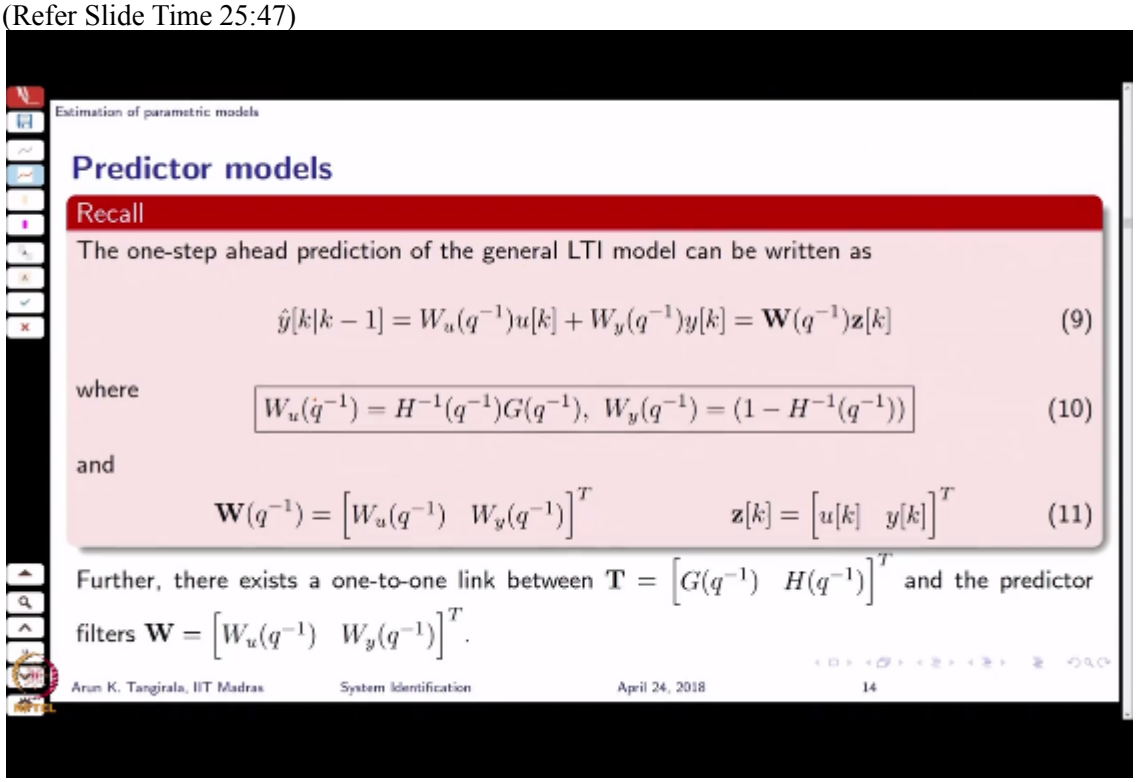



I have mentioned this earlier, at least a couple of times in some of the earlier lectures that this is what we'll get to see. And even in the liquid level example I had pointed out, as to why the output-error model structure gets you the nice estimate, the correct estimate of the liquid level system, whereas the ARX model fails to. So now we have the theoretical answers in front of us. Now when it comes to learning the results or studying these results, it is important to get into the notion of a true system, which is what is denoted by S. You can think of this true system as nothing but as the DGP. So this S is nothing but your data generating process. Or you can say it's a true system. Okay.

And then there is this notion of model that we know, but there is something called a model set which we now have to introduce, a notion of a model set. All of these are fairly intuitive but require some formalization. So if I talk of a model set, I've spoken about this earlier as well. Essentially loosely speaking, model set is a collection of models. As you traverse in the parameter space, you will be generating different models of the same structure, and a model set is a collection of models. But model set is even broader, perhaps we'll see. Then there is a third notion of whether the system belongs to the model set or does not belong to the model set. So we'll learn a notion of model set and model structure and then this notion of belongingness of the system to the model and so on.

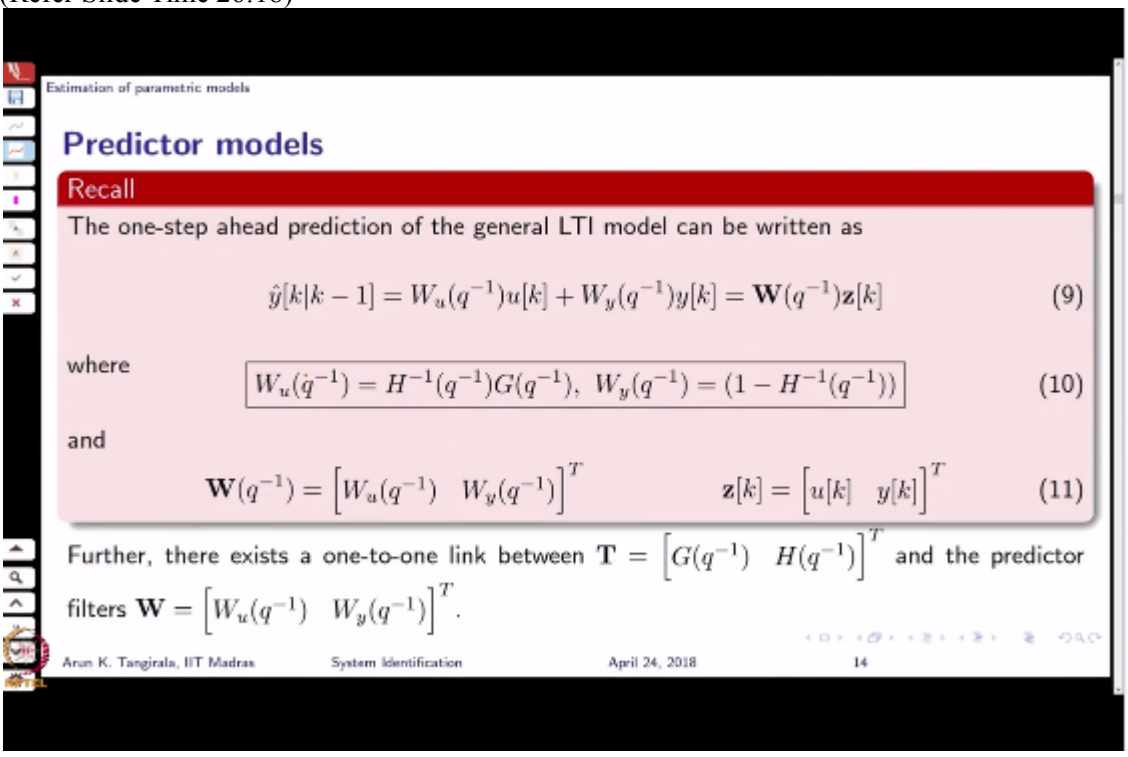
First, we'll study those and then we will be able to understand what it means to talk about these three different cases that I have listed. Depending on the case, you have the consistency result of the PEM estimator, right? So in passing, let me say that the consistency that we are talking about is not necessarily with respect to parameters. We are talking of consistency with respect to models. Whether that model will converge to the true one is what we are asking. Earlier we have asked the question whether the parameters will converge to the true ones. And to be able to understand -- and then there's this notion of parametrization which we already know. To understand these three different cases, we will now temporarily step into some formal world of definitions of model set, model structure, and what it means by S belonging to M, S not belonging to M, and all this notation. Then we'll come back to this slide.

(Refer Slide Time 25:47)




Now in order to understand what a model set is, we will start with the notion of what is known as a predictor model. All right. This predictor model is not something alien or some mysterious concept. It is an alternative way of making the prediction of your process. Or you can say, it's an alternative way of describing your process. It's also a model, but it's based on the idea of prediction straight away.

(Refer Slide Time 26:18)

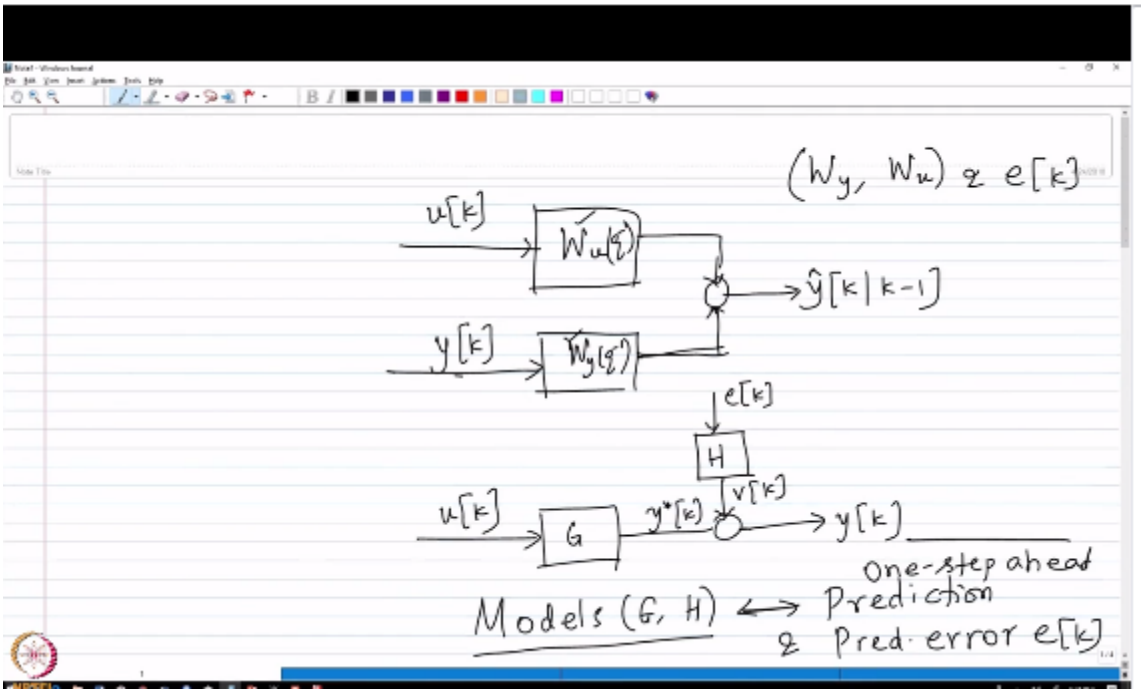


So if you recap, we had this one-step ahead prediction. We had derived this one-step ahead prediction expression long ago, right? Where Wu is given by H inverse G, if you remember this one-step ahead prediction and Wy is 1 minus H inverse. A very familiar expression to us. Now what we notice here is that, I can write the prediction, which is one of the ultimate objects of interest to me, as a filtered combination of the output and input. So it's just a different viewpoint. So what we are seeing is y hat of k given k minus 1 is essentially this. So what we are doing is we are feeding u[k] through Wu and y[k], of course, you will only need past y[k], although I write y[k] here, you will need only data up to k minus 1. And this is another filter. And then you combine the outputs of both these filters to get your y hat of k given k minus 1.

What is the difference between this and what we have been looking at? Compare this with what we have been writing. We are saying, for the model, we are saying u[k] excites G, which produces y star, the true response, and that's corrected by v[k], which we assume is generated by white noise passing through a filter H, right? A stable filter. And here comes your y[k]. So what we have learnt until now is given G and H, and of course I know e[k] is white now, so given G and H, I can construct a prediction for y hat -- for y, sorry. Correct, that's what we have learnt. And we have also proved that the one-step ahead prediction error is e[k], which means that we have learnt how to go from models G and H to prediction and prediction error. And we've said that the prediction error is e[k]. Prediction error. Given the models G and H, I can always construct the one-step ahead prediction and the associated prediction error which is nothing but the e[k]. Now it turns out that you can also go backwards. Given one-step ahead prediction and the fact that the prediction error is white, you can find a unique G and H. So there exists a unique mapping between the models, model find of description, GH description. What's this? Your prediction model. So if you are in this world, we are the standard description. If you're on this side of the fence, then you are in the predictor model world, where you are now not describing G and H, rather you are now giving Wu and Wy, and you are saying that whatever is left out is e[k].

So an alternative way of describing the system is to specify Wy, Wu, and also specifying what is e[k]. What do we mean by e[k] is that we give the p. d. f of e[k], and we say that that is a one-step ahead prediction error and so on. And one can show that there exists a unique mapping between these two. Why on earth are we talking about all of this is probably the question that might be occurring to you at this point.

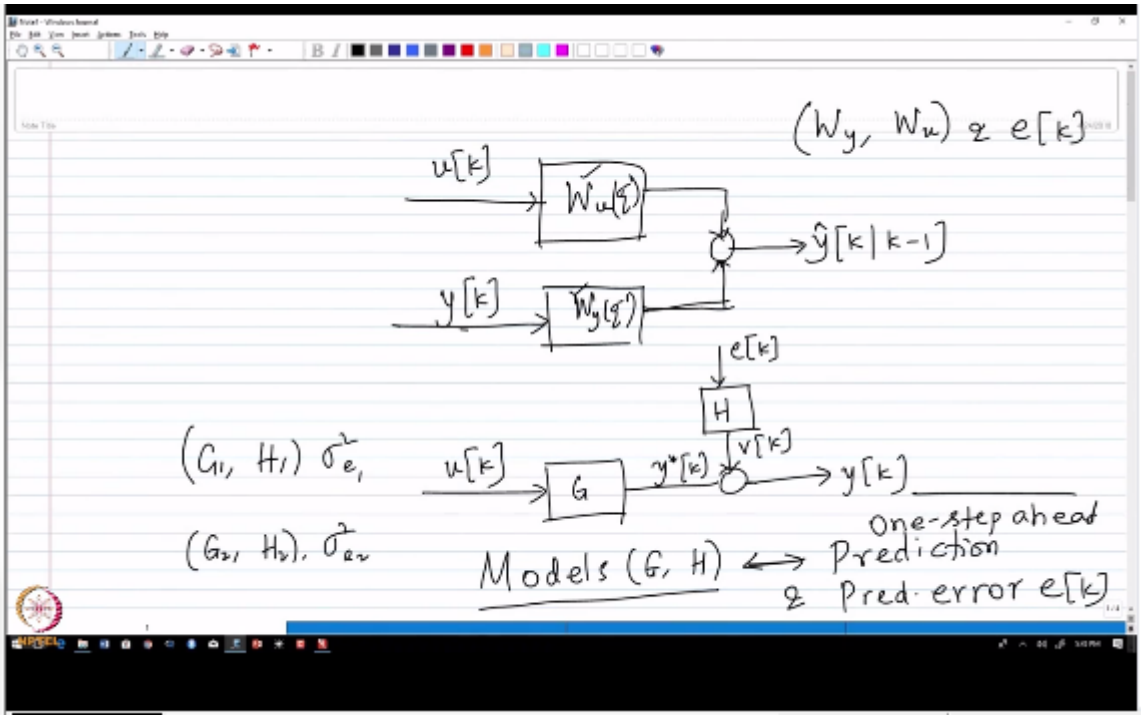
(Refer Slide Time 30:40)




Well, it turns out that when you want to define equality of models, for example, which concept requires this concept of equality of models? What is a concept in identification that may require you to talk of, think of equality of models? You have any idea? In what kind of situations and identification I may have to compare two different models and say that they're equal? Can you think? Grey box model, no. One of the key concepts in identification requires this idea of equality of models. Hmm. Identifiability requires. What is identifiability? I mean, identifiability has several branches to it. But when you talk of identifiability as affected by input design, how do we want to design our input? We want to design our input so that I will be able to distinguish between two different models. So it turns out that -- sorry. When you want to formally design the input, when you want to derive the conditions for a very good input, then you start by first talking of equality of models, and then say that I would like to design an input so that the input allows me to discriminate between two models, which means that two models, structures should be unequal at all frequencies. And that's what leads to the derivation of the persistent excitation condition, which we shall talk about a bit later.

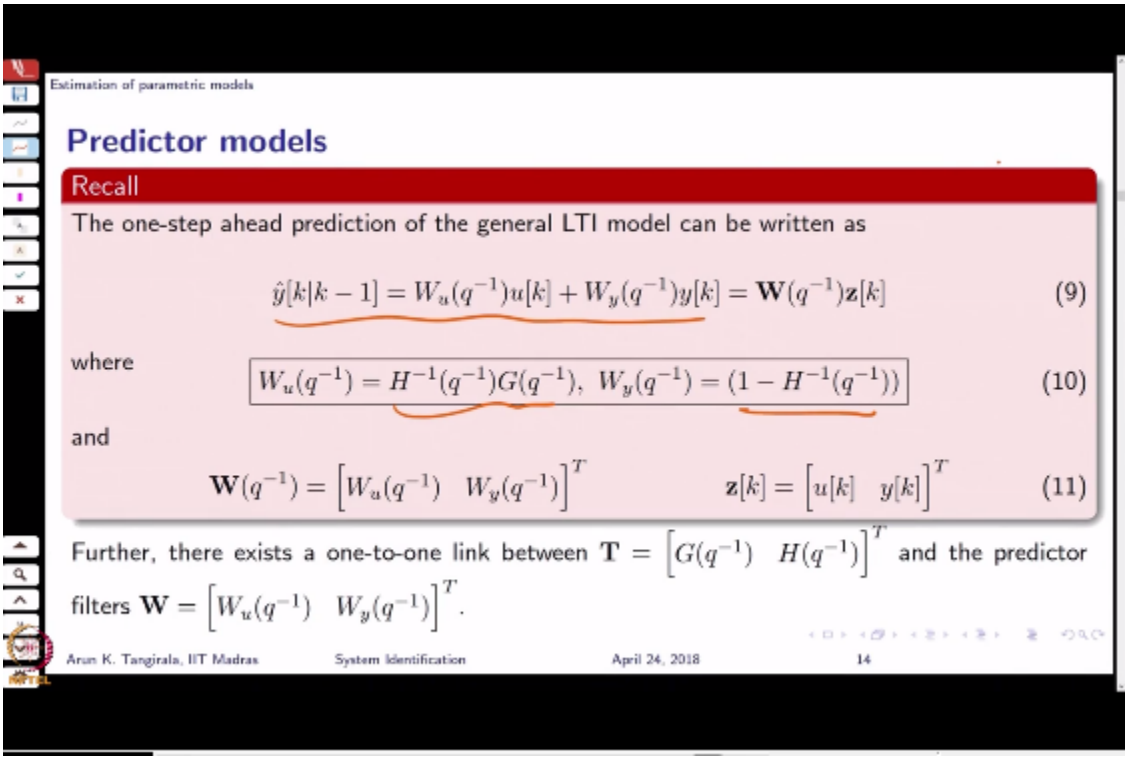
And in doing that, we will need this concept of predictor models. That is one place where you will have to talk of equality of models. So when I take two model structures, on the face of it, they may be different. So I may have a G1, H1 combination, and of course, accompanying sigma square e1, and I may have another model G2, H2 and sigma square e2. So when I compare structure wise, they may not be equal. But if they give me the same predictions, then we say they're equal. So that is one notion of the equality of models. And also in defining our model set, model structure, and then the system belonging to the model and so on, the predictor models are unifying. The reason is that, when we talk of models G and H, they could be nonparametric form or they could be in parametric form. Whereas predictions are predictions, right? So, y hat is y hat, whether you generate using a nonparametric model or a parametric model, doesn't matter.

(Refer Slide Time 34:00)




As long as two models, one being a nonparametric, other being a parametric, they generate the same prediction. We say that two models are identical in that sense. Why do we need this? Because remember, we had this case, we said, we want to say S belongs to M, for example, right? What is S? S is the true system. M is the model that I fit. Generally the system that we are dealing with may or may not have a parametric form. We do not know. It could be a nonparametric form. But we may be fitting a parametric model, we are in fact in that context of discussion. So how do you say a nonparametric model belongs to a parametric model? How can you say that? So you need to talk of this belongingness and equality in some sense. And that is where this predictor models come in. So to summarize, what we have is essentially the predictor models.

(Refer Slide Time 35:06)




We consider the predictor model as W. In fact, there is an additional requirement on W. But if you specify W, then what you are specifying is a predictor model. And as I have said, you can always derive the G and H from this W uniquely. You don't have to start from G and H. You can start from this W and then derive your G and H. Of course, you have learnt how to go there from the other way around.