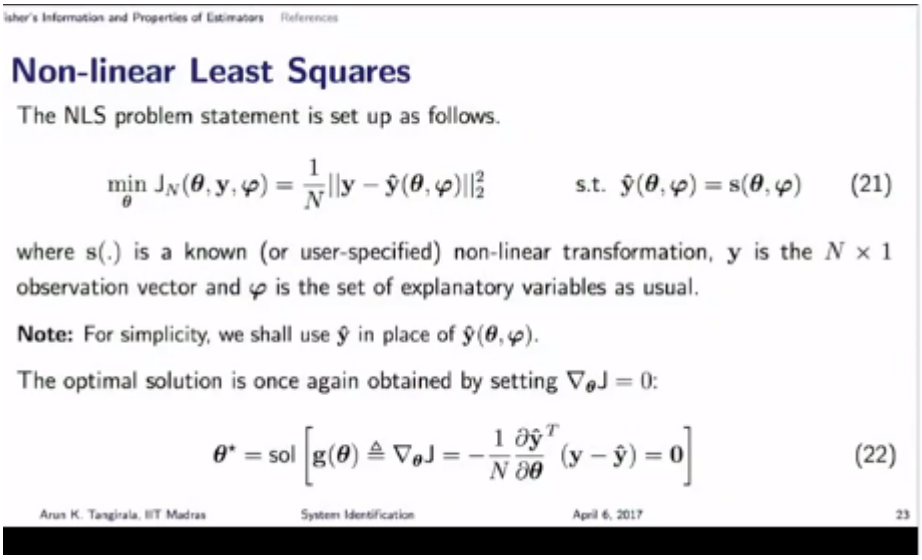
CH5230: System Identification

Fisher's information and properties of estimators

Part 13

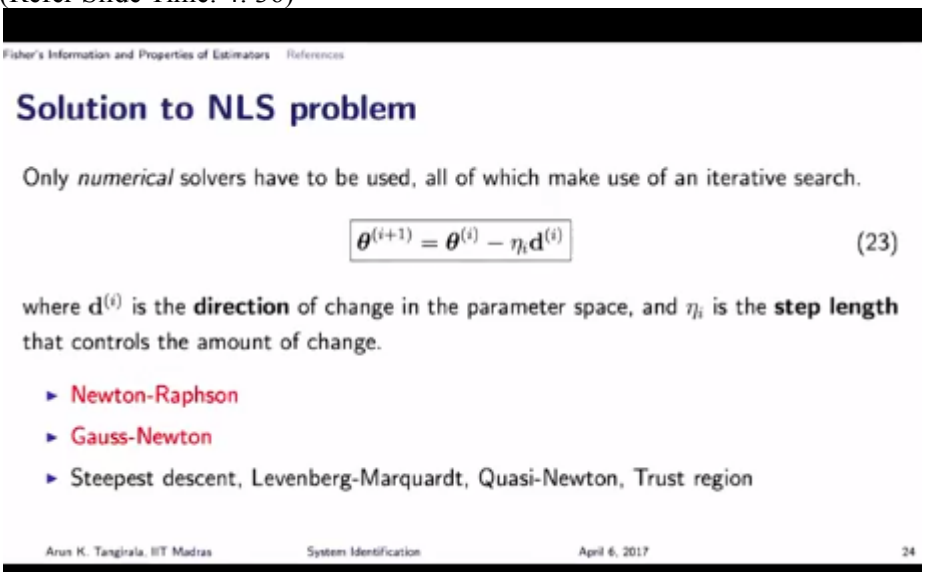
(Refer Slide Time: 00:13)



Let's now, just finish very quickly the NLS version. So there are many variants of least squares by the way. Right? If you turn to literature, you will find all the way from -- I don't know if there is a ZLS, maybe some zonal least squares or something like that, or Zen least squares. I'm not so sure, but almost all alphabet prefixes have been used up. ALS, you have alternating least squares, you have partial least squares, total like partial eclipse, total eclipse, total least squares, ordinary least squares not so ordinary least squares, weighted, nonlinear, extraordinary least squares, generalized least squares, everything. Seriously. So we are always, as I said, researchers are notorious for that, right? You come up with a method, and then you will have all prefixes attached to those acronyms that people come up with. I'm not going to discuss all those variants because they're not necessarily relevant to us in this course. Weighted least squares is relevant, so I discussed that and then nonlinear least squares. That's obviously very relevant. Now, there's not much to discuss about nonlinear least squares. Again, the lectures I have told you in detail. There are only two things to remember about nonlinear least squares. The first thing is there's no analytical solution, unlike in your linear least squares, which is actually, which may sound very bad but it's actually good news for all of us, for all of you particularly, because you don't have to remember the formula. And that there is not much work that you can do with pen and paper, to not only derive the solution but also to derive the properties of the estimator. It's not so easy because there's no close form expression for the solution. What you end up with in linear least squares is when you differentiate the objective function with respect to parameters, you end up with a set of linear equations and linear least squares or OLS. And that's how you have a unique solution, whereas with nonlinear equations as the bottom equation shows, you end up with a bunch of nonlinear equations, finding unique solution of which is not a trivial problem. And that's why you have to return to nonlinear optimizes, right? So what you will do in practice therefore is use a numerical optimizer -- work with a numerical optimum, in which case there is no guarantee that you will get a global optimum .You'll end up with some local optimum and that local optimum depends on how much time you have left with to submit the assignment or the exam and so on, right? There is a tolerance criterion that we say that if you're done -- if in five seconds whatever solution you get, give me the solution, that can be tolerance criterion also. But anyway, you're quite familiar with the solutions to nonlinear equations at some point in time, you've solved. Those are all the things that apply here. Sensitivity to initial guesses. So there is a whole lot of literature on what kind of initial guesses should be given. Secondly, computing gradients, because you're going to work with a nonlinear optimizer, you will have to generate gradients. And also sometimes you may have to generate Hessians, but all of that depends on what algorithm you're using, right? Generally, as I said, because you're working with nonlinear equations, you will work with iterative algorithms. General

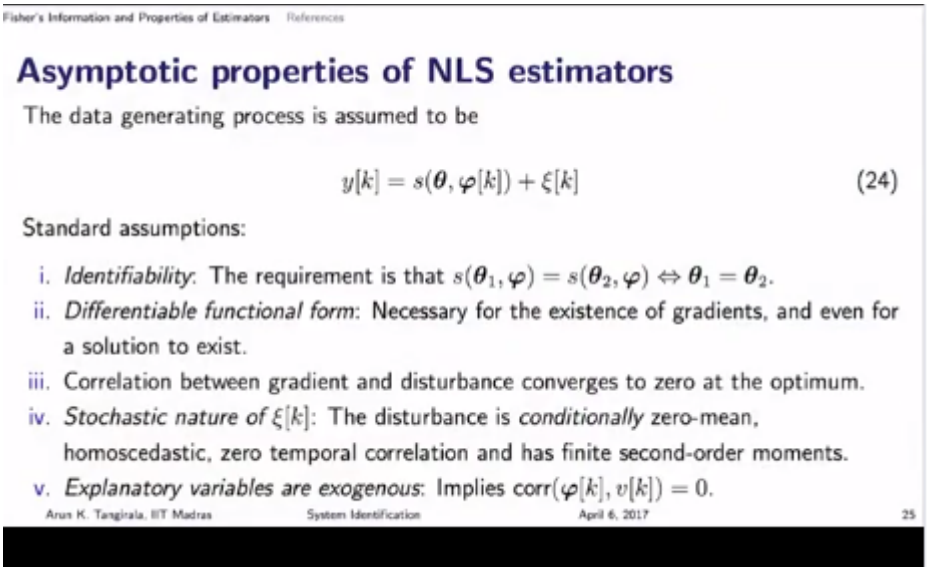
expression is what we see and then, as I said, must have listened to the lectures, there is Newton-Raphson method, there's Gauss-Newton method and then you have Levenberg-Marquardt, Trust region and all kinds of optimization methods that you have. The Gauss-Newton or the modified Gauss-Newton and Levenberg-Marquardt algorithms are quite famous. If you have more complicated problem then you need a more complicated -- sophisticated solver. The Gauss-Newton algorithm, as I've explained in the lecture, is very simple. At each iteration, it's also linear least squares problem. That's the beauty of a Gauss-Newton algorithm.

(Refer Slide Time: 4: 36)



And then you have a certain factor to control the rate of convergence and so on. So that is the first point to remember about nonlinear least square. No analytical solution, you use non -- you know, numerical optimizes, which employ one of these algorithms and so on. And the other thing to remember is, although I cannot assess the properties of the estimator very easily, I mean, if you've watched the lectures, you listen to those lectures, there are a bunch of conditions under which the properties of the estimators are given. What are the properties again of interest? Consistency, efficiency, and then the asymptotic distribution of the estimates, which will help me in constructing the confidence intervals, correct? So there are some conditions under which you can give those results asymptotic properties.

(Refer Slide Time: 05:25)

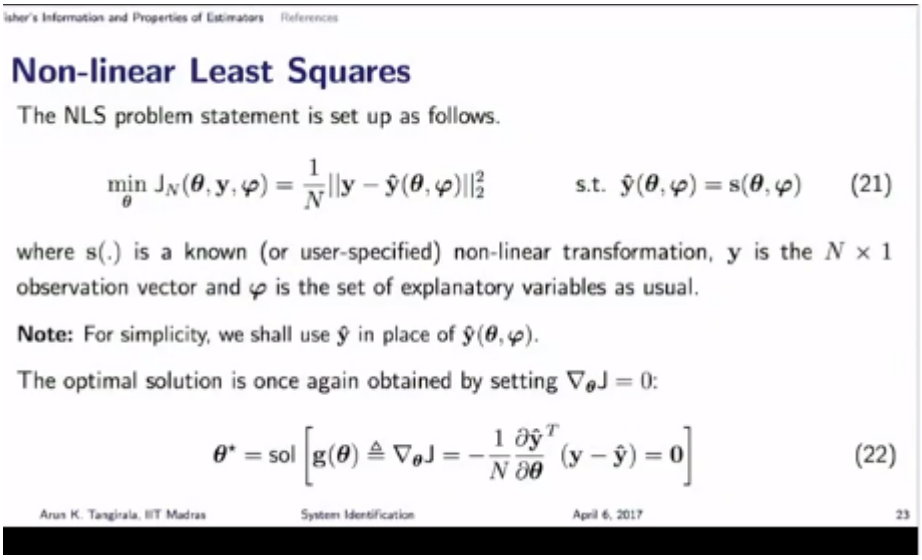


Now, very often these are, these can sound very intimidating, that is a conditions under which the properties have been given. Maybe as identify ability, continuity, stochastic differential ability, exogeneity, and so on. If you understand what they mean, then it's easy to remember. All it says is that there should be unique solution, the objective function should be differentiable, there should be no correlation between gradient and regressors. Okay. Now, earlier, we used to say, sorry between gradient and residuals. Here we say disturbance, but in the linear least squares, what is the condition that we said for consistency, no correlation between residuals and, or you can say z, if you say is z is the distance, between disturbance and regressors. We use the word regressors, here we are using the term gradient. And that is the catch, the best way to remember nonlinear least squares is, it is almost a linear, I mean, least square problem, where you replace the regressors with the gradient of the predictor, right? That is the key to remember. Now, if you look at the linear least squares, the predictor, assuming that z is white, this is a predictor, right? This is what is a linear regression problem is. If I were to construct -- take the derivative of this predictor with respect to theta, let us say that I have a vector of parameters here. What would I get here? In fact, here, I can use d, since I'm differentiating with respect to the entire parameter vector. What is the answer to this? What happened? You don't know how to differentiate the vector? Y hat is not a vector, but you're differentiating with respect to a vector of parameters. What would be the answer? Well, first of all, this vector consists of this. Now what is the answer? Psi you mean? [8:04 inaudible]. Now, it has to be that right? Because what would this work out to be, the first element, the first regressor, up to p th regressor. That's your regressor vector. So your regressor vector now has a different interpretation. What is the interpretation? It is a gradient of the predictor. In nonlinear least squares, you have to remember there are two different gradients. One that you will work with. One gradient is, the gradient of the predictor. What is other gradient? What is other gradient that you work with a nonlinear least squares? First of all, why do I run into gradient of the predator? To minimize? No. Why do I even run into the gradient of the predictor?

[9:22 inaudible].

Exactly. So there is an objective function whose gradient I evaluate and set it to zero, right? How do I get the equation that I had previously?

(Refer Slide Time: (09:34)



This equation where did this come from? The equation at the bottom? That comes about because I say the first derivative of the objective function should be zero. Now, hopefully that will get me a minimum. So there are two gradient -- when I am evaluating the gradient of the objective function, what do I run into? The gradient of the predictor. You should see that clearly. See, you are minimizing y[k] minus y hat of K which is a function of theta whole square. This is what, you can say one over n, doesn't matter. This is your j, when you take the derivative of j with respect to theta, what do you get here? One over n, sigma two times y[k] minus y hat of theta, I mean, y hat of k, theta, function of theta, times what? Is that correct? Is that complete? Have I written the expression correctly or there's something missing? This is simple differentiation. Are you telling to yourself?

Differentiate y hat.

With respect to what?

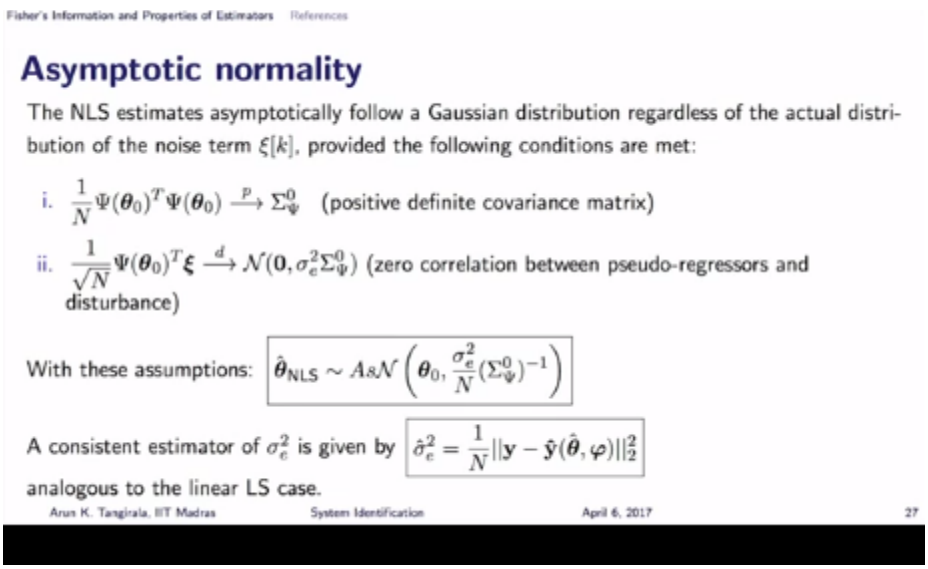
Theta.

Theta. Exactly. And then do what? I mean, do I divide, multiply? Multiply. So I have times this, minus dou y hat by dou theta. Correct? Now, in order for me to find the optimum, I have to -- so I do this for all thetas. I get the p equations. If y hat is linear in theta, what kind of equations do I get? Linear equation. That's what your linear least squares is. And that's why you're getting unique solution. When y hat is a nonlinear function of theta, I get a bunch of nonlinear equations. How do I arrive at those nonlinear equations? I have to know dou y hat by dou theta. It's very important to understand this. Because when we talk of the prediction error methods and so on, at that point, I'll show you with an example because in Armax model identification, you will run into a nonlinear least squares problem. And, in the code, whether you are aware of it or not, there is evaluation of this gradient, of this y hat with respect to theta. So, now, you should understand why we are bringing up dou y hat by dou theta. We are bringing up dou y hat by dou theta because we are evaluating the gradient of the objective function. So, there are two gradients, you should not get confused between these two. One is a gradient of J, which is objective function, other is a gradient of the predictor y hat. Now this dou y hat by dou theta, as I said, for linear regression case, is simply the regressive. What happens in the nonlinear case when y hat is a nonlinear function of theta? What happens to the gradient? Is it independent of theta? It's a function of theta. So at every iteration, you have to keep evaluating this gradient. All right? And many optimization nonlinear least squares algorithms may ask the user to supply this or they'll try to numerically evaluate it. It turns out that for system identification,

fortunately, this dou y hat by dou theta, I can write an analytical expression. It is still a function of theta, as I'll show you in the maybe hopefully the lecture tomorrow that you can write an expression for dou y hat by dou theta, it will be a function of theta, but there is an analytical expression. In general, in nonlinear least squares, there is no guarantee that you will be able to write an expression for dou y hat by dou theta. Okay?

You need to know this because tomorrow if you want to write a nonlinear least squares solver estimator for Armax model, or B-J model estimation, or OE model estimation, the first thing that you will be required to do is to write a function that will return the gradient of the predictor, at each iteration, right? At each iteration, you're going to be required to compute this gradient. Now, let's close the discussion on nonlinear least squares by making this important observation. Whatever we have learned for linear least squares. What did we say? First we said for efficiency, the residuals should be white or the disturbance, should be white, applies to nonlinear least squares as well. I'm just summarizing the results. Two, for consistency, v consistency there should be no correlation between the regressors and z, whatever you're leaving behind, residuals. In the nonlinear least squares, you can replace regressors with the gradient of the predictor. Because the regressor is nothing but dou y hat by dou theta. So, what nonlinear -- the properties of nonlinear least squares say, the results say is that, at the optimum, whatever optimum that you have, typically, it should be the, at the global optimum unique, which you will never be able to find. But theory will give you all the ideal results. It says, at the optimum, there are two things that are required.

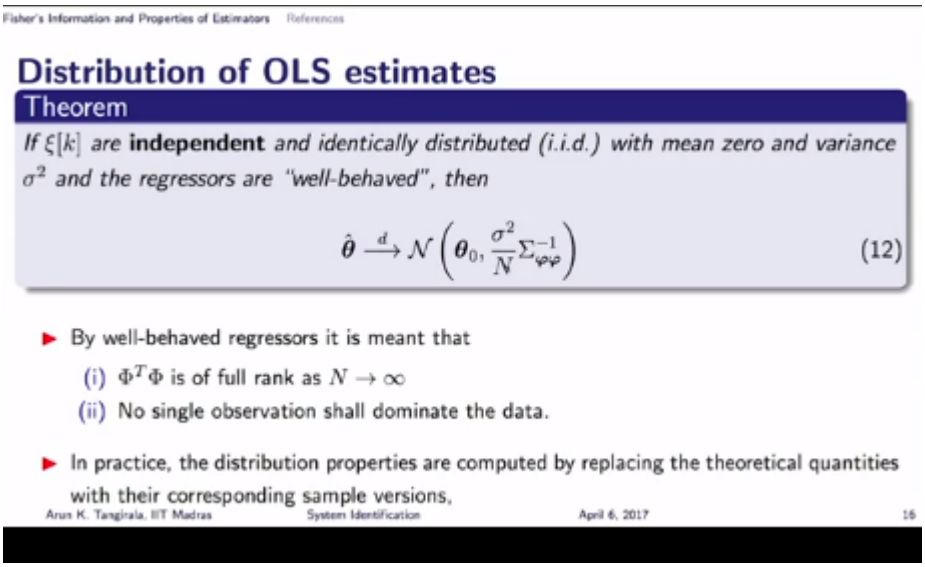
(Refer Slide Time: 15:39)



One, of course, that we, remember in linear least squares we said phi transpose phi should be full rank. Instead of phi, what do we have here, the psi. What is the difference between phi and psi? Not much. Phi in liner least squares, I directly construct, whereas psi here, I have to evaluate the gradient with respect to theta and then construct. But they are equivalent. In the linear least squares, the big psi matrix that you see on the screen is the same as phi, right? In many movies, the hero is so much in love with the heroine that any girl he sees he only sees heroine in her. Okay? Hopefully, I mean, he doesn't proceed further, but here also, if your heroine is least squares, OLS, linear least squares, then in the NLS, wherever you see big psi matrix, you should phi in it. They say, [thuj mein dekhta hoon] [16:45] and so on. So, [thuj mein phi dekhta hoon], [16:50] that's all you have to say. It's in the big psi you only see phi then you're okay, you'll understand the results. Yeah, okay, phi transpose phi, should

be full rank. That means the covariance of regressors there should be non singular. Here, covariance of the gradient should be non singular, that's all. Likewise, they should be no correlation between the residual and the, there we say regressors, here we say gradients. Which gradients? Of predictors. That's all. Otherwise, if you look at even the asymptotic distribution properties, look at this. There is so much similarity between this result and what you saw earlier, right? So, you remember this boxed result here for the asymptotic distribution and recall since we are in the same thing here. Look at this. This is for the linear least squares.

(Refer Slide Time: 17:45)



Do you see any difference? Not much. Same, theta hat, OLS is an asymptotically unbiased estimator of theta hat and here you have the variance, right? Same, sigma square e by n times the inverse of the covariance of the reggressors. Whereas with nonlinear least squares, same. Sigma square e by n times inverse of the covariance of gradients. So as long as you remember this parallelism, you're okay. Right. So, as I said, this is all to do that I have to recap for least squares. When we go into specifically, you know, when we talk of prediction error methods, the prediction error methods, least squares is also prediction error method, you must have already noticed that by now, you should have. We are minimizing the prediction errors. But, as you know, there is another class of methods called MLE. Which takes a completely different stance. On the face of it, MLE looks at the probability density functions, likelihood and so on, but eventually, as I will show you today and then, maybe in the interest of time also, partly tomorrow. When we set up the MLE problem for parameter estimation, you will see the prediction error somehow make their way through. And this is what motivated Ljung to come up with a prediction error method family. So you say, all these methods that you see least squares, MLE or regularized least squares. I have not talked about regularize least squares but I'll talk about it tomorrow. The regularized least squares can also be expressed and captured in the prediction error method framework. Suppose you're filtering the data, that also can be -- pre-filtering the data, that can also be captured in the PEM framework and so on. So this PEM, there is a routine also PEM in SysID toolbox, is a very generic solver which unifies the least squares, maximum likelihood and the regularized versions of those and so on. But we are not yet there to go to the PEM, there is only one thing that we have to understand how to set up the MLE for, you know, for a parameter estimation of a typical identification -- in a typical identification problem. And then we will see that the prediction errors appear, we've already seen least squares contain prediction errors, MLE will also contain prediction errors. And therefore, we are in a position to see the universality of the prediction error minimization and then quickly study the PEM algorithm.