Parameter Identification Using Sliding Mode Control

Welcome back. In previous classes, I talked about the observer design, differentiator design, and we also saw how to implement the super twisting controller based on the super twisting observer, and in each treatment, I discussed the application of the higher order sliding mode algorithm. In this class, I am going to talk about a very, very important class of problems that is called the parameter identification problem. So, actually in several practical application, parameter is not exactly known. So, now we are trying to understand how to identify the unknown parameter of a second-order system just by the application of the super twisting algorithm. And I have already told you that the super twisting algorithm is one of the very promising algorithms and it has a lot of practical applications.

So, we are going to see one more practical application of the super twisting algorithm to design or solve a parameter estimation problem. So, for the purpose of discussion, I have already mentioned that parameter identification is possible to show in several cases where we only have output information. So, suppose that I have a second-order system. So, a second-order system is represented by $\dot{x} = f(x) + g\,u$.

And in this particular class of systems, several parameters are also involved. And suppose that if I have just information about $x$, my main concern is how to identify the parameter. So, it means that first we also have to estimate another state, which is $\dot{x}$, and after that, based on the estimates of $x$and $\dot{x}$. By using an algorithm that is actually inspired by the super twisting, we are going to show you how to estimate some kind of unknown parameter using a known algorithm. So, now it means that whenever we are talking about the parameter estimation problem, we need two different kinds of things.

One is a parametric model and another is in regressor form. So, I am going to discuss both forms and how to solve the parameter estimation problem. So, here I am first going to express the system dynamics in the form of the unknown parameter. So, that particular model is called a parametric model because the parameter is not known. And after that, using the concept of a regressor, the meaning of the regressor form is somehow a parameter as well as a regressor vector, and the regressor vector is of the known function.

So, you can see here in this particular form that $\theta$is an unknown parameter we have to estimate, and $\phi = \phi(t, x_1)$, I know. So, simultaneously, I have to estimate $x_2$and control, I know. So, using the information from all three, I am assuming that $\phi$is known once observer dynamics converge. Why is the convergence of observer dynamics required? Because I am assuming that for a second-order system, I only know one piece of information. So, in order to give a guarantee that the regressor vector is known, I have to estimate $x_2$, which is $\dot{y}$, that is, $\dot{x}_1 = x_2 = \hat{x}_2$.

So, I have to estimate this $x_2$from the dynamics of $x_1$and $x_2$, considering the parameter is not known. If we are able to somehow separate the regressor vector from the unknown

parameter, then the solution to the problem is not so difficult. And due to that reason, the regressor somehow gives us one of the very good forms such that I am able to separate the known part as well as the uncertain part. And this somehow enables us to do parameter identification. Now, let us try to see some kind of system that is modeled using a second-order differential equation, because what I am trying to show here is how to implement higher-order sliding mode control, particularly the super twisting algorithm, and I hope that you remember the form of the super twisting algorithm.

So, the algorithm looks like something like $k_1 \mid x_1 \mid^{1/2} \text{sgn}\,(x_1)$, and after that, $x_2$and $\dot{x}_2$is nothing but $\dot{x}_2 = -k_2 \text{sgn}\,(x_1)$. and here uncertainty is also allowed, and here this gain is given. And it is possible to show that if you design $k_1$and $k_2$, either with the help of the measurement curve proposed by Professor Levant or using a strict Lyapunov function proposed by Morino, it is possible to show that $x_1 = 0$and $x_2 = 0$ in some finite time $t \geq T$. So, now I am going to utilize this particular algorithm.

as some kind of observer design. And how are we basically utilizing? We are using these two as a correction term. We are going to look whenever we have this kind of problem. So, basically, whenever I am talking about the super twisting algorithm that is applicable for second-order systems, I have in this particular lecture, I am going to start with some second-order systems, and several practical systems are modeled using the second order. Obviously, if I have a third-order system or a fourth-order system, I can again increase the order of the super twisting algorithm using the differentiator structure, and then I can again apply the same philosophy.

So, let us first try to solve for the second order, then I will discuss the third order and you will be able to easily extend this to the third order. So, for a second-order system, basically, the parameter and regressor are going to appear in the acceleration. So, I am assuming $x_1$as a position. So, basically, $\dot{x}_2$is nothing but what becomes the acceleration, because $\dot{x}_1 = x_2$means $x_1$is the position, then $x_2$is the velocity, and $\dot{x}_2$becomes the acceleration. And now, I am assuming that this part is the known part and that this part $\phi$is the unknown part.

And obviously, I am assuming that this $\theta^{\top}$is the only thing that is unknown. Once an observer I will design, at that time, $x_2$is also known to us. So, at that time, I am going to assume that this part is also known to us. So, now first try to see how to design the observer for this particular system if I only have information of $y = x_1$. So, in order to design the observer, what I am going to do is copy whatever things are known to us.

So, $\dot{\hat{x}}_1 = \hat{x}_2$, and this is nothing but some kind of correction term that is in the form of $x_1$, because in order to confirm the convergence of the super twisting algorithm, you can see that the exact same information is required here and here. And due to that reason, you can see that two correction terms I am going to add in this observer design. So, where $\tilde{x}_1 = x_1 - \hat{x}_1$is nothing but the error between $x_1$and $\hat{x}_1$. So, $x_1$is the exact information; $\hat{x}_1$is the estimated information. Now, here I am assuming $f$ is known, but if you look carefully, $f$is also not exactly known because $x_2$is not known.

However, since I will estimate $x_2$from this observer design, whatever value of this and this that exactly matches means that, for that particular reason, people can assume that this is known. Here, $\theta$is basically unknown, but I am going to substitute its estimate. So, I am going to substitute some kind of estimate here, and again I am assuming this part is known. So, this is the regressor part, and again this term is not known; due to that reason, I am going to replace it with the estimate of $x_2$. Now, I am going to express everything in terms of the error dynamics.

So, this is the error's dynamics. So, without loss of generality, people are writing $x_1 = x_2$ because we know that this algorithm is going to converge in finite time. So, once convergence happens, at that time $f(x_1, x_2) = f(\hat{x}_1, \hat{x}_2)$ and $u$, $f(x_1, x_2)$, and the cap quantities are exactly the same. And due to that, there is a difference between this vector and this vector equal to 0. At that time, $x_2$is simply replaced by $\hat{x}_2$.

So, this kind of condition comes into the picture once this observer is going to converge. So, I have retained this structure for all $t \geq T$. Even if you write $f$, that does not harm our analysis because at that time actually one extra term comes into the picture, and that term is nothing but $f(t, x_1, x_2, u) - f(t, x_1, \hat{x}_2, u)$. So, this term has some initial mismatch that comes into the picture and is compensated by this particular gain, but once the observer is going to converge, at that time this whole term equals 0, and due to that reason, just for the sake of clarity, people are not writing this, and for exactly the same reason, people are replacing $x_2$by $\hat{x}_2$.

by $x_2$. So, I hope that you are able to understand this particular formulation. Now, what is our main goal? Our main goal is to design $\alpha_1$and $\alpha_2$such that both $\tilde{x}_1 = 0$and $\tilde{x}_2 = 0$in some finite time. Easily, I can solve this problem by assuming that these whole things are actually bounded. Why can one assume this is bounded? Because you can see that this is a known function. And this is somehow a kind of parameter.

So, I can assume that the variation of parameters is not so high, and for that reason, I can design some finite gain $\alpha$, and since this system, this observer system, is running inside the computer, I can also put any large $\alpha$here. In this way, I can prove the convergence. Now, once convergence is ensured, what is the second step? Now, the second step is to actually identify the unknown parameter and to determine what kind of assumption I have. The only assumption I have is that $x_1$is measurable for this particular second-order system.

In literature, you can see several algorithms that discuss how $x_1$and $x_2$are both measurable, and then people are trying to estimate this parameter. But what is the beauty of the higher-order sliding mode control is that by using only the measurement of $y = x_1$, just a pure number of measurements, you can still estimate the unknown parameter as well as the unknown state. So, somehow this is a simultaneous estimation of the state and the parameter. So, now you can see here that I am assuming that $\theta$is not known, but the

nominal model of $\theta$is known. If this nominal model is not known, what can you do? You can substitute some values depending on the practical situation.

So, practically you can understand what the behavior of $\theta$is and what kind of $\theta$is going to come here. So, based on that, we will now proceed to the parameter identification task, and our assumption is again that I am only able to measure $x_1$, and obviously, we have a designed observer. So, $x_2$is also measurable, and due to that design, I am assuming that all states are measurable. And nominal model, what is the meaning of the nominal model? I am assuming that the nominal model of the regressor is given to us. So, what is highlighted whenever we are solving a parameter estimation problem? So, first, I have to create some kind of parametric model.

So, even if the parameter is not known to us, what will I do? I will write the differential equation in terms of an unknown parameter. So, that is called the parameter model. Once the parameter model is ready, we will come for the regressor part. So, how do you create the regressor part? I am going to separate the term that contains the multiplication of known and unknown variables, or it might possibly involve the addition of known and unknown variables. So, most of the time it is possible to show that regressor form; we have something, some unknown parameter that is multiplied by some kind of known dynamics part.

And after that, I am going to design the observer because I have fewer measurements. So, I have to first estimate all states and then I have to estimate the parameter. So, once we ensure the convergence of the observer dynamics, only then can I proceed with the parameter estimation problem. And last task, I have to design some kind of algorithm; again, that algorithm will work inside the computer, and for that reason, I hope that we will obtain some kind of differential equation, and that dynamical equation will finally converge to the true parameter; that kind of algorithm I have to create. So, now, parameter identification and observer design are essential for robust control because most of the time the exact parameter is not known, particularly whenever we are working in some kind of harsh environment; at that time, the parameter is also changing, and due to that reason, the parameter estimation problem has a very big role whenever we are talking about optimal control design or some other set of control design.

So, now, in order to simplify the problem, because most of the time the parameter is constant or piecewise constant, I am assuming that $\theta(t) = \theta$. Now, I have to estimate $\theta$, and it is possible to show that I am able to estimate $\theta$by this principle. You can see this particular algorithm. So, this is nothing but a super twisting-like algorithm, because whatever vector is here and their derivative $\dot{\tilde{x}}_1$, you can see that same vector here and exactly the same vector I have here. So, by the convergence of the super twisting algorithm, I can be able to give a guarantee that $\tilde{x}_1 = 0$.

And since $\tilde{x}_1 = 0$and $\tilde{x}_2 = 0$, $\dot{\tilde{x}}_1 = 0$also equals 0. But I cannot give a guarantee that $\dot{\tilde{x}}_2 = 0$. Obviously, I can guarantee that $\tilde{x}_2 = 0$, because that is a consequence of the higher-order

sliding mode control that $\tilde{x}_1 = 0$and $\dot{\tilde{x}}_1 = 0$. So, obviously, $x_2 = 0$and $\tilde{x}_2 = 0$, but $\dot{\tilde{x}}_2 \neq 0$. But obviously, it is possible to show that the behavior is exactly here just like the first-order sliding mode control, because I have a discontinuous term, but still I am not moving. What is the meaning of that? The average value of $\dot{\tilde{x}}_2$is equal to 0.

So, how do you calculate? Again, you can pass this term, this continuous term, through a low-pass filter, and whatever average of this is exactly equal to whatever is unknown here. Due to that reason, here I have written during sliding, during sliding

$$\tilde{x}_1 = 0, \dot{\tilde{x}}_1 = 0, \tilde{x}_2 = 0.$$

Now, the average value of $\dot{\tilde{x}}_2$is also equal to 0. So, this finally lands to this particular condition. Now, here you can see this part easily; you can measure it using the low-pass filter.

This $\phi$is known to you. So, now, I have to show that if $\theta - \hat{\theta}$is tending towards some known value, and obviously, I know $\hat{\theta}$, and due to that, it is an unknown parameter that I am going to estimate. $\hat{\theta}$knowing means any nominal value I will keep here. And once the algorithm is converged, it means that if I define this as $\theta - \hat{\theta}$and some $\hat{\theta}$is known, and suppose that $\theta$is the estimate of this algorithm, I will create such that this will converge to this in some finite time. So, it is possible to show now that

$$\theta = \Delta\theta + \hat{\theta}.$$

So, in this way I can estimate the parameter.

So, due to the consequence of the second-order sliding mode control, by designing a low-pass filter, it is possible to show that I am able to estimate the unknown parameter. And this is called equivalent output injection, because here we are in higher-order sliding mode control, but still the average value of $\dot{x}_2$. So, somehow you can treat the first-order sliding mode across $\dot{x}_2$. So, that kind of interpretation we have, and due to that reason, I am able to do equivalent output injection here to estimate the parameter.

So, now I have to create some kind of algorithm such that I can guarantee that $\Delta = \widehat{\Delta\theta}$. So, that kind of guarantee I have to provide. So, we have two different approaches. One is a discrete approach, and most of the time people are actually applying the least squares algorithm. What is the meaning of the least squares algorithm? I will create an error between these two, and after that, I will minimize that error.

Another approach that is the continuous version of the above approach is the discrete approach, and since this system is continuous, I am going to create some kind of continuous algorithm. So, let us come back here. So now I am going to create some kind of continuous algorithm. So, how do you create a continuous algorithm? So, I am going to define

$$\Delta\theta = \theta - \bar{\theta}.$$

So, this whole term I am going to define as $\Delta\theta$.

So, for the sake of simplicity, you can also define $\theta - \hat{\theta}^{\top}$here. In this way, I can write this particular expression. Now, you can see here that after doing this, I am going to take the average. So, how do you calculate the average? So, in order to take the average, I am going to take the equivalent value, $z_{\text{eq}}$, that equivalent value of control, $\phi$, and this regressor vector which is known to us, that I am going to multiply here. And after that, on the right-hand side, if you see in the previous slide, I have $\phi$.

And after that, I am going to multiply by $\phi^{\top}$. So, that comes into the picture, and after that, I am going to integrate from 0 to $t$. So, finally, it is now possible to show that this particular estimate of the parameter is nothing but the result of substituting the equivalent value. So, I will obviously not get the exact value, and for that reason, I will modify $\Delta\theta$by $\widehat{\Delta\theta}$.

is here, and $\frac{1}{t}$is here. So, we can assume that it cancels out, and after that, I will get this expression. Now, I have to express this parameter estimation algorithm in the form of the dynamical equation. So, how do you do that? So, for that, I am going to apply some kind of identity, and this is called the matrix inversion property. So, what is the theory suggesting? Suppose you have some kind of square matrix and that matrix is invertible. So, it is always possible to show that a square matrix inverse multiplied by the same matrix equals the identity, that is,

$$A^{-1}A = I.$$

So, the exact same property I am going to utilize, even if this is a time-varying matrix, will still preserve this identity provided that the matrix is invertible. So, I have to guarantee that whenever we are creating an algorithm, any square matrix I consider should be invertible for all $t$. So, giving a guarantee, so this is $\sum_t$ , this matrix is somehow the square matrix. So, giving a guarantee that this is invertible for all $t$is not an easy task. So, for that again we have to look into how to give a guarantee.

So, once you assume this, it is possible to show now, if I take the derivative on both sides. So, here this term is multiplied by the derivative of this, again the derivative of the inverse, and after that, the same term is equal to 0. So, this property of the derivative states that the identity is constant, and for that reason, the derivative is equal to 0. Now, I am going to utilize this kind of thing by defining some terms here, like this term as the summation, and then we will proceed. So, the summation I have defined is like

$$\left(\int_0^t \phi\,(\tau)\,\phi^{\top}(\tau)\,d\tau\right)^{-1}.$$

Now, I am going to apply the matrix inversion property. So, regarding the matrix inversion property, if you come back here, you can see that

$$\dot{\Sigma} = -\Sigma \dot{M} \Sigma,$$

where this is the inverse. So, the summation Σwill come here, and after that, the derivative of this term, the derivative of this term, and how we have basically defined this, like this. So, I have defined this as an inverse; I will then put this kind of term into the picture; the bracketed term comes into the picture. If you take the derivative of this particular integral, it means that these two terms come into the picture. So, in this way, I can express the regressor vector in terms of some kind of known square matrix.

Now, what am I going to do? I am going to take the derivative of this again. So, I am going to take the derivative here since I know the parameter estimation algorithm is given by this particular integral equation. So, if you take the derivative, then the derivative of the first term is exactly this term, which I have already defined as $\Sigma(t)$; after that, I have to take the derivative and preserve this term. So, in exactly the same way, I have written here. Now, you can see here that I also know that $\dot{\Sigma}(t)$, in the previous slide, I have already established is equal to this.

So, this further simplifies the $\Delta\theta$estimate, equal to like this; easily, you can see. Now, here since Σis known as some kind of matrix which is time varying, $\phi(t)$, that is the regressor vector; $z_{\text{eq}}$can be substituted from the low-pass filter. Now, here you can see that $\phi$is also known. So, the whole algorithm is just expressed in terms of the unknown.

And what is unknown? That is $\Delta\theta$. So, if this algorithm runs inside the computer, after a finite time, once $z_{\text{eq}}$comes into the picture, it is possible to show that $\Delta\theta$will converge to some known value. Now, I have to prove the convergence. So, two things I have to prove that I have to show are that in this particular algorithm, whatever summation is always invertible. So, after that, this algorithm is going to converge.

So, I also have to give a guarantee of convergence. If that diverges, then I am never able to estimate the parameter. So, now how do we guarantee these two things? What am I going to do? Since I know that the real parameter factor is given by this. So, basically, I have copied this equation from here; you can see this equation. So, I have copied the exact same equation.

So, here I have this equation. After that, what I am going to do, since I know the known parameter, known parameter is Δ, means known parameter is $\widehat{\Delta\theta}$, and after that this is somehow an estimated parameter; I have $\widehat{\Delta\theta}$and the known parameter is $\Delta\theta$. So, this is the original parameter and this is the estimated parameter. So, now, I have to prove that

$\Delta\theta$tending towards the real parameter means that whatever equivalent value, obviously, whenever we are calculating the equivalent, there is always some mismatch because low-pass design is not a simple task, and due to that reason, I am going to add some kind of correction term. So, now, once I add this correction term and substitute it inside this algorithm, then basically it is possible to show I can write like this. Why does this term come into the picture? Because I have already assumed this particular inverse to be equal to the summation.

So, now real parameter values are expressed in terms of the equivalent output injection and noise. So, this part is nothing but noise. I have already told you that whenever you apply some kind of low-pass filter, some noise will always come into the picture. So, I have to show that in the presence of noise, I can also get the real estimate.

I have two parts here; one is the noise part. So, this part's equivalent value is actually exactly equal to the estimate of $\widehat{\Delta\theta}\ \phi(t)$. This means that if I exactly know this, then I can exactly know what the equivalent control is. Now, what am I going to do? I am going to start with the previous expression, and after that, I am going to substitute this value. So, once you substitute this value, you will get this kind of expression. So, once you get this kind of expression, and since I know that this is given like this, finally, I will get this kind of expression.

So, now here you can see that you are able to see the relation between $\Delta\theta$and $\widehat{\Delta\theta}$. So, somehow I have to show that in the absence of error, both the estimated value of $\theta$and the true value are exactly equal. So, how do we show? So, for that, I have to give a guarantee that this term, this is nothing but the error term, is going to 0. So, now for that, I need some kind of condition, a convergence condition that is called persistence of excitation.

Now, this will tend to 0provided I am able to ensure this is greater than 0. And obviously, I also have to ensure that the summation inverse $t$is always non-singular. So, if the noise effect is vanished, then obviously, I can be able to give a guarantee that $\Delta\theta = \widehat{\Delta\theta}$. It means that the error between $\Delta\theta$and $\widehat{\Delta\theta}$is equal to 0. So, I have to guarantee that this is going to converge to 0as $t \to \infty$. It means that our parameter estimation task starts at infinite time, but that it is going to finish after a very long time.

You can see here, because I am going to prove convergence as $t \to \infty$. So, in order to actually guarantee that this is always non-singular, what am I going to do? I am going to add some kind of part that is $\rho$; $\rho$is a constant, and $I$is the identity matrix, and the identity matrix is always invertible. So, even if this is 0, I can be able to give a guarantee that whatever summation inverse that is always possible. The summation inverse always comes into the picture. It means that whenever we are initializing the algorithm, I have to initialize it like this.

So, because here in $\Sigma^{-1}(t)$, and due to that reason you can see that $\Sigma(0) = \rho I$. So, let us

again come back here; then you can easily correlate. Actually, in order to solve the parameter estimation problem, I have to run two algorithms together. So, I have to actually run this matrix differential equation on the computer as well as this unknown parameter estimation algorithm on the computer.

So, now I have to give the initial condition for both. So, how do we set the initial condition of $\Sigma(t)$such that $\Sigma^{-1}(t)$will exist? So, for that, I have to set the initial condition

$\Sigma(0) = \rho I$, so that $\Sigma^{-1}(0) = \rho^{-1} I$. So, $\rho > 0$times $I$. So, those kinds of things I am going to select here. So, that will maintain $\Sigma^{-1}(t)$always possible.

And after that, obviously, I can $\Delta\theta$; I have some kind of estimate. So, those kinds of things I will run. And after that, automatically due to this property, it will converge to the true value, because if this is very, very small, then obviously, I am going to converge to the true value. So, this is somehow the initialization criteria, and obviously, I ensure

$$\sup_t \ \Sigma(t) \leq 0.$$

So, numerically, whenever we are actually implementing inside the computer, almost all computers are working on digital principles.

So, this will give the numerical stability. So, here I have taken one problem for your homework. So, I am going to take the second-order system again, and I am assuming here that

$$u(t) = \sin(t),$$

but here you are free to select some kind of stabilizing control. So, you can be free to select some kind of algorithm

$$u = k_1 \text{sgn}(x_1) - k_2 \text{sgn}(x_2),$$

and here obviously, that becomes $\hat{x}_2$. So, that kind of algorithm you can implement here. After that, you can select the original parameters $\theta_1$and $\theta_2$; the first job you can do is to take the exact values of $\theta_1$and $\theta_2$.

After that, you can design a higher-order sliding mode observer, take these two initial conditions, and show that whatever observer dynamics will converge. Once you solve this particular problem, I have already told you how to initialize. Here, what I am going to do is directly write the algorithm in this way. So, what kind of assumption am I assuming here? I am assuming that the initial parameter is equal to 0. Sometimes this will create some kind of

difficulty, but if you are lucky enough, then this algorithm will converge.

Otherwise, you have to actually write everything in the form of $\widehat{\Delta\theta}$. And what is $\widehat{\Delta\theta}$? It means that somehow you are writing $-(\theta - \hat{\theta})$. An estimate of this is something you have to consider. You have to take it, but if you substitute $\theta = 0$, then obviously, the algorithm looks like this, exactly like this. In this way, you can estimate the parameter; parameter estimation you can obtain. And after that, now you have to actually ensure the persistence of excitation condition, it means that $\Sigma^{-1}(t)$remains non-singular.

So, how do we do that? So, for that, I have selected some kind of regressor matrix. So, $x_1 = \sin\ t$. Now, you can take different values of $\rho$and show that this summation inverse always basically becomes non-singular, meaning the inverse always comes into the picture, and somehow this creates some kind of tradeoff between convergence speed and estimation accuracy. So, in the simulation, please do this, and the algorithm you can easily develop here; I have substituted $\sin t$, but if you want to apply higher sliding mode control, you can use the twisting controller, and you can easily show that the error converges within 2 seconds. After that, you can simulate this algorithm, and it is possible to show that this algorithm will converge in 5 seconds or 7 seconds.

It might be possible that this time will vary based on what kind of ODE you are going to use. And after that regression analysis, you can see that if you set $x_1 = \sin\ t$and $x_2 = \cos\ t$, then the maximum eigenvalue is always given by some kind of linear growth. So, that is always invertible, and the persistence of excitation condition is also satisfied. Convergence you can see, and it is possible to show that if you take $\rho = 1$, then system performance becomes. So, now, it is time to conclude this lecture.

So, what have we seen? In this lecture, we solved two problems simultaneously. And most of the time, this is easily possible with the help of higher order sliding mode control. So, we have solved the estimation problem as well as the observation problem. So, I have estimated the parameter, and I am also estimating the state. Obviously, first, I have to converge to the true parameter and the true state, and then I will be able to create some kind of algorithm such that I can converge to the true parameter. During this process, I have to maintain this square matrix that is non-singular throughout the time interval, and due to that reason, some criteria come into the picture that is called the persistence of excitation criteria.

And I have to minimize the noise. And obviously, these algorithms are used because the solution of the matrix equation is very, very difficult, and for that reason, most of the time, these algorithms run inside the computer, and due to that reason, you have to give a numerical guarantee that this algorithm is going to converge. So, if you are able to select everything very properly, it is possible to show that the algorithm can help you get the true parameter. During the simulation, if you are not able to obtain the true parameters, it means that you have to reinitialize; it indicates that numerical instability has come into the picture. So, with this particular remark, I am going to conclude this lecture, because in this lecture,

we have discussed dynamic estimation.

Dynamic estimation means I have information about only one state. So, I have created a second state and am using that for dynamic estimation of the state and estimating the parameter. Obviously, I have talked about the persistence of excitation and regularization conditions so that I am able to achieve robust parameter identification, meaning parameter identification in the case of uncertain information. So, with this remark, I am going to end this lecture. Thank you very much.