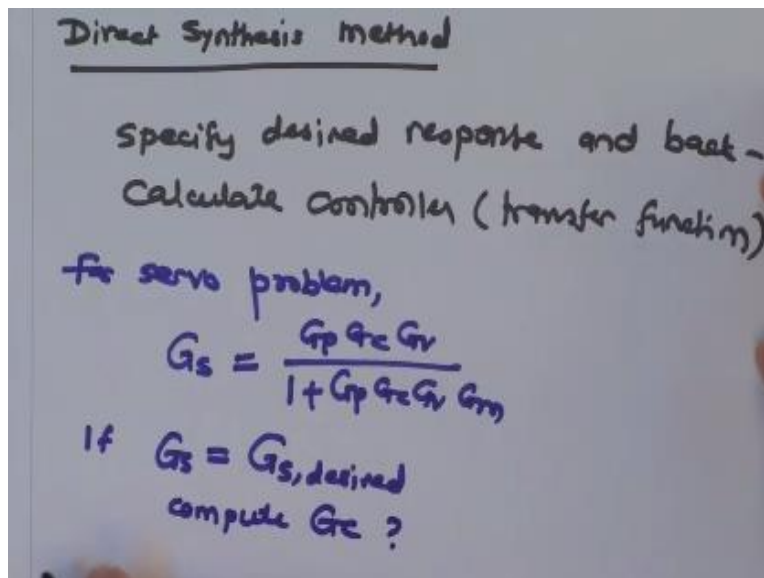


Chemical Process Control
Prof. Sujit S. Jogwar
Department of Chemical Engineering
Indian Institute of Technology – Bombay

Lecture - 36
Direct Synthesis-Based Controller Tuning

Hello students. We have been looking at methods to tune PID controllers. In this lecture, we will look at a method known as direct synthesis method. It belongs to a class of methods known as model-based controllers wherein we will use the information about process model in order to come up with controller parameters.



As the name of the method suggest it is a direct synthesis method, what we mean by that is here we specify desired response and back calculate controller transfer function. If you want to distinguish this method with the previous method of heuristic tuning or criteria based tuning, in both those methods what we do is we essentially select certain controller parameters so that they will give a certain property of the response. Here what we are doing is we actually specify the response which obviously can satisfy some of the performance criteria and then accordingly we back calculate what would be the controller which will give you that particular closed-loop response.

Let us say for an example, if we are talking about for servo control or servo problem, we know that the transfer function between output and manipulated output and a set point is given by,

$$G_s = \frac{G_p G_c G_v}{1 + G_p G_c G_v G_m}$$

We can rearrange this in order to, so let us say if G_s is equal to some desired transfer function which we want to be based on that response which we seek, we want to compute G_c .

The image shows a handwritten derivation on a whiteboard. It starts with the closed-loop transfer function equation: $\frac{1}{G_{s,desired}} = \frac{1 + G_p G_c G_v G_m}{G_p G_c G_v}$. This is rearranged to $\frac{1}{G_{s,desired}} = \frac{1}{G_p G_c G_v} + G_m$. Then, G_m is subtracted from both sides: $\left(\frac{1}{G_{s,desired}} - G_m\right) = \frac{1}{G_p G_c G_v}$. Finally, the controller transfer function G_c is solved for and boxed: $\therefore G_c = \frac{1}{\left(\frac{1}{G_{s,desired}} - G_m\right) G_p G_v}$.

If you want to rearrange this equation, what we will get is,

$$\frac{1}{G_{s,desired}} = \frac{1 + G_p G_c G_v G_m}{G_p G_c G_v}$$

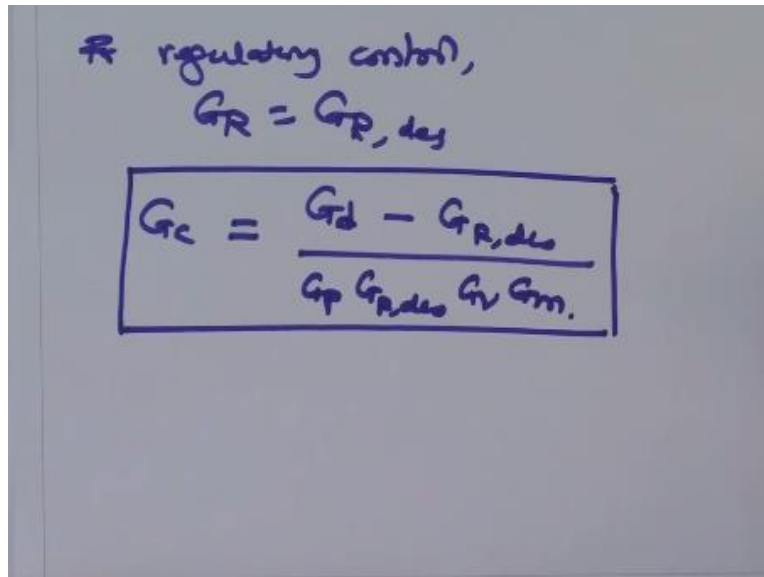
$$\frac{1}{G_{s,desired}} = G_m + \frac{1}{G_p G_c G_v}$$

That gives you a controller transfer function as,

$$G_c = \frac{1}{\left(\frac{1}{G_{s,desired}} - G_m\right) G_p G_v}$$

So what they say is if I use a controller transfer function which is given by this where this is $G_{s,desired}$, I will get the closed loop response equal to $G_{s,desired}$. So what, and if this controller, this transfer function is similar to a PI controller, you will implement the PI controller. If this looks like a PID controller, we will be implementing a PID controller.

Similarly, for the regulatory problem, you can similarly derive that.



Handwritten notes on a whiteboard:

For regulatory control,
 $G_R = G_{R,des}$

The controller gain is given by:

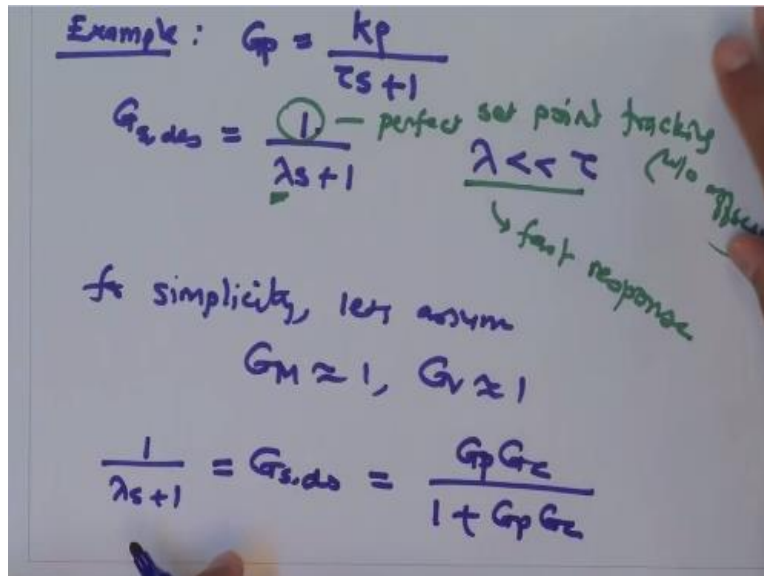
$$G_c = \frac{G_d - G_{R,des}}{G_p G_{R,des} G_v G_m}$$

If your G_R is $G_{R,desired}$, then you can derive that the corresponding G_c comes out to be,

$$G_c = \frac{G_d - G_{R,desired}}{G_p G_{R,desired} G_v G_m}$$

So that is the direct synthesis method. It is not really a tuning method, it is a method of synthesizing a controller and we are going to use it to tune the controller by such that when you select a certain response from as $G_{s,desired}$ or $G_{R,desired}$. So that the resulting controller comes out to be either a P controller, PI controller or a PID controller.

So let us take an example of a first order process.



Let us see your process is first order process and we want this particular process to undergo a servo response should be such that is,

$$G_{s,desired} = \frac{1}{\lambda s + 1}$$

where λ is much smaller than τ . So what we are interested in is we want a perfect setpoint tracking, so when you say the gain is 1, so this gives me perfect setpoint tracking without offset and this lambda being less than tau means it is a fast response. Let us say if this is our desired and for simplicity, let us assume G_m to be 1 and G_v to be 1. So what we get is,

$$G_{s,desired} = \frac{1}{\lambda s + 1} = \frac{G_p G_c}{1 + G_p G_c}$$

We can take the inverse of this equation and we get,

$$\lambda_{s+1} = \frac{1}{G_p G_c} + 1$$

$$\lambda_s = \frac{1}{G_p G_c}$$

$$G_c = \left(\frac{1}{G_p}\right) \cdot \frac{1}{\lambda_s}$$

$$= \frac{(\tau s + 1)}{K_p \lambda_s} \quad \text{PI controller}$$

$$K_c = \left(\frac{\tau}{K_p \lambda}\right) \left[\frac{1}{\tau s} + 1 \right] \rightarrow K_c \left[1 + \frac{1}{\tau s} \right]$$

$$G_c = \frac{1}{G_p} \frac{1}{\lambda_s}$$

So this controller is going to depend on what is the process transfer function that is why it is a model-based controller. So this becomes,

$$G_c = \frac{\tau s + 1}{K_p \lambda s}$$

which we can simplify as,

$$G_c = \left(\frac{\tau}{K_p \lambda}\right) \left[1 + \frac{1}{\tau s} \right]$$

As you can see that this is of the form, $K_c \left[1 + \frac{1}{\tau s} \right]$. So this is indeed a PI controller, with this

being the controller gain and tau being the integral time constant. So what direct synthesis method gives us is that if we use a PI controller with a gain given by,

$$K_c = \frac{\tau}{K_p \lambda}$$

and integral time constant equal to the process time constant of tau. Then the closed-loop response will be given by $\frac{1}{\lambda s + 1}$. So it will be a setpoint tracking first order without offset with a

time constant of lambda. So this is a simpler way to get controller tuning by actually specifying what is the closed-loop response we want.

Let us take another example.

Example 2.1 $G_p = \frac{k_p}{s+1} e^{-tds}$

Let's say $G_{s,desired} = \frac{1}{\lambda s + 1}$, $G_m \approx G_v \approx 1$

$$\frac{1}{\lambda s + 1} = \frac{G_p G_c}{1 + G_p G_c}$$

$$\lambda s = \frac{1}{G_p G_c} \cdot \frac{1}{\lambda}$$

$$G_c = \left(\frac{1}{\lambda}\right) \cdot \frac{1}{G_p}$$

Let us say a process has a dead time, so it is the first order + dead time process. Let us say, we want $G_{s,desired}$ of again a similar type ($\lambda s + 1$). So we have dead time in the process but we want the response which does not have any dead time. Let us see what happens. So we have,

$$G_{s,desired} = \frac{1}{\lambda s + 1}$$

Again we will make similar simplifying assumptions. So what we get is,

$$G_c = \frac{1}{G_p} \frac{1}{\lambda s}$$

$$G_c = \frac{1}{\lambda s} \cdot \frac{\tau s + 1}{k_p} e^{t_d s}$$

$$G_c = \left\{ \left(\frac{\tau}{k_p \lambda} \right) \left[1 + \frac{1}{\tau s} \right] \right\} e^{t_d s}$$

PI control

-ve deadtime
(future)

$$u(t) = u_0 + \frac{k_c}{I} \int_0^t \epsilon(t+t_d) dt + \epsilon(t+t_d)$$

future value & error P

Impractical

Now let us substitute what we have as G_p . So we get,

$$G_c = \frac{1}{\lambda s} \frac{\tau s + 1}{K_p} e^{t_d s}$$

So what we get is very similar to what we had for the previous case. We have controller gain and all this is multiplied by e raise to $t_d s$. If we analyze this controller what we are getting is this looks like a PI control with positive dead time. Actually, it is a negative dead time because for a dead time we get e raise to $-t_d s$ which is same as future.

If you invert this controller what it is going to require is if I want to find out the controller input, it will be the steady state value plus integral 0 to t epsilon and then because of this dead time, it will be $t+t_d$ the error value will be at $t+t_d$. So this is the proportional action, this is the integral action. So what we realize is that because of this negative dead time what we require is the future value of error which is actually not possible because we do not know what is going to happen at the $(t+t_d)$ time, so this controller is impractical.

The reason we are getting the impractical controller from this direct synthesis is that we are assuming or we are we want this controller to take action right away and this is feedback control, it is going to take action only when the output has shown any effect to the input. Because of this dead time, the output is not going to show any effect till the time of t_d is passed. So as that is the

case you cannot expect your controller to take action right away, you have to carry forward the same delay into your desired response as well.

for systems with deadtime, we need to carry-forward some deadtime in the desired response

$$G_{s,des} = \frac{1}{\lambda s + 1} e^{-t_d s}$$

$$\frac{1}{\lambda s + 1} e^{-t_d s} = \frac{G_p G_c}{1 + G_p G_c}$$

$$(\lambda s + 1) e^{t_d s} = \frac{1 + G_p G_c}{G_p G_c}$$

For direct synthesis, so for systems with dead time, we need to carry forward same dead time in the desired response in order to get a practical controller. For the same case a $G_{s,desired}$ should have been,

$$G_{s,desired} = \frac{1}{\lambda s + 1} e^{-t_d s}$$

Accordingly, if we derive the controller what we are going to get is.

$$(\lambda s + 1) e^{t_d s} = \frac{1}{G_p G_c} + 1$$

$$(\lambda s + 1) e^{t_d s} - 1 = \frac{1}{G_p G_c}$$

$$G_c = \frac{1}{G_p ((\lambda s + 1) e^{t_d s} - 1)}$$

$$G_c = \frac{(\tau s + 1)}{e^{t_d s} G_p [(\lambda s + 1) e^{t_d s} - 1]}$$

Let us substitute what we have.

$$G_c = \frac{\tau s + 1}{k_p [(\lambda s + 1) - e^{-t_d s}]}$$

$$e^{-t_d s} \approx 1 - t_d s$$

$$G_c = \frac{\tau s + 1}{k_p [\lambda s + 1 - 1 + t_d s]}$$

$$= \frac{\tau s + 1}{k_p (\lambda + t_d) s} = \left(\frac{\tau}{k_p (\lambda + t_d)} \right) + \left(\frac{1}{k_p (\lambda + t_d) s} \right)$$

PI controller

we approximate,

$$e^{-t_d s} \approx 1 - t_d s$$

So you will get,

$$G_c = \frac{\tau s + 1}{K_p (\lambda + t_d) s}$$

which is equal to,

$$G_c = \frac{\tau}{K_p (\lambda + t_d)} + \frac{1}{K_p (\lambda + t_d) s}$$

Again you can see that this is a PI controller, so this is a P action and this is I so it is PI controller. So for a first order + dead time system, if we properly select the desired response then again we can end up getting a PI controller.

This was for the case of dead time. A similar thing will happen if we have an inverse response into our process. So if the process has an inverse response which means right half-plane zero, then if we use a direct synthesis method. It will be based on the inverse of the process so it will have $1/G_p$ into your controller transfer function. So that right half-plane 0 will become a right half-plane pole into your process so into your controller, so your controller will become unstable.

for systems with inverse response,
we have RHP zero.
a controller $\propto \frac{1}{G_T}$
controller G_c will have pole in RHP
— controller will be unstable.
So, for inverse response system, the desired
response should also have RHP zero.

For systems with the inverse response, we have right half-plane zero as the controller is proportional to one over the process transfer function. Controller G_c will have the same zero will become a pole, will have a pole in right half plane, so the controller will be unstable. So such a controller will not be able to implement such an unstable controller because it will require an infinitely large amount of input as time goes to infinity.

For the inverse response system, the desired response should also be; also have the corresponding right half-plane zero. So inverse response cannot be avoided by simply cannot be eliminated by implementing any controller or direct synthesis controller. So as long as we take care of these two situations that direct synthesis method can give you P, PI or PID controller and the corresponding tuning will give you the desired response which was used to set up that particular controller.

So we will take a short break here and when we come back, we look at a final method of coming up with controller tuning that will be based on frequency response analysis. Thank you.