First Order Differentiator & Output Feedback Control Based on Sliding Mode

Welcome back. In the previous class, I talked about the differentiator design, and we saw the first-order differentiator that is basically based on Utkin's first-order sliding mode control. We have also explored the high-gain-based differentiator. In this lecture, I am going to extend the concept of differentiation based on the higher-order sliding mode control. And why are we basically looking for a differentiator based on higher-order sliding mode control? Because of the accuracy of differentiation in the presence of noise, it is possible to show that it is very, very high. And somehow, that differentiation process is robust and exact.

So, the main theme of this lecture is to talk about the robust exact differentiator developed by Professor Levant. So, for the purpose of discussion, I am going to talk about the first-order differentiator. It means that suppose I have some kind of signal that is $y(t)$. So, now I am going to develop some kind of algorithm such that if I substitute $y(t)$, then I will get $\dot{y}(t)$.

In this way, suppose I have a second-order system; then I can design control based on the output and its derivative. Another important thing is that most of the time this particular output is measured with the help of sensors, and we are already well aware that sensors contain several noises. So, how do you get the differentiation of the noisy signal? Obviously, most of the time I have high-frequency noises. So, high-frequency noises signal that I can get some kind of exact differentiation. So, it means that the differential process I have to actually finish in the presence of some kind of measurement noise; we know the characterization of the noise, and after that, we are trying to develop some kind of differential process based on the higher sliding mode control.

And since I am just interested in calculating the first-order derivative, this is called the first-order exact differentiation process. Why exactly? Because this differentiator will work in the presence of the measurement noise as well. And we can actually get the true derivative in finite time if I use this kind of concept. And obviously, in the presence of some kind of measurement noise, whatever differential results we obtain, it is possible to show that we will achieve accuracy. And why are we basically doing this? Because I want to design some kind of output feedback-based control.

So, I have second-order uncertain system. So, I have some kind of black box system. What is the meaning of black box? I just know the order of the system, but I do not know the dynamics completely. So, suppose that I have some kind of input, and I am measuring the output here, and now whatever $u(t)$ I am going to design is a function of $y(t)$ and $\dot{y}(t)$. So, that kind of things we will see, and it is possible to show that the differentiation process can finish in finite time.

So, I can able to design finite time controller based on this. So, first I am going to tell you the problem statement because most of the time we have seen that in control systems whenever we have output. So, basically, we are measuring output with the help of a sensor. And that is corrupted by some classes of noise. So, I am going to define some classes of noise that commonly occur in the system called bounded Lebesgue measurable noise.

I will define what the meaning of bounded and Lebesgue measurable is. And what we are assuming is that whatever signal I am going to develop, some kind of algorithm will be created such that if I apply $f_0(t)$, then I will get $\dot{f}_0(t)$. So, that will satisfy some kind of assumption. And one of the common assumptions is the Lipschitz assumption. It means that, suppose I have to calculate the first derivative of $f_0(t)$, I am assuming the first derivative is continuous.

I am also assuming that the second derivative is continuous and bounded by some kind of Lipschitz constant L; that is the kind of assumption I have. And what is our main intention? That if this signal f_0 contains some kind of bounded Lebesgue measurable noise, I will define the meaning of Lebesgue measurable. Then I have to get the differentiation. And due to that reason, what is the requirement? The differentiation process should finish in finite time, and it can handle some kinds of non-smooth signals. So, let us first characterize the meaning of boundedness.

So, the meaning of boundedness here is that whatever noise I am going to consider with this signal has finite amplitude. So, suppose that I have $f_0$and some quantity like n αsin (ωt). Whatever αI am assuming should be bounded and Lebesgue measurable, which means the noise can be integrated over any time interval. So, if you consider any time interval here, suppose that the interval is from $t_1$to t_2. So, I am assuming that whatever noise is here should be integrable.

So, that kind of noise is going to enter the system, or whatever differentiator I am going to develop is only applicable for those classes of noise. This is the definition of boundedness. So, here it means that there exists some constant $M > 0$such that

$$| n(t) | \leq M \quad \text{for all } t.$$

And after that, why this assumption is required is because somehow whatever process I am going to develop, after differentiation, I am going to utilize that as a control design. So, actually, the differentiator as well as our closed-loop system is running in some kind of closed loop. Obviously, the differentiator is running inside the computer, and somehow from the output of the computer, we are getting the derivative of the output, and after that, we are applying it to design the control.

Due to that reason, we have to make sure that everything remains bounded. So, using this condition, I can able to prevent our system to destabilize. And this is just a simple example. Now, I have already told you that whenever we have a system, it is generally represented in the form of a differential equation like this, and suppose I have some noise here, for example. To obtain the information of a state evolution, I have to integrate the right-hand side of the differential equation.

So, most of the time, if something is integrable, then it is possible to show that there always exists a unique solution, or at least I can guarantee existence, and for that reason, for a broader perspective, I am assuming that whatever noise is going to enter the signal will satisfy this kind of property. And obviously, if Lebesgue measurability is present, then the theoretical proof of control, once coupled with a differentiator, is easy; and for that reason, basically, whenever we consider noise, we are addressing these two conditions: one for stability and another to provide a mathematical guarantee in the presence of some kind of noise. And obviously, in a realistic world, I have sensors and environmental disturbances, and it is possible to show that these kinds of disturbances will satisfy all the above properties, and you can just consider some kind of noise that is periodic in nature. So, you can represent that noise in the form of constant sine and cosine, and it is possible to show that it is going to satisfy these two assumptions easily. So now let us come to the original problem statement.

So, I have original signal $f_0(t)$, but whatever signal that is coming to the system, that is corrupted by some kind of noise $\eta(t)$, and again I am assuming that $\eta(t)$is bounded and Lebesgue measurable. Now, I have to get $\dot{f}_0(t)$, but originally whatever signal I am applying is $f(t)$. So, the main intention is that even if I apply $f(t)$ to the system, I need to know how to get the derivative $\dot{f}(t)$with some kind of accuracy.

Obviously, I cannot exactly get it; you cannot develop some kind of algorithm. You can differentiate in the presence of some kind of noise, but with a certain level of accuracy, you can do this job. So, in order to do that, what am I going to do? I am going to consider some kind of a system. Suppose that $z_0 = v$, and what is the meaning of z_0? Suppose that I have to differentiate some signal like $f_0(t)$ because this is somehow the original signal. So, first I will develop a theory for some kind of system.

where there is no uncertainty or disturbance, and after that I will actually check the performance of the algorithm in the presence of noise. Due to that reason, I am assuming that I have to differentiate this signal $f_0(t)$, and after that, the differentiation of this particular $\dot{z}_0$is nothing but $\dot{f}_0(t)$that I have to obtain. So, for what I am going to do, you can see here that I have developed some kind of first-order system $z_0 = v$; vI am going to define, and $\eta_0$is nothing but the difference between $z_0$and f_0. Now, I have to maintain second-order sliding across η_0. What is the meaning of second-order sliding mode control? It means that $\eta_0 = 0$, as well as $\dot{\eta}_0 = 0$in some finite time $t = T$.

That is the meaning of second-order sliding mode control. Obviously, in this process, I have to design this signal $v$that somehow follows the property of the Filippov solution. So, now, what am I going to do? I am going to assume
$z_0 = f_0$;
$\dot{f}_0 = v$.

What is f_0? That is also equal to $\dot{z}_0$. Now, what am I going to do? I am going to take the derivative of η_0.

So, $\dot{\eta}_0 = \dot{z}_0 - \dot{f}_0$; I am assuming that whatever signal I am going to differentiate is differentiable, and after that, $z_0$is nothing but v, and I have kept this assumption. In this way, this differentiator is somehow different from Professor Utkin's differentiator that is based on first-order sliding mode control. It is possible to show in the case of Professor Utkin's differentiator that the assumption is that it should be bounded by some constant M. But in order to apply higher-order sliding mode control, you should know that the second derivative is also bounded.

It means that if the signal has just the first derivative, you cannot be able to apply this kind of algorithm; that is the limitation. Obviously, whatever algorithm is developed in a control system will always work within certain limitations. I am going to define a way such that I can enforce sliding mode across η_0, which is our main goal. And due to that, now you can see that $\dot{z}_0$, which is v, because in this system, now I know, I have to make sure $\eta_0 = 0$and $\dot{\eta}_0 = 0$. So, the same kind of term I am going to keep here.

Now, if you look carefully, you can see this is nothing but some kind of super twisting–like algorithm. That is obviously not exactly a super twisting algorithm, because in a super twisting algorithm, whatever term is here, the same term you have to repeat here as well as here. But here, if you look carefully, suppose that I define some kind of term, which I have defined here as
η_0.

So, $\eta_0$is nothing but
$\eta_0 = z_0 - f(t)$.

And if I calculate the first derivative of η_0, then that becomes
$\dot{\eta}_0 = \dot{z}_0 - \dot{f}(t)$,

and since
$\dot{z}_0 = v$,

this gives
$\dot{\eta}_0 = v - \dot{f}(t)$.

If you substitute from here, then finally, it is not difficult to see that this algorithm is exactly converted into some kind of algorithm that looks like the super twisting algorithm η_0, and after that, I have $+ z_1$. And obviously, one term, more terms will come here that is $f(t)$, and the second term is
$z_1 = -\lambda \operatorname{sign}(\eta_0)$,
and here I have η_0. Now, one more transformation you have to make. Why one more transformation? Because in super twisting algorithm, first part we do not have any disturbance. So, suppose that you write

$z_1' = z_1 - f_0$.

So, z_1' will come here, and z_1', that is equal to here, is nothing but $\dot{z}_1$, which is $-\lambda_0$ sign(η_0), and plus, obviously, here is a negative sign. So, after that, I have the double derivative of $f_0(t)$, and due to that reason, in order to ensure the convergence of η_0 and z_1',
I have to maintain a bound on $\ddot{f}_0$. If we are able to do that, then it is possible to show that

$$\eta_0 = 0$$

and $\dot{\eta}_0 = 0$. And obviously, whenever $\eta_0 = 0$, then $z = \dot{f}_0(t)$. And what is the meaning of $\dot{\eta}_0$?
If $\eta_0 = 0$, then it is not difficult to see that $f_0 = z_0$.

So, in this way I can able to prove the convergence of algorithm. So, now what is our main goal? Exactly the same as the super twisting algorithm, I have to design lambda1 and lambda0. And how do you design? You can either apply the theory of Professor Levant based on a measurement curve, or you can apply some kind of theory given by Professor Levant based on a strict Lyapunov function. Here, I am going to give the theory that was developed by Levant, because it is based on homogeneity and contractivity.

And things are very easy. We just have to maintain homogeneity everywhere, then things are very easy to follow. So, it is possible to show that in the absence of noise, I am assuming that whatever signal I am going to apply here, $f(t)$, is noise-free. So, that is exact signal $f_0(t)$. But in that particular case, it is possible to show that I can, if I am able to design gain like this, then what kind of things happen, that I can prove that basically
$z_0 = f_0(t)$.

One more important thing, whenever I am doing this kind of manipulation, I am assuming that $f(t)$contains $f_0(t)$.

But obviously, whenever I have an error, one more error term will come here, and that error is finally transferred here. Again, since I am assuming that the error is bounded and Lebesgue measurable, some condition will appear. And due to that reason, it is not going to affect any other analysis. So, the main thing is that you can see that the double bound

$$|\ \ddot{f}_0(t)\ | \leq L.$$

So, that constant you can select to design λ_0, which is the gain associated with the second term that is going to compensate for the uncertainty, and after that, you will be able to select λbased on this particular expression.

Then it is possible to show that you are able to get the derivative. So, in a computer system, you have to actually run this particular algorithm, keep this signal, and then measure the derivative. So, z_1, you can just measure. If you are going to measure, then you will get the derivative of the signal.

I hope that you can understand my point. So, what is highlight since super twisting controller avoid the chattering of first order system. So, it means that I can also able to utilize this kind of concept exactly same kind of concept for the differentiator design. So, you can see that during this construction I have started with a first-order system. and then we have proceeded towards the differential process. The proposed differentiator handles some kind of non-smooth signal and measurement noise.

Finite time is guaranteed because this is somehow a super-twisting algorithm. And obviously, the relative degree of the differentiator is 2. So, using this kind of differentiator, whatever I have proposed, this is the differentiator that is only applicable for the calculation of the first-order derivative. Now, this is very, very important to show that in the presence of the noise, our differentiator accuracy is good enough. That kind of thing I have to establish, because we know that the same kind of job I can do using Utkin's differentiator or high-gain differentiator.

But why are we actually making some extra assumptions? Because here in this process, you might have seen that I have to make one more extra assumption that

$$| \ddot{f}_0(t) | \leq L.$$

It means that not only the derivative of the signal, but their double derivative also should be bounded. Then I can only propose some kind of super twisting–like differentiator. So, it is possible to show that if

$$| z_0 - f_0 | \leq \text{constant}.$$

Why am I telling about this? You can see that in the original differentiator, since you are going to apply some kind of signal; so what happens? That whenever you are generating some kind of differentiator, suppose that this is a computer system and this is your plant.

So, whatever sensor information is coming here, you are going to keep it directly inside the differentiator. So, this $f(t)$signal is a corrupted signal, and for that reason, whenever you measure $z - f_0$, this signal somehow comes with some kind of accuracy. Now, during the differential process, it is possible to show that if.

… That will follow $\mu_1\ \varepsilon$; then, after the differential, that will become $\mu_2\ \varepsilon^{1/2}$. This is just due to homogeneity. We have already seen that whenever I have a super twisting algorithm and if the sampling interval is τ, then the accuracy of $x_1$is actually bounded by τ^2. What is the accuracy of x_2? $\dot{x}_1$that is proportional to $\mathcal{O}(\tau)$.

In big O notation, you can write it like this. And if $x_1$contains some kind of noise, at that time, accuracy is given by $x_1$and, obviously, some kind of big O notation $\mathcal{O}(\varepsilon)$and $\dot{x}_1$that is $\mathcal{O}(\varepsilon^{1/2})$. So, the exact same kind of consistency is also here. Now, how do we prove it? So, the proof is not difficult. What can you do? You can create η_0, and $\eta_1$is nothing but $z_1 - \dot{f}_0$. You can define it, and after that, you can construct this kind of differential inclusion.

Why does differential inclusion come into the picture? Because here I somehow have the

$$\lambda_0 \text{sign}\ (\eta)$$

and the $\ddot{f}$, $\ddot{f}_0$ term will appear here. So, I know that $\ddot{f}_0(t)$ is bounded by L, and due to that, it is $\lambda_0 - L$ and $\lambda_0 + L$. So, I have infinitely many differential equations here. So, this is nothing but some kind of Filippov regularization that I have used. And now, what am I going to do? I am going to apply some kind of coordinate transformation.

So now, I am also going to include some kind of noise. So, how do we include the noise? Because if f_0 is not exact, then obviously η_0 contains some kind of noise that will lie between $-\varepsilon$ and $+\varepsilon$. So, everywhere I am going to substitute that. Now, whenever I have this kind of differential equation, obviously I cannot converge exactly to $\eta = 0$and $\dot{\eta} = 0$in finite time $t \geq T$. So, what we have observed is that whenever we have some kind of noise, then obviously, I can converge in the vicinity of η_0 and η_1(or ζ_1). That is the property of homogeneous equations that, in the presence of some kind of noise, if you look carefully, the noise is inside this particular variable.

So, that is not going to destabilize our system. So, a homogeneous differential equation is somehow robust in the presence of noise. So, same kind of theory we have applied here, and after that we have calculated this bound. Now, I can easily get the exact bound. How do I get it? Since I know that this particular differential equation is homogeneous. So, how do you maintain homogeneity? Provided this η_0 is actually of the same degree of homogeneity as ζ_0, which contains the same kind of homogeneity as η_0, and for that reason, I have used the same kind of scaling factor here. It means that somehow the magnitude of η can be represented by $\alpha^2\zeta$.

And in this way, I can get the αin the presence of the $\eta_0$that is known to us and the actual value of η. And finally, I will get this kind of bound $\eta$and η_1. So, in this way, I can also see that $\eta_0$is proportional to ϵ, because whatever error I have here is ϵ, and this is proportional to $\sqrt{\epsilon}$. So, using this simple transformation, I can talk about the accuracy of this particular differential process. So, what is the implication of this transformation? Since due to homogeneity, the system is going to preserve its structure.

What is the meaning of perseverance? It means that if I scale with this, then our differential equation remains the same. So, by scaling this, I can come very, very close to the origin by making αvery, very small. And due to that reason, I never become unbounded. Now, $\eta_0$and $\eta_1$are scaled by $\zeta$and $\sqrt{\zeta}$. And in this way, I am able to talk about the stability and boundedness in the presence of the noise.

So, homogeneous transformation gives us systematic way what kind of response we will see in presence of the noise. Now, another important factor is how to design gains, because signal is going to vary, but algorithm remains the same. It is possible to show that if you obtain one set of gains by simulation, you can easily scale that gain by some kind of parameter $\ell \geq 0$, because everything is homogeneous. So, just as you can also scale the gain, all other properties remain the same.

In this way, accuracy is basically going to be affected. So, in order to maintain accuracy you have to now select this Lvery carefully that is the meaning. And again, proof is not very difficult. What can you do? Suppose you have some signal f; you can divide that by L, initially $L = 1$. So, for that I have designed 2gain, because 2gains are required for the convergence of differentiator.

Now I am going to scale it out. And after scaling, it is possible to show that I have a homogeneous differential equation again. And how do you get a homogeneous differential equation? Because here whatever λ_1, since $\tilde{z}_0$and $\tilde{f}$are scaled by this, and due to that reason, this particular factor comes into the picture. Then we can be sure that this equation, when $L = 1$, is exactly the same as when $L > 0$. In this way, I can design a different set of gains by considering just one set of gains. And one of the valid sets of gains we have already seen is given by Professor Levant using the measurement curve method, and somehow I have repeated several times that this gain is very, very optimal.

The second set of gain, also provided by Professor Utkin, will give very optimal convergence; therefore, whenever you are doing some kind of practical implementation of a system we are designing based on a differentiator, it is better to utilize this kind of gain that is actually given by Professor Utkin and Professor Levant. So, v becomes noisy, z_1 remains Lipschitz, but a small phase delay is always present. So, delays of this kind are also compensated because these kinds of homogeneous differential equations are robust with respect to delay. We have already seen that. Now, in order to show the performance comparison, what am I going to do? I have taken some kind of signal; this signal does not contain any noise, and then we have developed some kind of high-gain based differentiator.

So, I have designed some kind of low pass filter, and after that, you can see here that I have added some noise, and I am going to increase the frequency of the noise. Then we are trying to check the actual accuracy of the proposed differentiator. So, if there is no noise, you can see that both the algorithm is performing approximately same. So, this is the purpose differentiator, and this is actually the linear one, and this is the true one.

Now, what am I going to do? I am going to increase the noise. And in the presence of noise, you can see the behavior now. Now, further you can increase.

Now, as the frequency is going to increase, things become more... more prominent. It means that the differential process you are going to lose if you have very, very high switching frequency noise, and theoretically, one can easily see that during the differential process, 30 is going to multiply here, and due to that reason, what happens is that if your differentiator.

.. If it is not robust, then that kind of effect will come into the picture. And most of the time we have a high switching frequency; it might be possible that this is 500, 600, 1000, or something like that. Due to that reason, it is better to utilize the robust exact differentiator that is based on the super twisting algorithm and the gain set we have also provided. And obviously, the theorem established the finite time convergence and accuracy because one of the main intentions is to show accuracy in the presence of the bounded level measurable noise. L will maintain the performance. So, if you can correctly identify L, then your performance on your differential algorithm is very, very high.

Practically, practical tuning of this differentiator is also very easy because most of the time, whenever some engineers are working, they are not worrying about the theory. They just worry about the gain, which set of gains I will use so that I will get a robust, accurate, and suitable real-world application. So, for that we have already discussed how to select the gain. Now, the next part of this lecture is basically based on how to design a controller.

So, I have second order system. So, relative degree 2, what is the meaning of relative degree 2? You can take the output equal to σ, and you can calculate the second derivative $\ddot{\sigma}$; then control is going to explicitly appear. Here, I am assuming that both $A$and $B$are not exactly known. Now, what am I going to do again? I am going to take the information from the output σ. This whole algorithm is going to run on the computer. From the computer, I will now get information on the σI am giving, and I am getting information on $\dot{\sigma}$.

So, $\dot{\sigma}$is nothing but what we have already seen is equal to z_1. So, $\sigma$and z_1. So, here $\dot{\sigma} = z_1$. So, now whatever z_1I have to take out from the computer system that will give the derivative of σ. I am going to utilize that information to design robust control. What is beauty? This control, since now whatever information we are going to utilize for the control design, so accuracy will also satisfy. And due to that reason, it is possible to show that whenever you apply two sliding mode controls that are based on a differentiator, exact stabilization you are going to get, and finite time convergence you are also able to achieve in the absence of noise.

In the presence of noise, you can see that your trajectory remains very close to the origin, with $\dot{\sigma} = 0$. Suppose that I am solving just a stabilization problem because $\varepsilon$is very, very small. So, I have taken this practical system $c\sigma = x_1$, which is the output K_m, and the maximum and minimum values of the control through which control is going to enter that I have selected. I have designed this kind of differentiator-based output feedback.

So, actually, you have to somehow ensure dynamic output feedback is needed. So, $z_1$is nothing but the information that is coming from x. And what is z_1? $z_1$is nothing but the output of this particular differentiator that is running inside the computer. And in this way, I can get the dynamic output feedback. Why is it dynamic? Because you can see here, $z_1$is going to satisfy some kind of differential equation or differential inclusion.

In this way, you can also be able to calculate the precision of the trajectory. For a steady state accuracy of that trajectory and the initial condition we have selected, it is possible to show that all states are going to converge in finite time. So, in this way, just using the output and by taking one kind of differentiator, even if the dynamics are not known, which means the dynamics are a black box, I can design a controller for a second-order uncertain system. And obviously, whatever performance I have used, that performance is much higher because the differentiator I have used is robust and exact. So, two sliding mode controls with differentiators achieve, obviously, finite time stabilization in the presence of very, very small noise, meaning bounded or labeled noise, and after that, the differentiator is also characterized in terms of epsilon, and in the simulation, we have already seen that performance is extremely good in the presence of the disturbance.

So, in this way, I can utilize this kind of concept for any second-order system. Okay, with this remark, I am going to conclude this class. Thank you very much.