Estimation of Signals and Systems

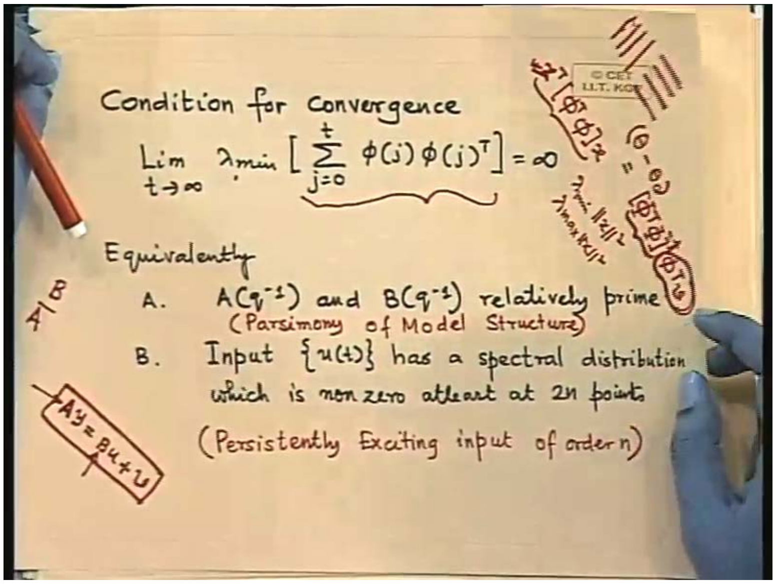
Prof. S. Mukhopadhyay
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

Lecture - 26
Least Square Estimation

So today, we are going to look at convergence conditions. We have seen convergence condition before but we did not see the full condition, so we will today see that. Then we will see, you should you know you know I mean, what happens in that; very often you when you are when you are trying to identify the system, even at that time the system might be operating close loop. So one thing you have found that, we have noted that if the system is operating in close loop then what tends to happen is that, the data vector phi get correlated with the noise because the noise comes through y; if you are using feedback to determine u from y then the same noise gets into the data vector, gets correlated and you might get some bias in the parameter estimate.

And we have also seen that, you get an at least one step that you can take to get around; that is to model the system in in greater detail using you know, like extendedly squares and things like that. But here we will see a new phenomenon when wish which might happen; if you are doing identification in a close look, so that will see and some structurally something changes, if you if you have data enclosed. And finally before leaving least squares we will look at some practical variants of least square, something you sometimes you know we have to do some tricks here, to do to ensure to take care of certain conditions. They are very simple modifications, which are intuitively clear as to why they have done, so we will we will look at them.

(Refer Slide Time: 03:31)



You will perhaps recall, that we had noted that one of the conditions; you know we we we found that the that the parameter error theta minus theta naught was given as something like phi transpose, phi inverse, phi transpose v where v is the noise, such a condition came. So we so this inverse, so we argued that; first of all this has to be small, so for ensuring that this is small v, propose several things like you know extendedly squares, instrumental variables which will make these things small. And the second condition is that, this even if one thing is that ensuring this is small but this cannot always ensure to be to be zero.

So the other thing to ensure is that, this will continuously grow; as m increases let the phi transpose, phi inverse these are capital phi's, matrix will continuously grow insides. Now continuously, now what is this? So continuously grow means, if size; means it is it will gradually this phi transpose, phi matrix will will continuously grow, if it continuously grows then its inverse will continuously grow smaller.

Now what is meant by phi transpose, phi continuously growing? We mean that, if you take any vector; it is a same thing, this is supposed to be a positive semi definite matrix, being phi transpose phi. So we are so that if you take any vector and compute phi transpose, phi x then this will continuously grow, this value for any x will continuously grow. That is what for me, this will be a scalar.

So what is phi transpose, phi? phi transpose, phi is nothing but this sum; here I used small phi's that we have seen in class, that basically I mean phi the capital phi actually consists of small phi's as row, each row is a is a small phi, actually each small phi transpose. So when you take phi transpose phi then these rows will get you will get this some because this is your capital phi transpose and this is your phi, each one of them are small phi transposes; so you have phi one phi one transpose, phi two phi two transpose, phi three phi three transpose and so on.

So what you are requiring, now if this it is it is well known that; for any vector x this this value this is the number for any vector x and if you are have any matrix x transpose, any matrix a then x transpose a x always lies between lambda mean norm of x square and lambda max norm of x square. This value will always lie between lambda mean and lambda max times of the of this x, transpose x.

In other words, if you multiply by a matrix I mean a vector by matrix you are what you are doing? You are turning it and you are stretching it. So the length of the resulting vector will always be between lambda mean and lambda max times, this that is a result. So if if lambda mean their minimum, i m value of this become infinite then obviously this matrix for all x, this will becoming infinite that that is the idea.

So the minimum i m value should go to infinite, this is a this is a statement. Now this is okay, this we have already studied. But now the question is that, how do I ensure? I have to I have to ensure this. So how do I ensure this? I have to ensure this by two things, one is that I have to ensure this by what does phi contain; phi is a vector, so it contains some input, output values, outputs are again generated by inputs.

So I have to first ensure this by choosing my inputs because that is an that is an actionable item and and I can actually apply the input; if I want if I want this go to infinity then what input should I apply? That is the question that is the very concrete question which you are which you should know. And the second question is that the the way these elements are arranged that is y k minus one u k, whether there is u k minus one, whether there is u k minus two; that is the structure of this vector, that is also important.

So the structure of the vector is important and the values are important. So the structure of the vector is actually selected by what? By the model structure; what A and what B you are chosen from their comes the comes the structure of phi, right. So so this condition will finally translate to a condition on the model structure you are assumed and a and a condition of the input, if you ensure these two things then phi transpose phi will grow unbounded. So so how do you ensure that, I am stating this without proof; there is a proof which is reasonably complicated even in the deterministic case, in the in stochastic case is extremely complicated. So we will not go into the proof, but we will remember the result because the results are very important thing for identification, very well-known results.

So the first result says, that if you have chosen; let us take a let us take a model structure that is y A y is equal to B u plus v, this model structure we are talking about A r x. So the first condition is that, this A and B should relatively prime what was it mean? What is meant by relatively prime, we will see. The other thing is that is you must choose such a model structure; such that, there should not be any any common factor, if you if you take up if you take up system y and u which fix in these kind of a description, that is the data is generated in this manner where A is a polynomial, B is the polynomial. That is the data itself fix this description and you have taken this model structure itself; you have taken this A and this B to estimate your parameters then then this A and this B should be relatively prime, that is they should not have any common factors, any polynomial can be broken up into factors.

So A and B cannot cannot have common factors, why? We will see, it is very intuitively clear reason. This condition is called parsimony of model structure, Actually the word parsimony means something like simplicity; which means that you know if A and B becoming relatively prime means what? Means that the transfer function B by A cannot be reduced further; if they have common factor then you can cancel it. So if if they have common factor then this B by A can be further simplified by cancelling that factor, but if they do not have common factor then it means that B by A cannot be simplified further.

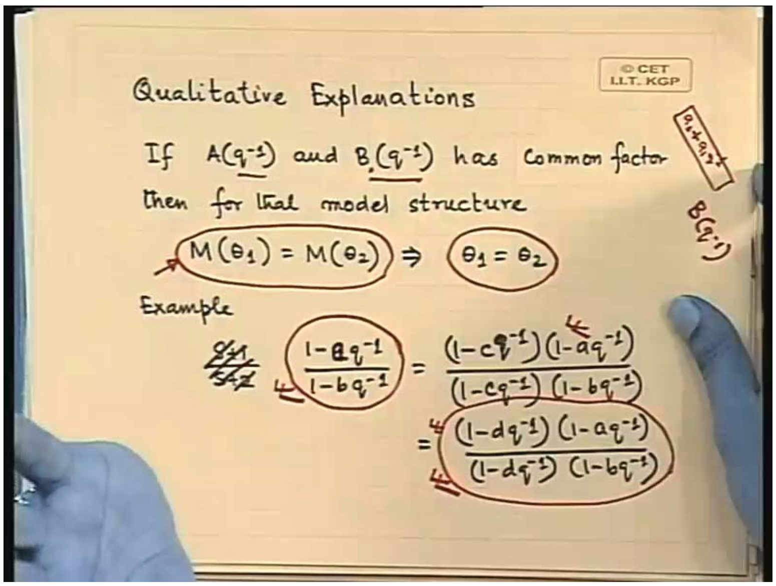
So it is in the it is in their most simplified form, that is why it is called a parsimonious structure, right. Parsimony means simplicity. The second condition is that, this condition also I mean actually it can be stated in time domain also but I have deliberately stated the the equivalent frequency domain condition, because that makes more meaning; the time domain condition is basically a sum of again, I mean something like this only, something like this only so it will not make you will not know how to ensure that condition. So so the frequency domain condition says; that the input has a spectral distribution, if you take it spectrum it should be non-zero in at least at two n number of points, what does it mean?

If in its two sided spectrum, for example; if you take a sine wave then a sine wave has how many non-zero points? Two, if you take a if you take a D C, it has how many non-zero point? Only one. If you take two sine waves, it will be non-zero at four points two sided plus minus omega. So which so basically this means, that it should at least have n sine waves; if the order of A is n, if it is an n dimension of nth order polynomial then the input that you are exciting the system with must have at least n sinusoids of sufficient power. So it should be this this is rather theoretical some more theoretical looking condition I mean condition; although if it has a practice of y, this will this also we will see. And if in such a input is called persistently exciting input of order n, right.

So the idea is that, if you are trying to know a system you have to excite it. If you cannot excite it properly, if you if you do not excite all its moulds then you cannot identify it. So you should be able to excite all the moulds together, in your input, output data all the moulds should be there. So to be able to excite all the moulds of a n dimensional system because this order is n, you need n science. So if you are so whatever input you give, that must be composeble as a sum of n sinusoids of nontrivial power; only then if you ensure these two condition, there is also a condition of stability or bounded-ness of data, that we have not considering because I mean practical situation all data are bounded, that is that is somewhat theoretical condition.

Now now let us see little bit as to why these conditions arise, okay.

(Refer Slide Time: 14:59)



The first thing is what I have said that, if A q inverse and B q inverse has common factors; actually what what are we doing by by by specifying, what what is the identification? Identification is says that, it first specifies a structure of A and B, so it said that it is equal to a naught plus a 1 2 inverse etcetera. And similarly it specifies a structure of B q inverse and in that structure it now looks for parameters. So this, what is the model? See the model is the same system can be parameterised in various structures, right. For example, any first order system which is 1 minus A q inverse by 1 minus B q inverse, this is a first order system, we say first order it can also be written like this.

So it can always be seen in the second order system. See these two if you really break it up then the forms will look different, but actually for for arbitrary value of A B C D; this system is the same as this system, right. So a model is the system itself in whatever way it is parameterised. So actually these two systems are same but the parameterisations are different. Now if we want to find the parameter, then we should take a parameterisation which is unique; which means that if two systems are equal, it will imply that they have the same parameters, this conditions will be satisfied, otherwise data is generated. Now if we if we have several kinds of parameterisation, all of which will make the same system then then from that data what which parameter will be the solution, there will be there will be I mean no unique solutions to the identification problem.

So first of all we have to ensure that, there is a unique solution to the identification problem; in other words if you have generated the data, data is generated during model, the data if you feed u and y to this and if you feed u and y to this, you will get the same u and y. So if data is generated from the same model from one model, it will imply that in that model structure, there should not exist any other parameter which will give rise to that data. That that parameter should be unique but if A and B have have a common factor then that parameterisation is not unique because this is the parameter estimation, this is the parameter estimation, what I have done what I have done, here have multiplied by 1 minus c q inverse, here are multiplied by 1 minus d q inverse.

So the so in this model structure this parameter and this parameter value are going to be different, if you really multiply but they are the same system. So if you feed the data, if you are given a pair of input output data; there is no way of determining which one of this parameter is the right one because both are light ones, so the parameters are actually is is not unique.

[Conversation between Student and Professor – Not audible ((00:18:11 min))]

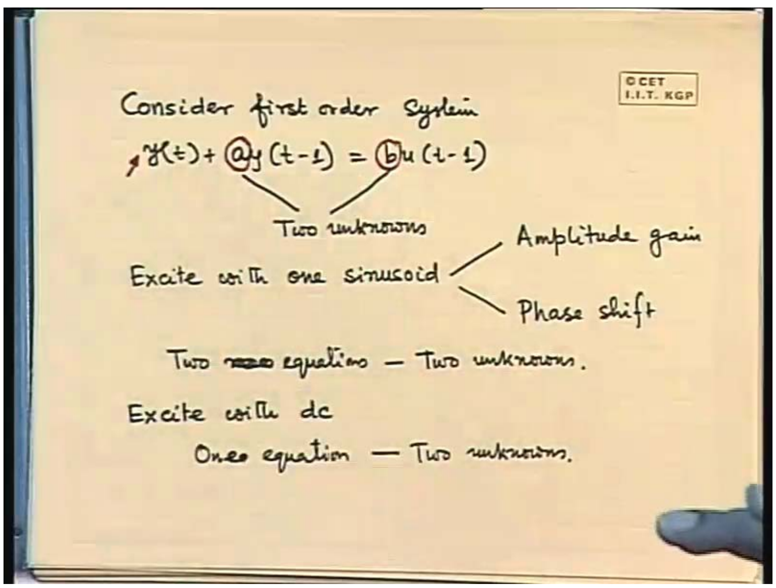
Student<Sir when we get a solution we'll get the value of v and then we come to know that which it is a common factor>

[Conversation between Student and Professor – Not audible ((00:18:32 min))]

Right, but but but the thing is you see; but but how do you know that, let you will even come to a come to a unique solution? We are we are trying to find the condition such that there will be a unique solution but in this case, there there does not exist any unique solution; which means that there may be input, output pairs for which different different solution, right. So that is a without really getting into the proof, we are trying to understand that the the rational behind. So first of all this is needed to make the problem no unique and and we will we will see that this thing comes again and again even in the case of identification, close loop, again we will encounter the situation where this parameterisation will become no unique.

So first of all the parameterisation should be unique, such that there exists only one solution to that to that I mean; which feeds the input output data under that structure of the model, that is the first thing.

(Refer Slide Time: 19:39)

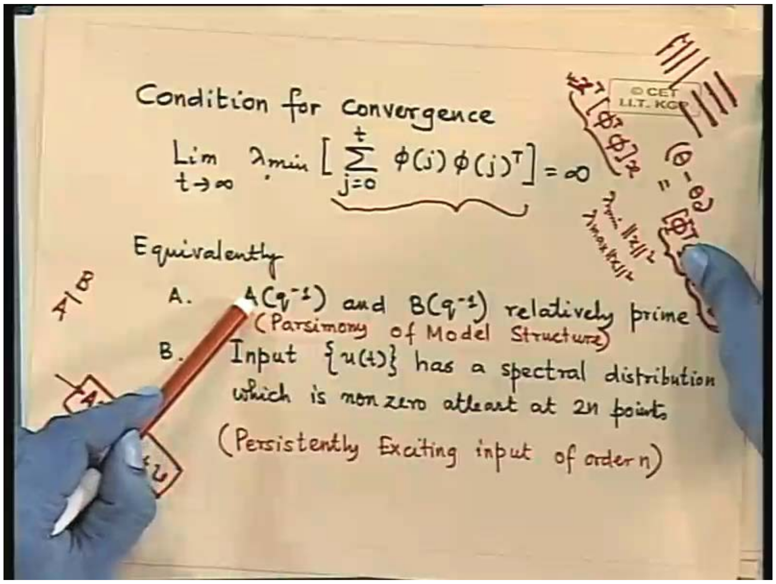


For example, now the second thing, why n should be unknown? Why there should be n sinusoids? Basic idea is let us say, consider first order system; how many unknowns, a and b, two unknowns. So we must whatever when we excite; suppose we are suppose we are given a box and you are told and you are given a sinusoidal signal generated and you are told that this is a first order system, so please identify, you are given a C R, no computer nothing. What will you do? You will apply a sine wave. So input sine wave you know, output sine waves you know, you will first measure the game difference; one volt whether it become half volt or quarter volt and then you will see the phase difference by feeding to double channels C R. So you see how much phase shift has occurred from the input. So so you are basically making two measurements, one is the amplitude gain and another is the phase.

So you have now you will try to feed it, so you will find a formula in terms of n b of the gain and a formula in terms of n b of the phase. So you have two equations and you have two unknowns; so you will solve, that is how will you get a n b? So if you excite with one sinusoid a first order system, you get enough number of equations such that you can solve it. If you excited with D C, will you able to solve it? You will get only one equation and you have two unknowns, so you cannot solve it, so this itself shows that; if you have similarly you can you can find, if you have a second order system, you require two because you have two poles.

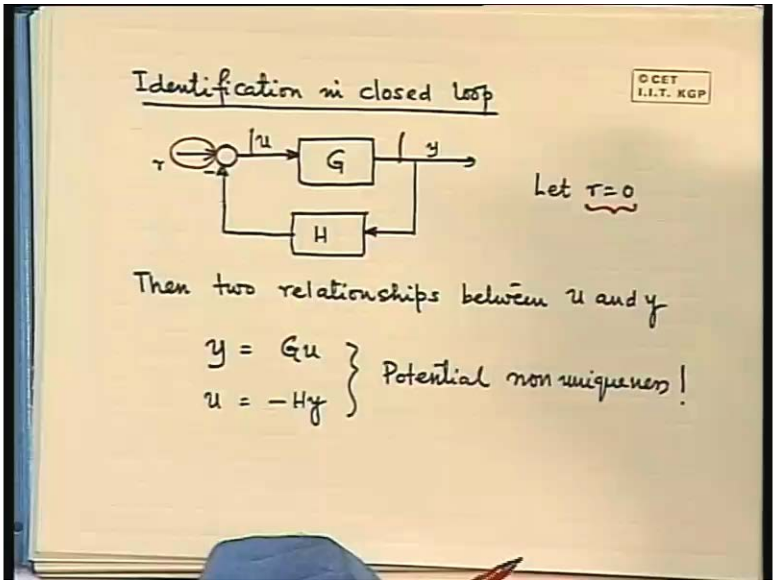
So if you have two poles, you need two sinusoids to excite them, right. So this is also intuitively very clear, it can be proved in a very rigorous manner using huge mathematics but the basic idea is that; this this also applied to a person if we really want to know I mean, you have to you have to interact with it properly, if you really want to know what a sort of person he is, without interacting, if we if we always speaking of cricket you cannot know whether he has whether he has interested in literature, so it is like that. So this is but remember that, these two are very very well-known conditions for for any identification, that the model structure should be simple enough, it should not be over parameterised; if if it is over parameterised then you encounter this this non-uniqueness of solution probably.

(Refer Slide Time: 22:10)



On the other hand if it is and you should choose, even you are after all where where from have you got that identification data; you must have got it from your experiment you want to identify the models, so you must arrange for some experiment, arrange for collecting some data. So when when you are arranging for the experiment, you have to have at least some knowledge and then you have to select your inputs properly; so that you have excited all the models of input, if you are not excited something, you you you will not get that.

(Refer Slide Time: 22:59)

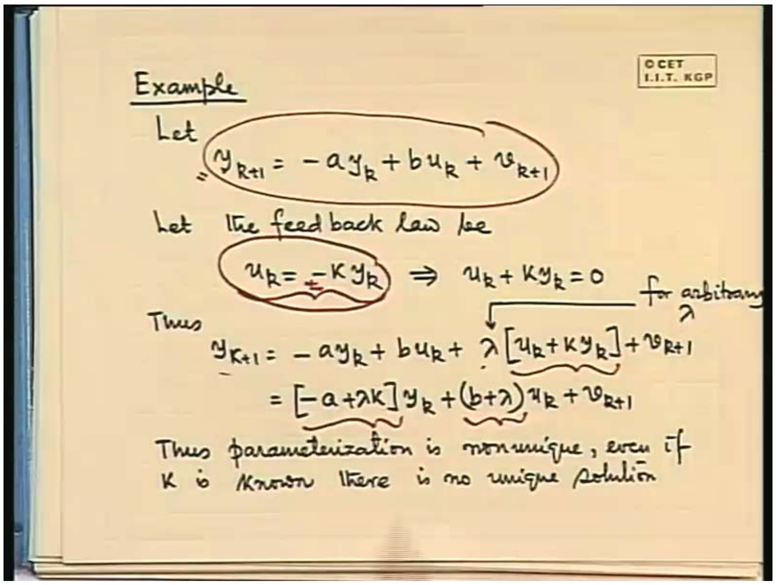


Now let us look at identification in closed loop. So this is our closed loop situation, right. For the time being let r is equal to 0, there is no referencing; this happens there there are some controls loops for which the control is I mean the set point is zero. One one example I know of with which I am involved is the lower loop of the of any error specific, there is a set point is always zero because you do not want it roll, you might you might want your vehicle to go up, to go down, this way, that way but it generally do not want your vehicle to roll. So your roll control set point is always zero, you want a zero role all the time, so that is such a loop.

Then there may be other for example, when you are having a linearized control then then you are always writing the control loop in terms of incremental variables. So if you are writing it in terms of incremental variables, then also your set then also your set point is zero. You want the increment from the operating point is zero, so your set point is zero. So it not a very uncommon situation; though I mean having a non-zero set point is also common, but if r is equal to zero then then something very interesting happens, what is that? If you have this to be zero then it turns out that, between u and y their exist actually two relations both of them are correct, y is equal to G u and u is equal to H y.

So if you identify between u and y which you do in identification, which one will you get? Both are both are correct. So so there is a possible potential non uniqueness solution, you are very likely to get a situation where this the these solution is going to be non- unique, right.

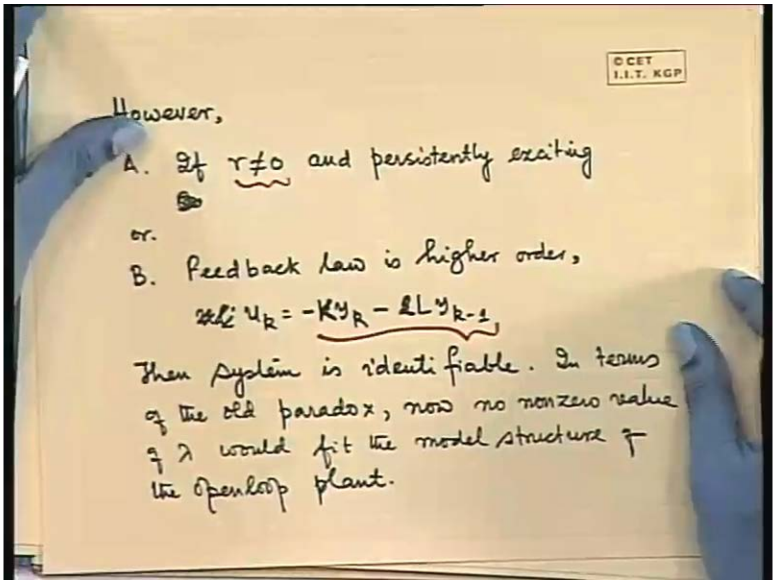
(Refer Slide Time: 25:14)



You will for example, if you take this system, simple first order system; this is u this is my g. So simple first order plan e and assume you have taken a simple proportional feedback, u u k equal to minus k y k, then obviously u k plus k y k equal to 0. So this equation, you can always add any amount of this term; that that equation is also satisfied, this lambda can be arbitrary because u k plus k y k is always 0, so this equation is also true. So if you breakup, you will get so so the open loop plan satisfies this equation for arbitrary values of lambda, and not only that worrying part is that even if you know k; even if you know the feedback law you cannot solve out because lambda is arbitrary, you do not know what lambda is, it can be anything.

So you will never get this is a typical case where there is multiple parameterisation, I mean non unique parameterisation in the presence of feedback, right. Now why does it happen; because this law is actually simpler than this law, so the data simultaneously satisfies this also but you are trying to identify the data using higher order model. You have taken a data which actually satisfies a simpler model but you are trying to identify it using higher order model that is the reason. So so what are you going to do?

(Refer Slide Time: 27:12)



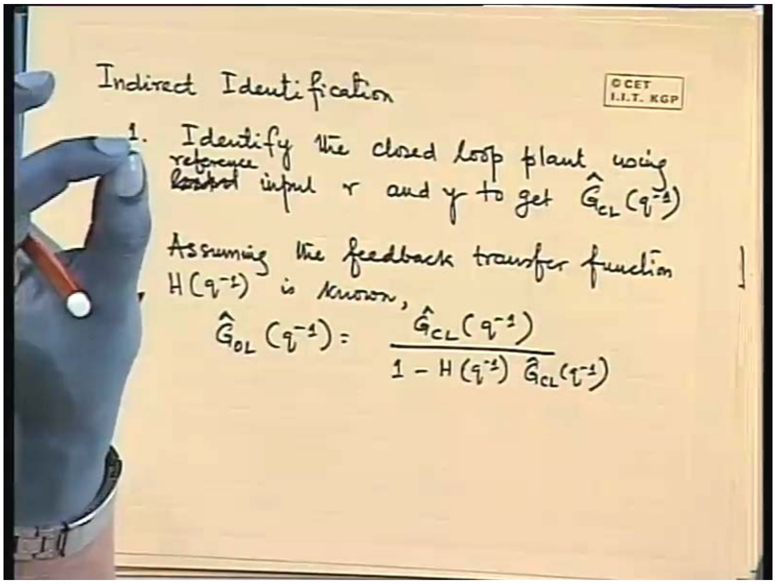
So you can do several things; for example, if this does not happen, if r is not equal to 0 whether if r is not equal to zero then this model structure, if you put suppose you had r then you will have a r here and you will have a r here, if you substitute you will get a minus lambda r, correct. So if you are now, if you identify using you are identifying what is your identification? Your identification is using this model structure, you are trying to feed the data using in in this form but when you have a lambda r term; obviously this this does not feed this model structure, so it is unique.

This model is this data will satisfy this model structure this model structure, only for lambda equal to zero, otherwise no other model; if I mean especially if r is persistently exciting and all then this data will the data which actually satisfies these can be identified using this structure, only if lambda is equal to zero for no other value. So you will get the right parameters.

Similarly you could also do another thing, you could also you could choose a higher order feedback; not only minus K y k, also minus L y k minus 1, if you choose that and now if you substituted the same thing, then here you would have got a minus lambda L and write y k minus 1. So now you now you model structure would have been second order, there is a polynomial order would have been second.

So so the data really satisfies a second order parameter, if lambda is non-zero but you are trying to identify with a first order model; so so so that will be satisfied only lambda is equal to zero, not not not otherwise therefore you have you again have unique parameterisation. So this shows that, there is potential danger of getting non unique solutions in closed for identification; especially when either the reference input is not significant and the control noise is is too simple, in such situations you have s potential problem. So the idea is that so what you are practically you going to do is that, you probably cannot do much on the control law because it may be already existing; but you have to have a you have to have a decent amount of reference input, which is significantly varying while you are being the experiment.

(Refer Slide Time: 30:30)



So how do you identify? If you if you ensure that then there is no problem and then so close loop identification has two problems; one problem is that generally the error will get tend to get correlated with the data vector, that is one problem that will give you some parameter errors. That problem you can encounter you can solve, by either choosing by choosing for you know those various mechanisms like; having a longer model structure or highly instrumental variable, you have to solve that problem, if the data vector tends to be correlated with the noise. On the other hand, it might lead to some structural problem, in the sense that the parameterisation itself may become non unique; if if it if that happens then you have to the most common solution is to have a significantly significant and persistently exciting referencing. If you have these two, you will get a decent parameter even when the system is operating in close loop.

So you know all systems cannot be operated in open loop; firstly if you go to a plant, suppose you suppose you go go to the factory and say please let me open the blast furnace loop I I want to identify it, you will be throw you out because all plants there there are plants which cannot be operated in open loop, nobody will allow you. So you have to have to operate it in closed loop, you might request the operator to you know give some set point pattern information, so that you can collect your data, you cannot do more than that

So so in such a case, you will be forced to identify the plant in closed loop. So there are there are two approaches; one approach is now it it may you do not know, whether if you want to do a kind of identification, if you want to do identification based on u and y then you first have to get u. That depends on again again it depends on whether you have transfused structure for it, y is generally measurable because because after all if the operator has to control y, he has to see it. So that value is generally available in the control room, command he himself is giving, generally available but whether the plant finally the input that it gets may be in the form of some exotic variable like; a very high current or may be some some some fluid flow that, may or may not be measurable, it may measurable in the field, it may not transfused, you may have so many so many practical difficulties.

So there may be situations where getting u and y for identifying a model is may not be so simple. In such a case; what you have to do is you you identify a model between r and y, reference input as y. So if you do that, what will you get? You will get the closed loop model, for why are you; so if you know the controller, if you already know the controller beforehand because it will be there in some manual return, after all some company came in and set up that control, even let say its digital controller. So if you know the controller and if you have identified the closed loop plant, then you can solve out the open loop pattern just numerical, solving, right.

So that is possible, so so so this this kind of identification is called indirect identification. Direct identification is where you directly identify the plant taking its input terms out, indirect identification is when you identify the closed loop plant and solved out the open loop plan, right. So this now we now we more or less have covered some most aspects of least squares, least square is a huge area; two-hundred years of research on least square started from Gauss, so enormous amounts of results but anyway we more or less in the context of identification, we have we have at least touched upon most points.

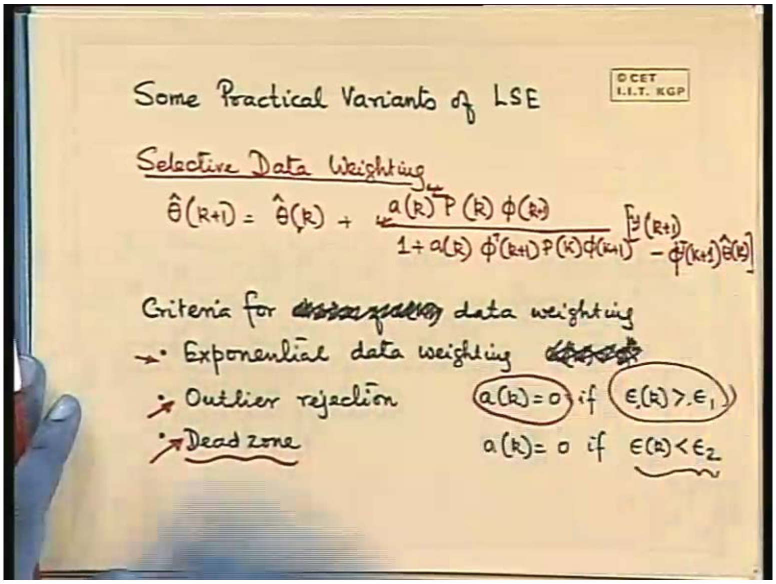
So now we just before leaving will will look at certain practical variation to the basic least square algorithm that you can do, sometimes you you need to do that; there there there could be various situation, we already I mean sort of got taste of that we we had seen the exponential data weighting case where we we wanted to weight in the least square criterion, we wanted to give different errors, different weights. Then we had also seen the instrumental variable algorithm, which also leads to the least square; the only thing is that in certain cases phi gets substituted by another vector called the instrumental variable vector, in this case we will also see some cases for example, in there may be some situations where you do not want update your data based on at each data point, why why could that happen?

One reason is that, you know you have you have modelled your noise, then okay generally noise is; for example suppose its so happens that, you know that the data is noise suppose when when you have when you have tested the data found then the data has generally has really white type of data from noise variant some point one. Now so I mean, weird things may happen; it may be a plant where sometimes may be when you took the data there is some there is some arc furnaces running around and there may be some welding done, when such things do I mean take place because there is a very high current flowing in each one of them, either either in electric arc welding or arc furnances or in very large motors, you get you get a very high amount of electromagnetic interference, suddenly coming.

So what I want to say is that, there may be situations in the data where for some short periods the normal property of the data is disturbed. So you get some samples which has for some reasons very erroneous, such data are called out layers. They they they do not confirm to to to your to your normal model of the data; the point is that you have to careful about the out layers, why? Because one out layer coming, suppose your parameter was nicely converging, it has it has come to the true parameter and and it is holding it is slightly oscillating around the true parameter; suddenly if one or two out layer data points comes, it will create so much of prediction error y minus phi transpose theta, that your immediately your parameter estimates will will get its jerk, they will just simply go away because those two data points some for some reason you got some bad data you should never update your model using bad data.

So you have to build build when you when you practically build a least square algorithm, you have to build in protections against these things. So one way of protecting should be could be out layer rejection; that if you know that that if your parameters have more or less converge, you cannot get more than this error, if you have good data.

(Refer Slide Time: 38:18)



If your prediction error, suddenly goes beyond and beyond some really some sort; you you can be conservative in your estimate but if it is goes even beyond that, then you do not update on that you simply assume that, this data has to come due to some bad reason, this is not normal data so I will not update.

So so then you can say that, I will set this a k is equal to 0 in this equation; if the error is greater than seven, you you states on their limit that this is bad data I am not going to update my parameters on that it will pass. Then then what will happen is that, simply theta k plus 1 will will remain as theta t and similarly t k plus 1 will remain as t k. You are you are just automatically, you just picking out one data point and throwing out.

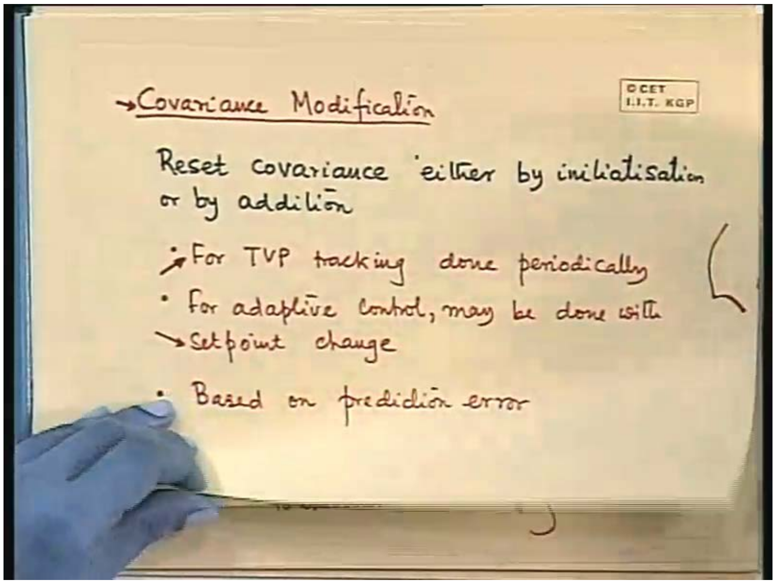
Similarly there may be sometimes, you you you also want to incorporate what is known as the dead zone; you in some cases, see the it is not good that all the time you are going to you may not like to estimate, I mean keep on estimating all the time. So if you for example, you may know that I have a noise of point one whole coming, there is a white type of noise of point one whole standard deviation; if you if you already have aware of this and if your prediction error is giving you a prediction error of point zero two then why should you update the parameter? Even even in the phase of the noise, your your parameter has sufficiently, accurately predicted the output that much of point zero two volt is coming because there is there is a much larger noise.

So as long as your prediction error is within the noise limit, actually your parameter is satisfactory; that noise will come anyway, that prediction error you cannot reduce because of the presence of the noise. So then you should not update again, you should not update unnecessarily all the time. So in such a case also; you might like a selective data weighting, set a k equal to zero if the error prediction error falls below a certain limit, such things may happen. So this you you you may like to build in this selective data weighting which will give you an inherent protection, suddenly your parameters will not go away.

See these things are very dangerous; I mean especially these things were were employed when people started working on adoptive control industry, especially in the process control industry people tried online estimating the parameter, online design the controller and then applying the controller. So if your if your parameters suddenly gets a shock then your control are suddenly gets a shock. I mean you will get a upset controller, so for sometimes suddenly your your products your your controlling some stripe width or some wet paper tension, if your controller suddenly changes then then you are going to apply, I mean meaningless input for a little while at least.

So during that time, it might it might jeopardize your operation. So so that is why people do not want such things to happen; better not change the controller during that time, let it continue after all how much it could have lifted again using one sample? The other thing is covariance modification.

(Refer Slide Time: 41:44)



See sometimes actually what actually; it is the covariance is the gain of the least square estimator. So if the covariance goes to zero then what happens is that the controller, that the estimator gain goes to zero. Now it may happen; that because you know covariance is what covariance phi transpose phi inverse, roughly speaking, not roughly speaking exactly speaking, so that that obviously goes to zero.

So now the the problem is that, suppose suppose you you are having a case; where the parameters are you know parameters typically change with what, set point, the linear model parameter of a boiler which is operating at fifty which is operating at fifty percent load are different from a parameter from from parameters of the boiler unit is operating at hundred percent load. So and and and the and the operator changes the operating point from time to time may be over two, three, four, five times in a day. If it if it is a power plant boiler, right in the morning till about eight o clock it will have one; it will have low set point, at eight o'clock everybody will will start going to office.

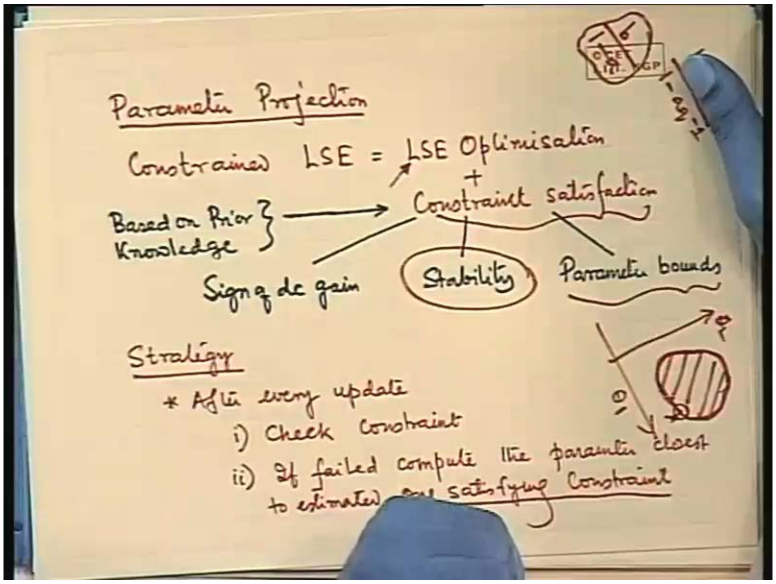
So the load will increase, everything will start. So that time he will increase the boiler setting to may be initially fifty percent then may be ten o'clock, it may be it may be to hundred percent again at three o'clock or it may be lower, again during evening he will make it higher. So he continuously changing his operating point not too frequently but at least five six times in a day; so at each of those points the linear model changes. So your, what suppose; you your your your you are running on online estimator, so what we are going to have is that the parameters are converged and your covariance has died, it will die, now suddenly the the operator has changed in operating point. Now now the model does not fit the model does not fit, so there is some error generated but your covariance is now is gone. So there is though the the error is crying, change parameter change parameter; there is no strength in the covariance to push the parameter to the new value.

So if you if you expect parameter change then you should always keep your covariance live; at least those incident, because if the covariance are not live it will not going to push the parameter, it is the gain. So that is why in fact you can see that in exponential data weighting, we do precisely the same thing because there we are expecting a time variant parameter, we keep the covariance from dying. So the covariance will not die because there is a one by lambda factor in the covariance update and lambda is less than one. So each update we are giving is a little boost, so it so it does not go to zero.

So that is what we do in time varying parameter, every instant we give a boost; for some times when when when we expect a set point, that is if if the prediction error goes beyond the certain value and may be if it is stays for two three instance. If it goes up just for one value, it might been an out layered; but if it is stays that means your model is consistently bad, for five sample you are getting large prediction error which means, that the parameters need to be changed at least you need to retune your your estimator, boost the covariance, again set it to some may be may not be exactly like an initial value but give it a strong value, right.

So this is also required very much if you do not do it then after sometime, you have you you will not able to. It will take a long time for the parameter to track. Third thing is how do we incorporate, if we know something about the process we must incorporate it in our parameter estimator.

(Refer Slide Time: 46:05)



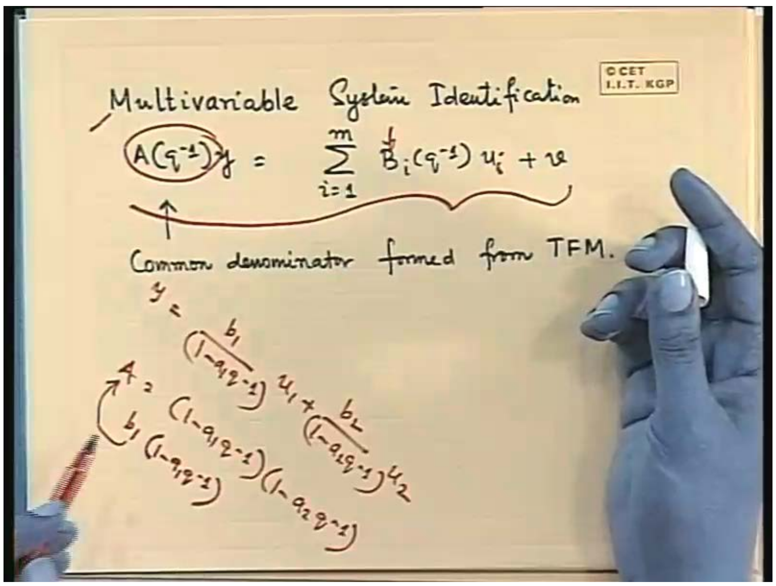
If we know that the process has a positive d c gain, we should have an estimator which all forces that the d c gain is positive, now how will you get dc gain? For example, a q inverse suppose you are troubled in model is b q inverse by 1 minus a q inverse, what is the d c gain? b by 1 minus a. So you are you might say things like b by 1 minus a should be positive, that that you could have such conditions you could have conditions you you know the system is stable.

So at any condition, if you find that the you are you are you are getting such an a a q inverse which has unstable poles; obviously it is not correct that is the a priori knowledge. You could say that this parameter cannot go beyond beyond this limit, such knowledge may exist. So so in which case you have to you have to in that in that case you have to have not only just least square optimisation but plus constraint satisfaction.

So at every instant with the the parameter that, you get you have to inspect whether you satisfy the constraints; if it does not satisfy the constraints, if you if you are directly the parameter which is given by least square does not satisfy the constraint then you have to find the closest parameter to that which satisfies it. And that is that is achieved by what is known as projection. So you have a part of the parameter space which will satisfy the constraint. See you have a part of the parameter space; over all suppose this is theta 1, theta 2 parameter spaces, two dimension just assuming.

And suppose you have given a constraint, that the this is the visible part. So every time suppose; if you get something here then you have to find the closest parameter which falls in the region, why closest? Because otherwise after all how do you get this parameter? You got this parameter by L S E optimisation. So it leads to low prediction error that is how you got it. So you must maintain low prediction error as well as satisfies this constraint, right.

(Refer Slide Time: 48:34)



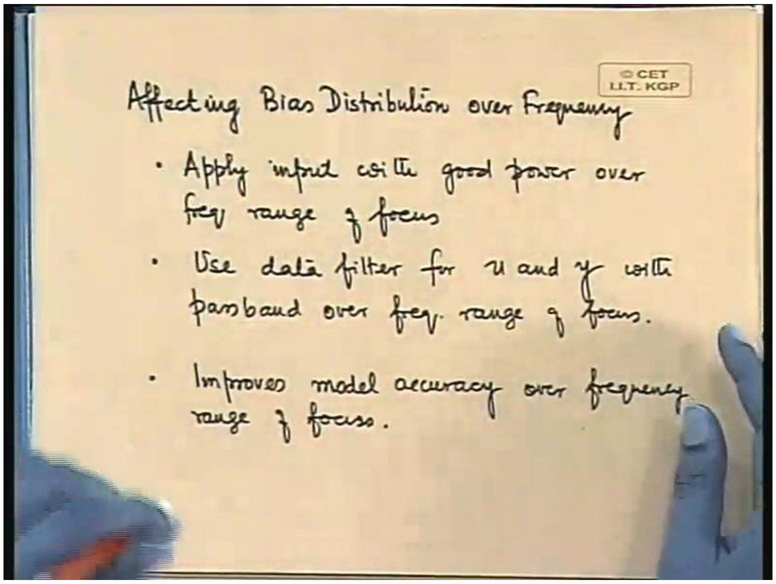
So that is about all, two little things firstly; we did talk about multivariable that is very simple, actually for a for a multivariable system your parameter estimation equation will become like this, this is u i. So rather than having A q inverse y is equal to B q inverse u, you will have a number of B s, there is nothing very unusual.

So the multivariable identification can be done exactly in the same manner, sorry and this is the common denominator; that is if you have b 1 by 1 minus a 1 q inverse u 1 plus b 2 by 1 minus a 2 q inverse u 2, y is equal to. And you have to take common denominator, so your A will become

1 minus a 1 q inverse into 1 minus a 2 q inverse. And your b 1 will become this this b become b 1 into 1 minus a 1 q inverse; note that in that case, this and this will have will have a common pole, yes it will have but all these parameter that is capital B 1, capital B 2 and A should not will not have one common factor. Between B 1 and A there will be a common factor, between B 2 and A, there will be a different common factor, between B 3 and A, there will be another common factor.

It will not happen that, B 1 B 2 B 3 and A all have one common factor; that will not happen because between B 1 and A the common factor is 1 minus A 1, between B 2 and A B common factor is 1 minus A 2, right. So so the for for for multivariable that is the condition, that all of them should not have a common factor; so that is simple. All all all I want to say will have going into it is that the multivariable cases nearly similar.

(Refer Slide Time: 50:45)



Third thing is last thing is, no; may be may be I should stop here, somebody waiting here. So just wanted to stay that, there may be situations where you want to focus your estimator towards a certain frequency bands; you want to know the low frequency properties very well, do not care some much about high frequency properties. In such a case, the idea is that you have to you have to apply input which is reached low frequency, we cannot give a high frequency input and expect good accuracy in your low frequency model; and to improve that further, you should use a filter which will further focus the input output on to the frequency band. If you do that then on that focus frequency band, you will get good accuracy, compared to other bands.

So that is is all for today, thank you very much.