CH5230: System Identification

Estimation of non-parametric model

Part 2

Now, we also learnt WLS, remember. The WLS thing comes into play under two conditions, right? One is, what are the two situations, scenarios? There are many scenarios in which you may have to do a weighted least squares formulation. But can you name at least two scenarios? Sorry? So one is, that

the errors are heteroskedastic. Earlier we stated for OLS, you obtain efficient estimates if the errors are white, right? And it goes without saying there that it is, it has uniform variance, that means variance at all times, same variance at all times. Its variance doesn't change. Now, many times you may be fusing, you may be obtaining data from sensors, either from different sensors, if you recall weighted least squares, or from the same sensor but at different times with different variability, right?

You may be fusing data either that has been collected over a period of time from a single sensor or you may be fusing data that is coming from different sensors at the same time. Either way you can run into this so-called heteroskedasticity, as I've talked about in the least squares, where the variance keeps changing with time, in which case the OLS gives you inefficient estimates. Will consistency be at stake if the errors are heteroskedastic? What do you think? You understand, I mean, you must be able to recall what is heteroskedastic. The variance keeps changing with the data points. The data points could be positioned in time or positioned sensor wise, doesn't matter. The error at each k has a different variability. Will that put consistency at risk? Yes or no? Yeah, it should not affect the correlation between the regressors and the residuals, right? So, consistency is not the issue when you have heteroskedastic errors. What you have issue is that you will get inefficient estimates. Efficiency is the issue.

So weighted least squares takes into account the nature of the varying data, you remember. And what is the other scenario in which you want to have weighted least squares? Correct. So coloured equation errors. Because once again, story is the same. Coloured equation error does not mean that you will have necessarily an inconsistent estimate. Consistency will not be necessarily at stake. It only means that, that you may get inefficient estimates. But it can be both ways. As an example, suppose I'm fitting the data generating process is simply some a times u[k] minus one, plus e[k]. So this is one scenario. Sorry, let's say, v[k], v[k] is coloured. Let me specifically write, plus e[k] plus c1 e[k] minus 1. This is the data generating process, a not -- let me use b not, b1, not, the true value. This is the DGP, let us say, again, an FIR model but very simple FIR model. But the errors are coloured. And suppose I fit a model, as y[k]equals, let us say, I fit it correctly. Now let me ask here, so I have b not u[k], I don't know the delay. And I'm going to fit b not plus b1 u[k] minus 1 and plus e[k]. Typically in FIR model we assume this. For example, in the MATLAB's impulse test, it doesn't necessarily assume this to be e[k]. But typically, you assume this to be e[k].

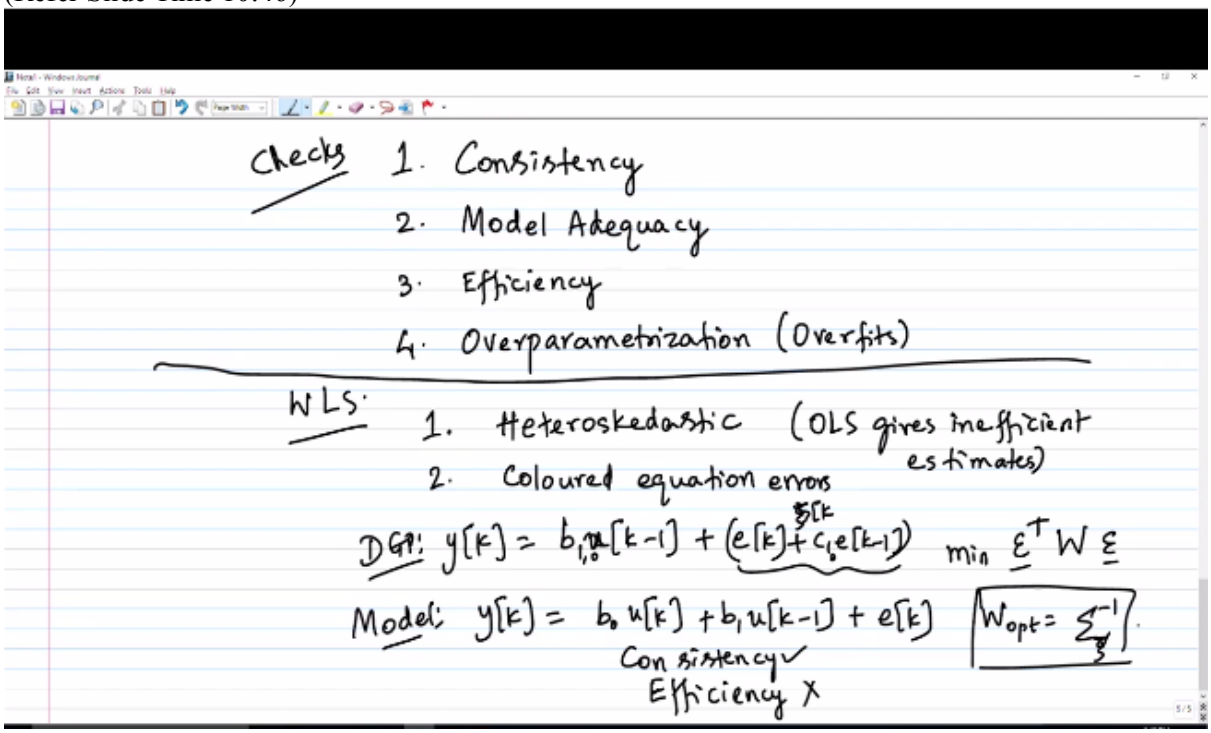
Now in this case, will I obtain consistent estimates of b1 and b not? I made a mistake in the error, right? Will that affect consistency? Yes or no? What do you think? Yes or no? It will effect consistency? Are you sure? Why? Sorry? What is it that I'm leaving out? What is it that epsilon k that I will have? Theoretically? That is what I'll -- minus b not u[k], yeah, but b not may turn out to be very small, right? Then what do you think? What are correlated? Why? I don't follow. Why would the thing be uncorrelated? See, what is it that I'm leaving out? Okay, really? It is just, although I included this additional term. Well, actually what I'm leaving out is c1 not e[k] minus 1. In fact, the residual will be simply e[k] plus c1 not e[k] minus 1, right? That is what I will obtain because b not will be almost zero. In this case, consistency is not an issue at all. Because whatever I'm leaving out is not correlated. Under open loop conditions, you won't have any issue. So consistency is not at all an issue here. What about efficiency? Right. Right. No, but you can always say that the true model also has a b not u[k], with b not value being zero.

So structurally, you can always include that. Sorry? No, it won't change because what happens is, so you can always write this as zero times u[k]. What you can't do is, when you have under model, when you have under model, what happens is, a significant portion of it will go and sit in the residual. Suppose I had simply written b not u[k] plus e[k] as my model, then there's going to be a problem.

Always, when I have under model, there is more of an issue than over modelling. What about efficiency? Will I get efficient estimates of b1 and b not? So, now whatever I'm leaving out is coloured, right? The residuals are going to be coloured. So now in this case, efficiency is not guaranteed. It's going to be y minus y hat, y hat will not contain e[k]. So the left out will be e[k] plus c1 not e[k minus 1], because I'll never be able -- model is this, left out will be based on y hat, right?

Y hat will not contain e[k]. So what you leave out will be e[k] plus c1 not e[k minus 1]. So in this case, I will not get efficient estimate. So this is the case when I have the coloured, and in this case I can use a weighted least squares formulation. And all the weighted least squares does is it weighs the observations, and if you recap, what we have said in weighted least squares is optimal weighing is simply the inverse. So if you were to write in terms of the prediction, the errors, one-step ahead prediction errors, it's essentially epsilon, in fact, let me write it this way. Epsilon transpose W epsilon, then if you're going to minimize this, this is the weighted least squares objective function, where epsilon is your stacked vector of errors that you're making one-step ahead prediction errors. Then the optimal W, if you recall, for obtaining efficient estimates is sigma v inverse or sigma z inverse, if you want to call this as z[k]. So that is what is the story of WLS.

(Refer Slide Time 10:46)



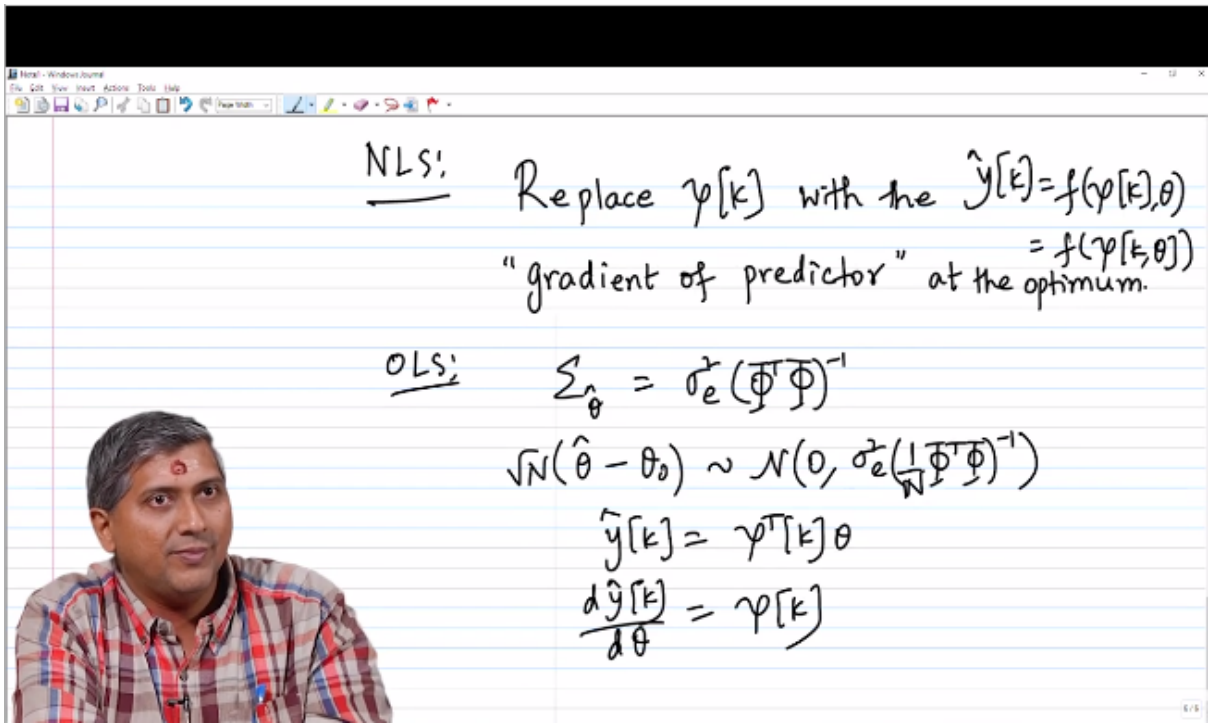
And the other thing that we have studied is nonlinear least squares. The nonlinear least squares, it gets a bit more complicated, but it is simplified, if you recall the perspective that I had given in the lectures. The difference between linear least squares and nonlinear least squares is, now the predictor is a nonlinear function of the parameters, that is what is important, understand. We are not worried about non linearity with respect to regressors. The nonlinearity is with respect to the parameters. For example, I can fit a polynomial also using linear least squares. I can have y[k] equals some alpha1 u1[k] u[k] plus alpha2 u square k and so on. That is not counted as a nonlinear least squares necessarily, because I can always have regressors as u[k], u square k, and it becomes a linear least squares problem. So, linear in parameters and nonlinear in parameters is what we are worried about all the time.

So, the best way to remember the nonlinear least squares results, as you recap, is to remember the linear least squares results and replace wherever you see the psi k, sorry. So wherever you see the regressor psi k, replace it, replace this -- sorry. So replace psi k with, what is it? Do you recall? With the gradient of the predictor. All the results that we learnt in linear, ordinary least squares or linear least squares pretty much carry forward just by recalling those results and replace psi k with a gradient of predictor at the optimal point. In fact, if you recall, we said, for example, in OLS, comparing notes with OLS, we said that sigma theta hat, which is the variance covariance matrix for the parameter estimates. Do you recall the expression? What did we have? Sigma square e times phi transpose phi inverse, right? This is the expression that we had. There's no m in here. Is there an m missing? What do you think? Is this correct? You remember this expression? Correct. Okay.

So this is the expression for sigma theta hat and we also said that theta hat follows, in fact, we say that root N theta hat minus theta not follows a Gaussian distribution asymptotically with mean 0 and variance? It cannot be 1. 1 over N, in fact, what do you get? If sigma theta hat has a variance of sigma square e times phi transpose phi inverse, then this would be sigma square e times 1 over N phi transpose phi inverse. Right? If theta hat has a variance of sigma square e times phi transpose phi inverse, what will be the variance of root N theta hat? It'll be N times that, right? And I just observed N as 1 over N here. Now, the same thing applies to NLS as well. All you have to do is replace phi. How is phi constructed in OLS? From the vector of your regressors stacked from K equals zero to N minus 1. So if you recall y[k], this is the regressor equation that we have. In fact, this is -- sorry, the predictor equation that we have. And in fact, if you differentiate, so dy hat by d theta is actually psi[k]. That means in the linear least squares, the regressor has, if you recall, is nothing but the predictor gradient. You take the same idea and now apply to NLS. The regressor in linear least squares is a fixed one independent of theta. And rather than calling it as a regressor, we call it as a predictor of the gradient.

Whereas in NLS, the regressor is a nonlinear function. The regressor is also a function of theta. That's why the predictor becomes nonlinear. Remember, if you recall y hat of k is a nonlinear function. For example, the regressor is also a function of theta. It could be that there could be a very complicated function here. One example is that, okay, let me write the most general one, that it's a very complicated function of the theta and psi, or it could be that the regressor itself is a function of theta. We don't know. It could be one of those. Either way, when you take the derivative of this predictor or the gradient of the predictor with respect to theta, you will obtain a local regressor. So all you have to do, think is, if you recall the Gauss-Newton method that we talked about, essentially the Gauss-Newton method is also locally linear least squares problem based on this idea. So you can simply carry forward the OLS results for consistency and for efficiency, that the NLS estimates are consistent if whatever you have leftover, that is the residuals are uncorrelated with the predictor gradients at the optimum. Of course, you will never get the global optimum. You'll only get the local optimum because you have to employ numerical optimizers. So you evaluate the predictor gradient at the optimum, which will become your local regressor matrix and you correlate the residuals with the rest.

(Refer Slide Time 17:50)



If there is no correlation, then you will get consistent estimates. Likewise, efficiency as well. Efficiency is pretty straightforward. Efficiency is again, NLS estimates will guarantee efficiency if whatever you have left out are white. And asymptotically we know from NLS literature that you will get Gaussian distributed estimate. So this is the recap. And in this process I've shown you how to apply and interpret least squares for estimating FIR models. What I'll do in the next lecture is that I'll just go over this specifically in the context of FIR model estimation. We have already discussed quite a bit in detail, but I'll go a bit more in detail, and then we'll talk tomorrow exclusively on two things, least squares estimators with regularization, which we have not recapped here. In fact, we have not discussed so much even earlier. And secondly, what we shall discuss is the frequency response function estimation. That will complete the estimation of nonparametric models and then we'll move onto parametric model estimation, okay. Thank you.