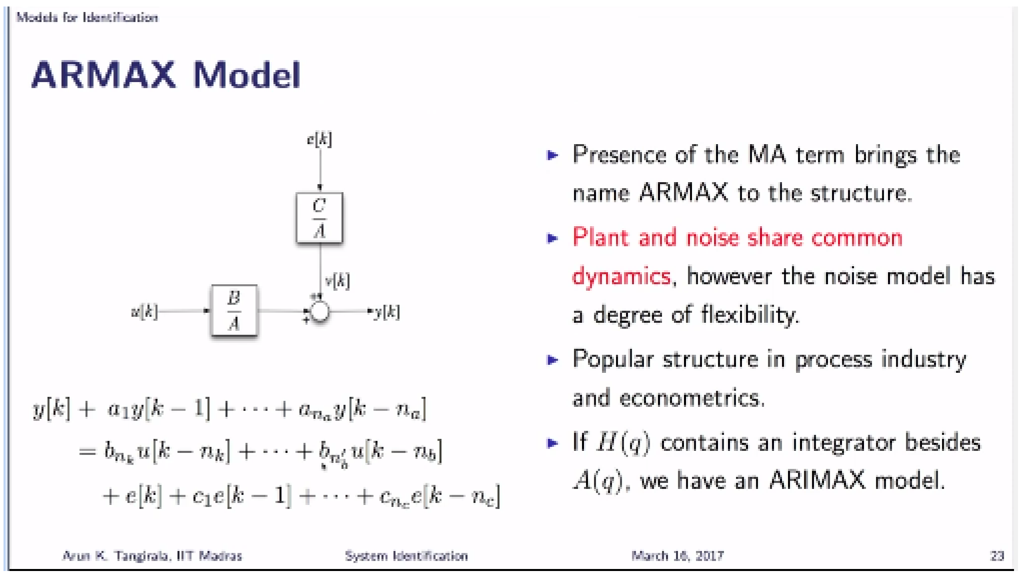
CH5230: System Identification

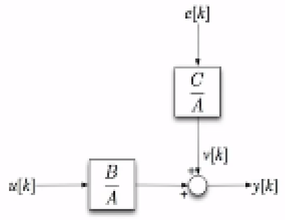
Models for Identification 4

So let's move on to the ARMAX model. We have talked about ARX model quite a bit. The ARMAX model builds upon the ARX's model and it says, yeah, you know my hands are tied, agreed, so it's pretty restrictive. Let me introduce some flexibility in the noise model. These are quite popular in the industry. What is the flexibility here? We are introducing a moving average term in the noise model and that's why it acquires a name ARMAX. That x stands there for exogenous as I've said before. There is still a joint parameterization but it has its own identity also. In the form of c, that obviously enhances the ability of this ARMAX model to explain a lot of processes. Very nice.

(Refer Slide Time: 00:58)



ARMAX Model

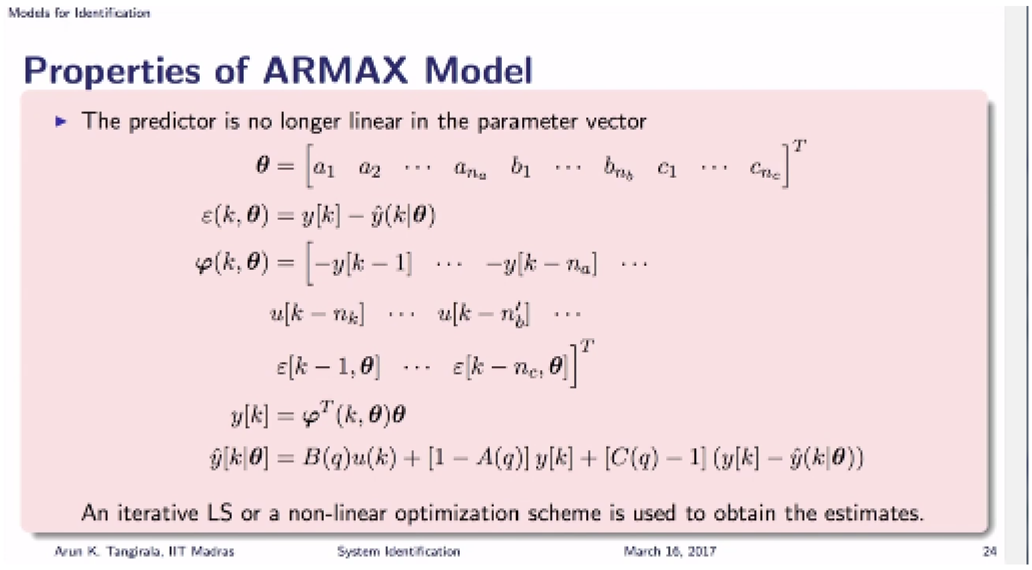


$$
\begin{aligned}
y[k] + a_1 y[k-1] + \cdots + a_{n_a} y[k-n_a] \\
= b_{n_k} u[k-n_k] + \cdots + b_{n_b'} u[k-n_b] \\
+ e[k] + c_1 e[k-1] + \cdots + c_{n_c} e[k-n_c]
\end{aligned}
$$

- Presence of the MA term brings the name ARMAX to the structure.
- Plant and noise share common dynamics, however the noise model has a degree of flexibility.
- Popular structure in process industry and econometrics.
- If $H(q)$ contains an integrator besides $A(q)$, we have an ARIMAX model.

But you should not forget that there is still a giant parameterization. There is still that assumption that dynamics are shared. So it's still a bit unrealistic. However, the inclusion of C gets it out of that restrictive mode to uncertain extent. However with the introduction of the C, if you look at the difference equation form now you should straightaway guess that we are going to have some trouble with the predictor, correct. Look at the prediction expression that you would get. Of course, I have not written the prediction expression in the difference equation form but I've written this in the regression from. That's only difference. Earlier on the board I have written it in the difference equation form, now what do you notice about the regressor? Regressor is also a function of theta. Why has that happened?

(Refer Slide Time: 01:55)


Models for Identification

Properties of ARMAX Model

- The predictor is no longer linear in the parameter vector

$$\boldsymbol{\theta} = \begin{bmatrix} a_1 & a_2 & \cdots & a_{n_a} & b_1 & \cdots & b_{n_b} & c_1 & \cdots & c_{n_c} \end{bmatrix}^T$$

$$\varepsilon(k,\boldsymbol{\theta}) = y[k] - \hat{y}(k|\boldsymbol{\theta})$$

$$\boldsymbol{\varphi}(k,\boldsymbol{\theta}) = \Big[-y[k-1] \ \cdots \ -y[k-n_a] \ \cdots \ u[k-n_k] \ \cdots \ u[k-n_b'] \ \cdots \ \varepsilon[k-1,\boldsymbol{\theta}] \ \cdots \ \varepsilon[k-n_c,\boldsymbol{\theta}] \Big]^T$$

$$y[k] = \boldsymbol{\varphi}^T(k,\boldsymbol{\theta})\boldsymbol{\theta}$$

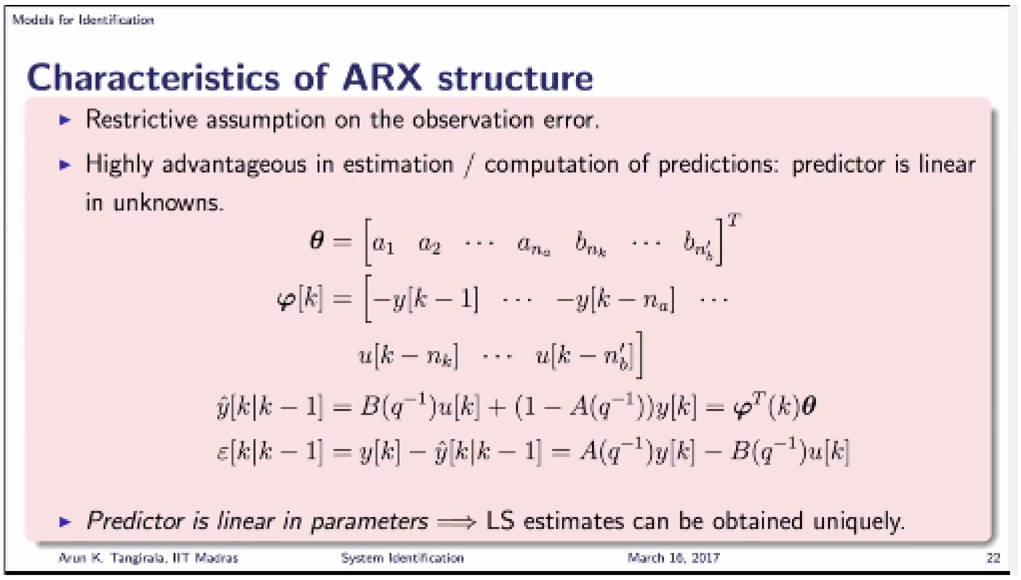
$$\hat{y}[k|\boldsymbol{\theta}] = B(q)u(k) + [1 - A(q)]\, y[k] + [C(q) - 1]\, (y[k] - \hat{y}(k|\boldsymbol{\theta}))$$

An iterative LS or a non-linear optimization scheme is used to obtain the estimates.

Arun K. Tangirala, IIT Madras System Identification March 16, 2017 24

Why has that happened? What is the difference between the regressor here and the one in the ARX? Is there a commonality between the regressor, and ARMAX and the ARX? You look at the regressor here, right. This is for ARX.

(Refer Slide Time: 02:14)


Models for Identification

Characteristics of ARX structure

- Restrictive assumption on the observation error.
- Highly advantageous in estimation / computation of predictions: predictor is linear in unknowns.

$$\boldsymbol{\theta} = \begin{bmatrix} a_1 & a_2 & \cdots & a_{n_a} & b_{n_k} & \cdots & b_{n_b'} \end{bmatrix}^T$$

$$\boldsymbol{\varphi}[k] = \Big[-y[k-1] \ \cdots \ -y[k-n_a] \ \cdots \ u[k-n_k] \ \cdots \ u[k-n_b'] \Big]$$

$$\hat{y}[k|k-1] = B(q^{-1})u[k] + (1 - A(q^{-1}))y[k] = \boldsymbol{\varphi}^T(k)\boldsymbol{\theta}$$

$$\varepsilon[k|k-1] = y[k] - \hat{y}[k|k-1] = A(q^{-1})y[k] - B(q^{-1})u[k]$$

- *Predictor is linear in parameters* $\Longrightarrow$ LS estimates can be obtained uniquely.

Arun K. Tangirala, IIT Madras System Identification March 16, 2017 22

And you look at regressor for ARMAX. What is the difference that you notice? Huh. Why did that error time creep into the regressor? Because we introduce the MA term. So you look at the difference equation form here on the right hand side, in addition to ek, you have c1 ek minus minus 1, c2 ek minus minus 2 and so on depending on the order. They also have to be known if I have to estimate a, b and c using a linear regression framework. Note what I'm saying. If I want to use linear regression framework I need to know ek minus 1, ek minus 2 and so on. But do I know that? They are fiction. I don't know. But if I give you a model, can you estimate ek minus 1? If I give you the model would you be able to estimate eks? That's a very important question you have to answer. In general even go back to the ARMAX model. Let's say, I give you this ARMAX model. Will you be able to get me an estimate of ek? I'll give you a few seconds to think.

Is it possible? Yes or no? Estimate of ek. You will not be able to get me exact ek. What is ek? What are the different interpretations of ek? One of the earliest interpretations of ek with which we introduced? What is that ek? Its inherent uncertainty, unpredictability in the measurement. So in other words, it is Y minus Y hat. That is optimal one step ahead prediction. The difference between the optimal one step ahead prediction and the measurement itself. So you look at the-- go back to this example and you see, if I subtract these two, what is left? Ek, of course, when I give you the model I'll give you not theta. I'll give you theta hat. So in addition to ek, you will also have estimation errors or some modelling-- slight estimation errors also creeping in so that you will never be able to recover ek exactly. But theoretically if I give you theta, you will be able to recover ek. Correct? If I give you theta and if I give you all the measurements, all you have to do is construct a prediction. Take the difference between y and my hat. And that will give you ek. So in other words, if I give you a model, you will be able to estimate ek for me. Estimate only in practice. Now I can use that estimate into the regressor for ARMAX. And I can complete my regressor. Otherwise, I have to beg borrow or steal ek from somewhere. Right? So one of the algorithms for estimating ARMAX models using linear regression farm is that use start with some other model. Let's say, you start with an ARX model of this order, get an estimate of ek. Because ARX's models that easy to estimate, linear regression, linear least squares, I'll estimate, I'll get an estimate of ek and then construct my regressors for the ARMAX and now I pretend, I know all my regressors. I know y, I know u, I know, I have written Epsilon in place of e because now an estimate of ek is being used which is at one step ahead prediction error. I'll estimate now the ARMAX model parameters. From the ARMAX model parameters, I will construct my prediction, improve the estimate of ek and then go through an iteration. I'll plug that back into the regressor and go through iterative, you understand?

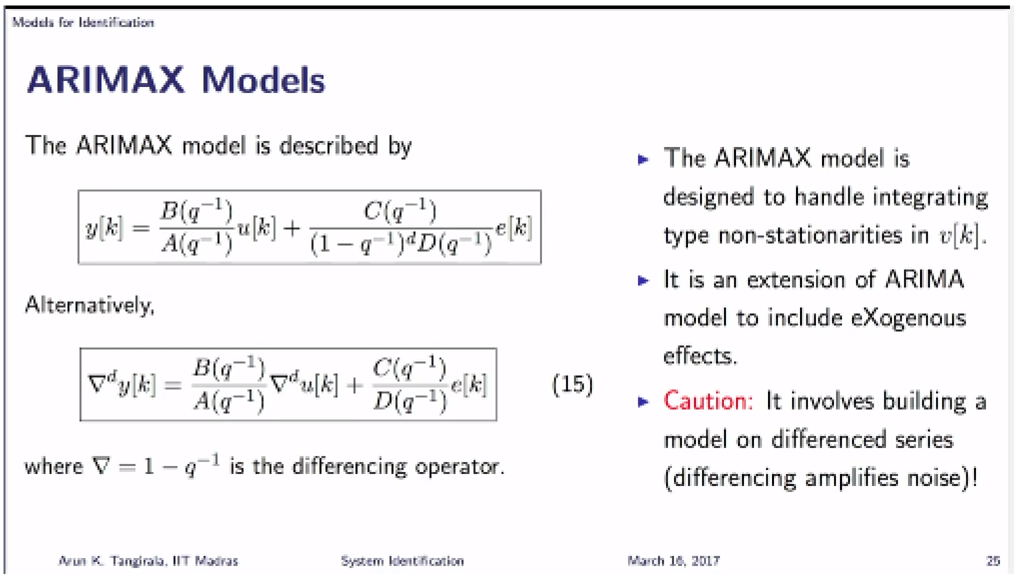
So you kick start the algorithm with an ARMAX model. This is one of the algorithms for estimating ARMAX model. You kick start the algorithm in an ARMAX model, get an estimate of ek. Completely regressor vector. Construct your regressor vector. Now it's a linear regression problem. Right, now it's purely regression problem. Estimate the ARMAX model parameters. Once your ARMAX model is estimated, you use this model to improve your estimate of ek. So it's like an iteration. With the new model I'll re-estimate my ek. Always ek by definition regardless of the model structure. What is it? It's a difference between y and y hat. So with this new ARMAX model I will construct a y hat. Take the difference between y and y hat. Update my regression vector. And then go through that cycle. This entire approach or even writing this in this form is called a pseudo linear regression. It a pseudo linear regression. It's not a pure linear regression.

If it was pure linear regression the regressor should have been independent of theta like ARMAX case. But this is pseudo in the sense that given the some estimate of theta. Need not be optimal. Then it becomes a linear regression form. Okay. But that is only one way of estimating an ARMAX model. You can estimate ARMAX models using other approaches as well. What is other approach? The nonlinear least squares approach. We'll discuss that later on. But the fact that of the message that you should take is introduction of the moving average term has resulted in a predictor that is non-linear in parameters. I'm I right? Because the dependency of the Epsilon's or the eks on theta is not linear. It's a very complicated function. Therefore, my predictor is non-linear in unknowns. The other way of looking at it is, you go back to the ARMAX model here. And say, well. Look at the right hand side. If I throw away, ek I get the predictor. Correct. I take all the terms to the right and just threw away ek I get the predictor. What are the unknowns a b and c but eks are also unknowns. And unknowns are appearing in a product from. And these unknown eks are complicated functions of a b and c.

Remember, ek is not an external signal. It is internal. Right. So it's an implicit function of your model. And that is another way of looking at it at the fact that ARMAX models result in non-linear regressors. So in general, there is some difficulty as compared to ARMAX model in estimating the ARMAX model parameter. It's not impossible, I'm just saying, it's a bit difficult one that you will not get a unique solution. Guarantee. There's no way you'll get a unique solution because after all what you are solving is the nonlinear least square problem.

As a result you don't have a unique solution. You don't have a closed form expression, you have to be content with a numerical optimum and so on. As compared to ARMAX models. But what is a benefit that you have gained by working it ARMAX models? You have no managed to explain a larger class of processes. Okay, but still it has the DNA of an ARMAX model which is that the joint parameterization exists and that again has implications in your ability to recover the true model as we shall learn later on. Now our EMAX models are essentially an extension of the ARMAX model to handle integrating type non-stationarity. There's not much to talk about this. We are talked about ARMA and then we have talked about ARIMA, correct. What is the difference? In ARIMA, You fixe one of the poles or some OF poles to the unit circle because you believe that the stochastic process as integrating effects. Same story here. The noise model now is an ARIMA model. Instead of an ARMA that's all.

(Refer Slide Time: 10:20)


Models for Identification

ARIMAX Models

The ARIMAX model is described by

$$\boxed{y[k] = \frac{B(q^{-1})}{A(q^{-1})}u[k] + \frac{C(q^{-1})}{(1-q^{-1})^d D(q^{-1})}e[k]}$$

Alternatively,

$$\boxed{\nabla^d y[k] = \frac{B(q^{-1})}{A(q^{-1})}\nabla^d u[k] + \frac{C(q^{-1})}{D(q^{-1})}e[k]} \qquad (15)$$

where $\nabla = 1 - q^{-1}$ is the differencing operator.

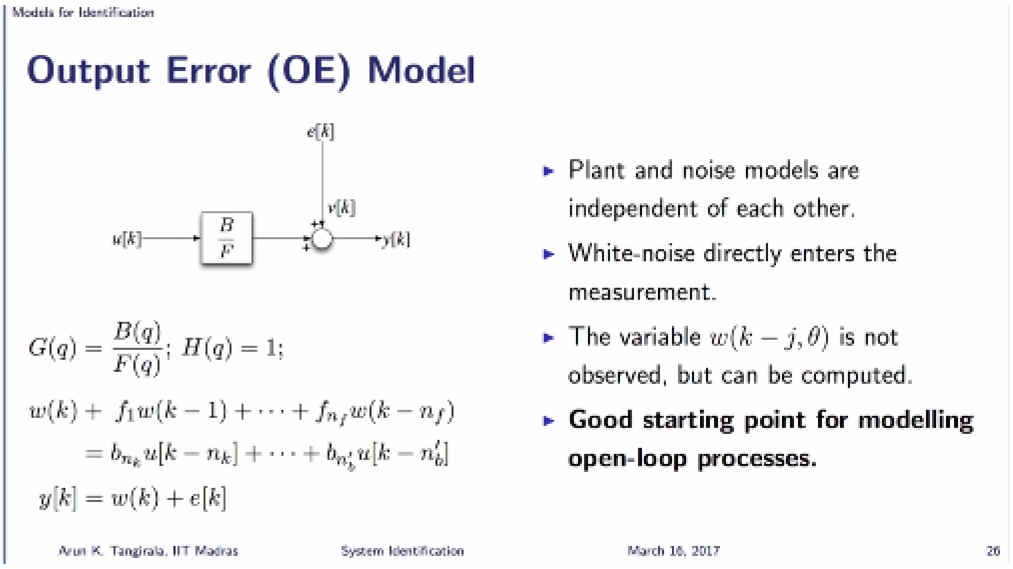
- The ARIMAX model is designed to handle integrating type non-stationarities in $v[k]$.
- It is an extension of ARIMA model to include eXogenous effects.
- Caution: It involves building a model on differenced series (differencing amplifies noise)!

Arun K. Tangirala, IIT Madras System Identification March 16, 2017 25

So I'm just giving you as an extension, now how do I know, if I have to build an ARIMAX model as versus an ARMAX. Again you have to go through some checks, you have to look at the ac f of y, you will get to see and I'll go through a case study later on, on building ARIMAX model at that point I will highlight the steps that you take to determine if an ARIMAX model is warranted. Okay. But the difference between ARMAX and ARIMAX is just that you are now including integration integrating effects into the noise model.

So let's move on and talk about output and more quickly. This is one of the most beautiful class of models widely studied. Because of its simplicity and because of its ability to recover the true model despite making a very restrictive assumption on the noise.

(Refer Slide Time: 10:20)


e[k]
v[k]
u[k]
B/F
y[k]

Models for Identification

Output Error (OE) Model

$$G(q) = \frac{B(q)}{F(q)}; \; H(q) = 1;$$

$$w(k) + f_1 w(k-1) + \cdots + f_{n_f} w(k - n_f) = b_{n_k} u[k - n_k] + \cdots + b_{n_b'} u[k - n_b']$$

$$y[k] = w(k) + e[k]$$

- Plant and noise models are independent of each other.
- White-noise directly enters the measurement.
- The variable $w(k-j,\theta)$ is not observed, but can be computed.
- **Good starting point for modelling open-loop processes.**

Arun K. Tangirala, IIT Madras System Identification March 16, 2017 26

So it assumes h is 1. Need not be the truth. However it assumes that, it assumes that the white noise directly enters the measurement. Without being filtered. Is that going to be true? In fact this is a lot more realistic than ARMAX model, right. See, suppose, I have a process, let's say, a measuring liquid level. I can assume that the measurement noise let's say, let's ignore all the other disturbances liquid level is too simple. So therefore, I can say, no other disturbances exist. The only source of v is measurement noise. In which case it's fair enough to assume that the sensor noise is white. Need not be but it's fair, it's realistic. How do you check by the way? If an OE model, let's say, I give you, I asked you to do this experimentally. How will you check if an output model is warranted? Later on I'll show you how to learn this from data without doing the experiment. But suppose, you were to perform an experiment, simple experiment to figure out if an OE model is suited. What would you do? No, the same instrument, same sensor. No. Very simple experiment. What are you out to check, if ek is white? How will you do that? No. There is no y hat, nothing, simple experiment. Just turn off the input. Good. Turn off the input and now observe the measurements. And look at the ACF of the measurement. That's all. Very simple. You didn't think of this simple thing. You thought it must be pretty complicated. No.

See, vk you have to understand is all the things that the input doesn't contribute. So you turn off the input you'll get a feel of v. This is okay, if you have experimental access. Access to the experimental setup but many times you are working with routine operating data. The industry has running an operation, you can go and fiddle around with the process. Then you have to learn to figure out if the output at a model is suited to the data. That's where you go through statistical checks and so on. But it's also possible to check even without fitting a model. We shall learn later on when we learn to estimate non parametric models as to how you can figure out from the data whether an output error models is suited or not. Now the beauty of this output error model is that as I said, it will allow you to recover the true G provided you choose a high I mean, sufficiently high order. So in other words, if the system is second order, the true G second order and you have chosen a second order or third order and higher orders then you are guaranteed theoretically that you will recover the truth. Whereas that is not the case with ARX as we have seen in the liquid level case study. In the liquid level case study although I chose the same order for G as a true process, it didn't give me. It said, you have to choose the fifth order. Correct. That's because of the joint parameterization. With output error struck. There of course, I added white noise but I have also asked you to do an assignment that colored noise and see, if you are still able to recover. And hopefully you have figured that out that even when the noise was colored, the output error model structure allowed you. It is a feature of the model structure. It has got nothing to do necessarily with the estimation. It has to do with estimation algorithm but by and large it's a feature of the output error model structure itself. That's why output error models are preferred. But what can you say about the predictor? Is it going to be linear or non-linear and parameters?

Why? Simple argument. Very simple argument. Okay. Anyway it doesn’t matter. How did you come to the conclusion it's non-linear? I gave you the difference equation form here. By the way that w that you see there is simply y star. I've written this as w but it's actually y star. I have not given you the expression in terms of y but I have said, is w plus e, correct. How is it non-linear? One way to imagine output error model is, it's a simple. It's a special case of ARMAX model with C being equal to a. Right. You can think of the output error model as a special case of ARMAX model with C being equal to a. And then you will figure out that yes, it has an nonlinear form but we will work that out a bit later on.

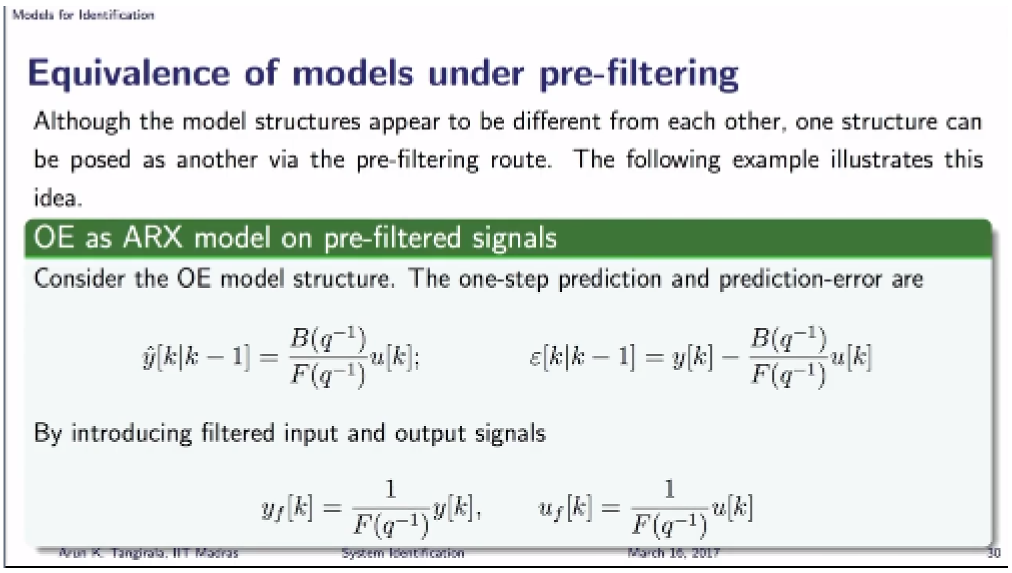
But the fact is yes, even output error models result in up except ARX, all the other model structures will result in predictors that that non-linear in parameters and you to keep on repeatedly asking yourself what has this non-linear and parameters and linear and parameters got to do? And as I said, it has got to do with your estimation. Okay. And as I've said here earlier that the output error model has the advantage that you will be able to recover the truth despite the fact that you assume h equals 1. And the truth maybe h is not equal to 1. That's a beauty but this is true only under open loop conditions.

Okay so let's finally talk about the Box-Jenkins model. The Box-Jenkins model as you know is the super dada of all model structures. And it was proposed by Box and Jenkins in 70s and that's why the name given to this model structures. The early models that prevailed were equation error models ARX and output error for a long time. It's only then when people came along and said, now it may be very obvious, oh, you should have actually thought about this early on. But every invention, every discovery seems to be like that every invention seems to be like that, oh, that's pretty obvious.

But again here the story is you should expect naturally the predictor to be non-linear in parameters and but the big advantage of Box-Jenkins models is among all the model structures it can cater to the largest class of LTI process. In fact, if you choose suitable orders for B C D and F it can model any LTI process whose responses have a parametric form. So it's really versatile and therefore you should ask the question, why don't you begin with the Box-Jenkins model structure? What do you think? Why don't they begin with the Box-Jenkins if it actually is able to in any modelling exercise. In the liquid case study I didn't discuss the bj model structure at all. What is the catch there? So I get so many orders, I have to go through so many trial and all, trial and error approaches. Instead, I begin within an OE or an ARX. Where I need to guess fewer number of orders. Get my G right, for example, if an output error model I'm guaranteed that I'll get the orders of G currently then you look at the residual from the output error model and see, if it needs modelling. That means, whether h equals 1 is correct or not? If it isn't then you build a time series model to h, you get an idea of the times h model itself. Now you come back and say, I'm going to fit a Box-Jenkins model. I have good guesses of the orders of B C D and F, I have good initial guesses and I'm going to feed that and then estimate the BJ model. This is the standard approach that you should get into. This is what I strongly recommend and you should try and if you do this you will not go wrong. At least, if the data generating processes is LTI. You will not go wrong. Okay, so that is something to remember.

Now very quickly I'll just take about two or maybe two to three minutes and we'll be done. We have discussed different model structures. What was the difference between all of this. The choice of plant and noise models?

(Refer Slide Time: 20:14)



Models for Identification

Equivalence of models under pre-filtering

Although the model structures appear to be different from each other, one structure can be posed as another via the pre-filtering route. The following example illustrates this idea.

OE as ARX model on pre-filtered signals

Consider the OE model structure. The one-step prediction and prediction-error are

$$\hat{y}[k|k-1] = \frac{B(q^{-1})}{F(q^{-1})}u[k]; \qquad \varepsilon[k|k-1] = y[k] - \frac{B(q^{-1})}{F(q^{-1})}u[k]$$

By introducing filtered input and output signals

$$y_f[k] = \frac{1}{F(q^{-1})}y[k], \qquad u_f[k] = \frac{1}{F(q^{-1})}u[k]$$

Arun K. Tangirala, IIT Madras System Identification March 16, 2017 30

Okay. Particularly the way I parameterize them and also the difference is like if you take ARX and ARMAX, what is the difference? The difference is for example, the moving average term. Now there is a beautiful perspective in all of this which is that if I change the noise model, so look at the ARMAX model, I have yk equals B of q over A of q uk plus C of q over A of q ek. This is my ARMAX model structure, right. Now suppose, I filter y and u with 1 over C, of course, I'm assuming it to be a SESO system. So I write it this way, what do we notice? I'm going to write this 1 over C uk I can do this only for SESO systems. Now what do you notice for the new model that I have written?

What model structure it is? ARMAX on what? On filtered data. So this is what we call as pre filtering. So what is an ARMAX model of that perspective now. ARMAX model is nothing but an ARX model on pre-filter data with the pre-filter being 1 over C. So in other words, changing the noise model amounts to pre filtering the data. So if somebody says, I've pre filtered the data, you can very well argue that oh, you are change the noise model. That's a beauty and why is it so beautiful because of two reasons one, it helps me in developing different algorithms. So for example, with the output error model there is a very nice algorithm called the Steigitz-McBride Algorithm that came about in mid

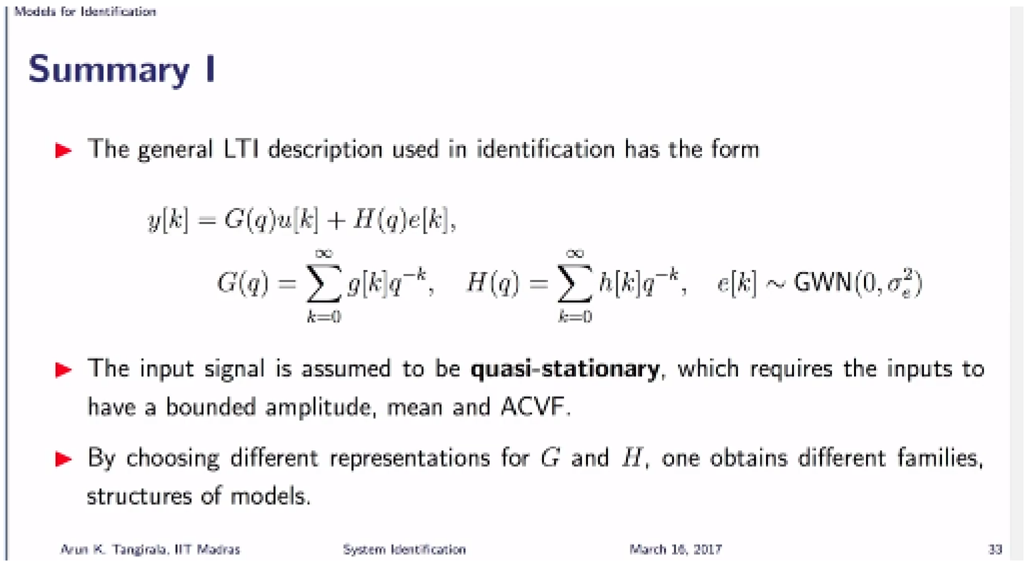
60s. Based on this equivalence between pre filtering and noise model. Okay. So if you look at the output error model you can also show that output error model is nothing but an ARX model on pre filtered data. So you look at this, this is the output error model. Now y equals B or F. Suppose, I introduce this filtered outputs 1 over F y and 1 over F u then I can write the although I don't show the right equation here, when we talk of predictions I'm going to come back to this and show you that the prediction error that you have for the OE model is the same as a prediction error for the ARX's model but on pre filter data.

What is the advantage of this equivalence? The equivalence here that we have shown is OE model is also an ARX model but on pre filtered data. What is a big deal about this observation? Well, ARX's models allow me to use linear least squares. So all I have to do is pre filter the data starts with a guess of the pre filter that pre filter happens to be 1 over f itself and then estimate the parameters, update my pre filter and get into an iterative algorithm that is the idea of this Steigitz-McBride Algorithm. Because then I can use a liner least squares at every iteration. Of course, it may not be the best way to do it but it's a very good way of estimating an output error model. And this equivalence also will throw light later on when we talk of how you know fitting this models makes a difference to the estimation FRF's. We will show and will realize that fitting a particular model structure amounts to again shaping the model bias in a certain frequency range.

So if I switch over from OE to ARX for example or ARX to OE, what I am doing behind the scenes is, I'm saying, I'm telling that algorithm, please get me a good model in a certain frequency range because I'm pre filtering. That is the consequence of this observation, that changing the noise model amounts to pre filtering means that when I change the noise model I'm focusing on different frequency range as I move from one model structure to another part of structure. That's a beautiful implication that you should remember. And we will again go through an example later on. So tomorrow when we come back, we'll exclusively talk about predictions and just get briefly started on estimation.

Okay? So this should give you-- I have talked about the equivalence of use. So there's a summary.

(Refer Slide Time: 25:10)



Models for Identification

Summary I

- The general LTI description used in identification has the form

$$y[k] = G(q)u[k] + H(q)e[k],$$
$$G(q) = \sum_{k=0}^{\infty} g[k]q^{-k}, \quad H(q) = \sum_{k=0}^{\infty} h[k]q^{-k}, \quad e[k] \sim \text{GWN}(0, \sigma_e^2)$$

- The input signal is assumed to be **quasi-stationary**, which requires the inputs to have a bounded amplitude, mean and ACVF.
- By choosing different representations for G and H, one obtains different families, structures of models.

Arun K. Tangirala, IIT Madras System Identification March 16, 2017 33

This lecture essentially has given you an idea of the model structures. Briefly talked about what are the implications of choosing model structures, features of each model structure and you should go through this again. Read the associated chapter in the textbook chapter 17 actually is devoted to a discussion on whatever we have discussed today. So tomorrow we'll meet and talk about predictions.