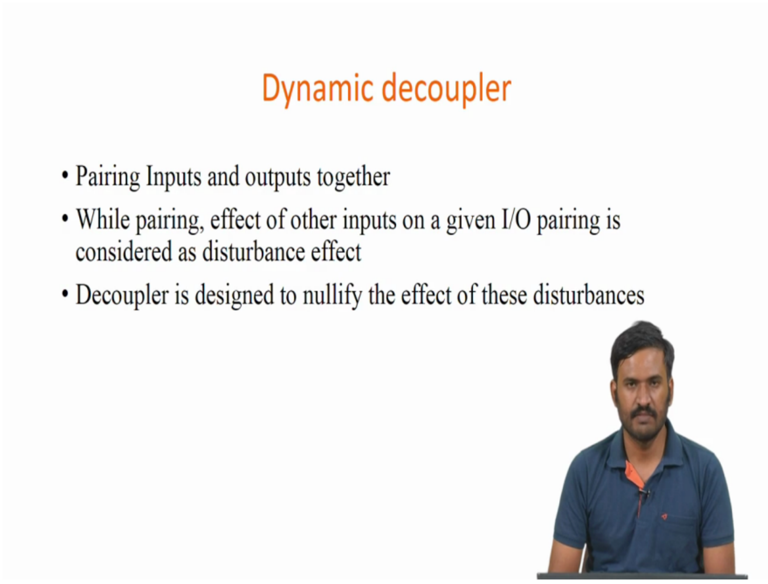
Process Control- Design, Analysis and Assessment
Doctoral Research Scholar Manikandan S
Department of Chemistry
Indian Institutes of Technology, Madras
MATLAB Tutorial – Controller Design -Part 2

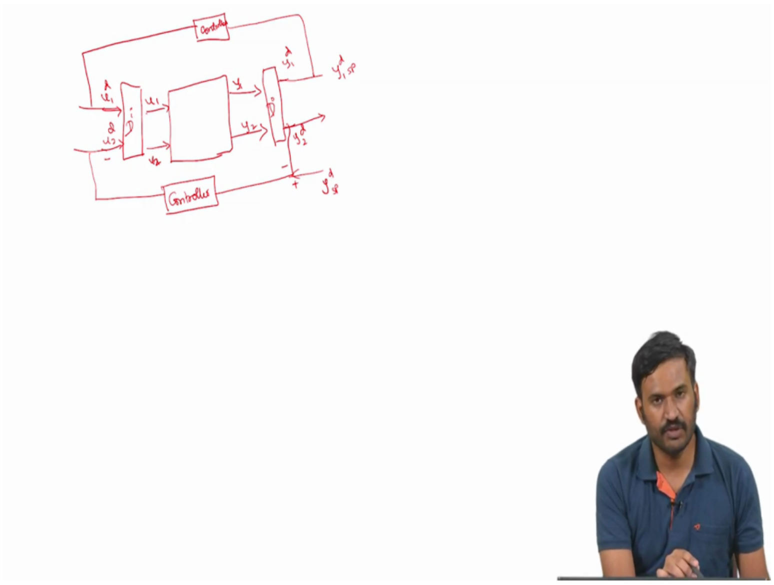
Welcome everyone to the tutorial for process control analysis, design and assessment. In this tutorial we will start looking at MIMO systems and the controller design for MIMO systems. First we will look at Dynamic decoupler design where we try and pair inputs and outputs in a one-to-one basis and then control the whole system based on this configuration.

(Refer Slide Time: 0:49)



So Dynamic decoupler tries to nullify the effects of other inputs on the given input and output parrying. So in order to do that we consider the effect of other inputs on this output to be equivalent to some disturbance effect, so that D coupler is essentially design to nullify the effect of such other input other than the pair which we are pairing particular output with.

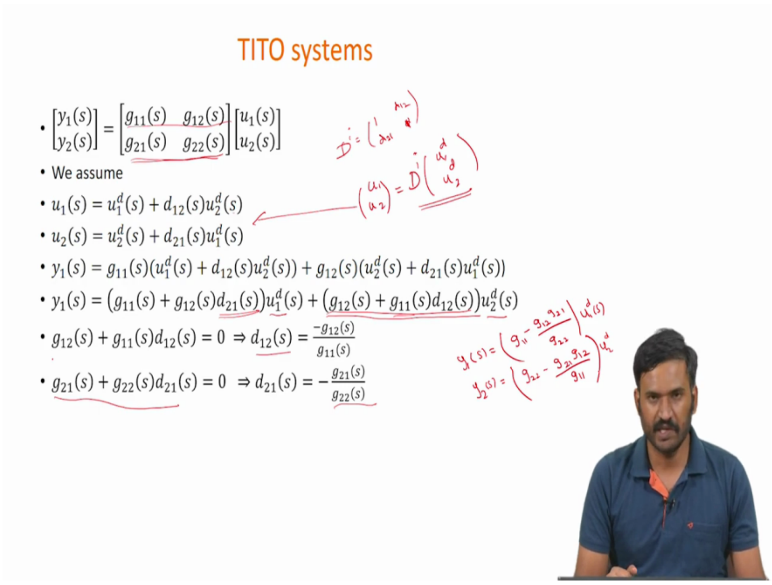
(Refer Slide Time: 1:29)



So let us consider a mime system where we have 2 inputs and 2 outputs, now both U1 and U2 will have to effect on Y1 as well as Y2, because of that we want to find some linear transformation such that we have variables which are decoupled such that this Y1 D and Y2 D are dependent on only the decoupled use called U1 D and U2 d. So now this outer loop can be thought of as a seesaw controller if we were to put a controller here. Y D set point and Y 1 D set point are the set points we choose the system to reach, based on that we will compute the error and the controller will give the corresponding decoupled U. So we have some decoupler the transformation for input and decoupler transformation for output.

By default we choose output decoupler to be identity so that we have a notion of the set point as the objective of variables Y1 and Y2. But the D I will be we will have to choose D I such that the effect of U2 for example and Y1 is 0 and effect of and the effect of U1 on Y2 is 0.

(Refer Slide Time: 3:54)

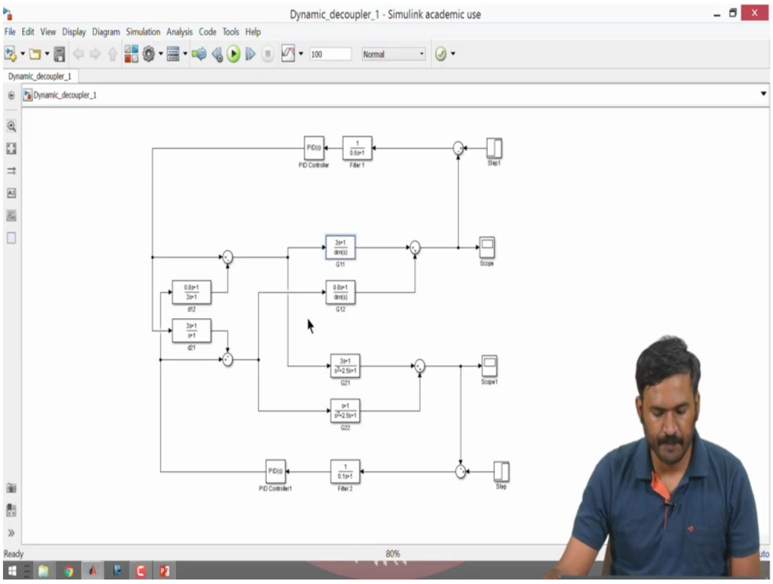


So let us take two input two output system where we have G 11, G 12, G 21 and G 22 as the transfer function multiplying U1 and U2 for Y1 and Y2. So we choose the linear transformation matrix such that it is 1 and D 12, D 21 and 1, so how we decouple U1 and U2 is like this. So U1 U2 equal to D I times U1 D and the U2 D. If you were to expand this equation we will get these 2 set of equations, now you can see U1 of S is U1 D of S + D 12 times U2 D similarly, we have U2. This we can back substitute in these 2 equations and separate the terms based on the decoupled input variables U1 D and U2 D.

The Dynamic D coupler design works in such a way that the effect of either U1 and Y1 or U2 and Y2 is made to 0, so in this case we choose to make the effect of U2 on Y1 to be 0. In order for that to become 0 this term has to go to 0, based on that we compute D 12 as minus G 12 divided by G 11. Similarly we have Y2 and we eliminate the effect of U1 and Y2 and that equation is this and solving this will lead to D 21 of Sbeing minus G 21 divided by G 22 of S. The important point to notice U1 is U1 D of S plus D1 2 into U2 D of S, but Y1 S is G 11 of S plus G 12 of S times D 21 of S into U1 D. So this D 21 is the function of G 21 and G 22 and D 12 is the function of G 11 and G 12.

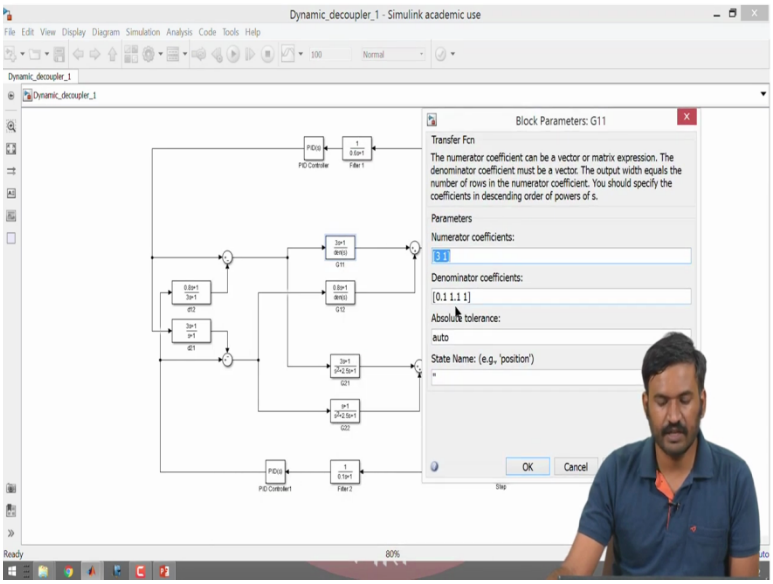
Now we can back substitute and see whether it will decouple properly, so why of S is G 11 minus G 12 into G 21 divided by G 22 into U1 D of S. Not because we equated this term to 0 this will go to 0 similarly, we will have Y2 of S equals to G 22 – G 21 times G 12 divided by G 11 into U2 D of S. Now how do we configure and implement such the coupler in simulink is what we are going to see next.

(Refer Slide Time: 7:43)



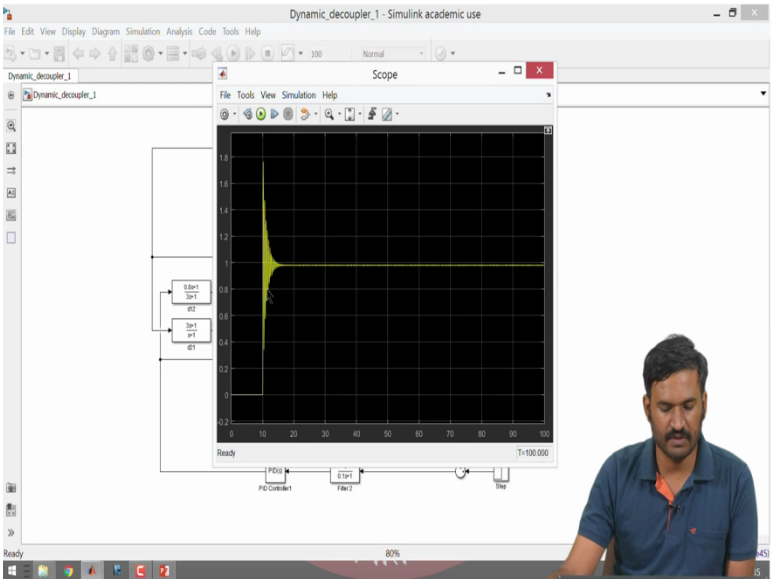
So here notice that I have given G 11, G 12 up to G 22 as separate individual transfer functions. The transfer function block in simulink allows seeing more system to be specified, but if we were to specify the whole 2 cross 2 system as 2 two single systems, then we cannot have the coupler effect so because of that I have defined the transfer functions individually; G 11 individually, G 22 individually, G 21 and G 22.

(Refer Slide Time: 8:34)



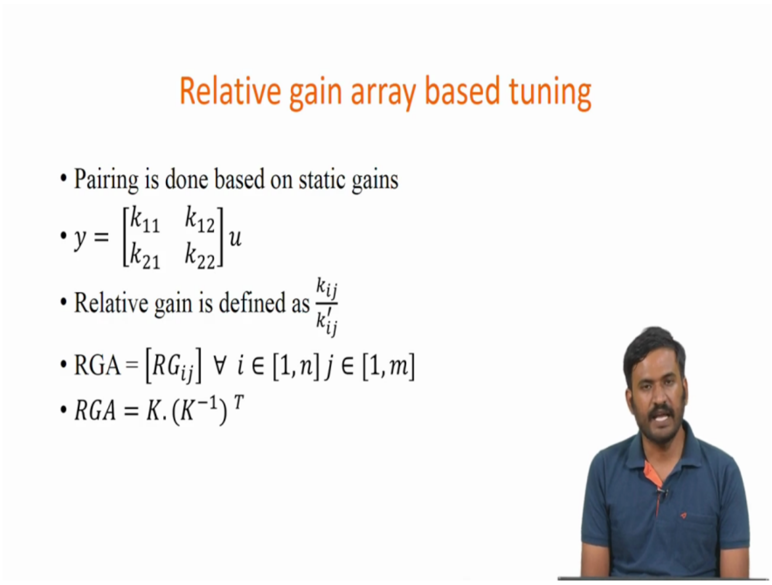
So the values are like this, so we have 3 S plus 1 divided by 0.1 S square + 1.1 S plus 1 and 0.8 S + 1 divided by 0.1 S square + 1.1 S plus 1 and 3 S + 1 divided by S square plus 2.5 S + 1, and S + 1 divided by S square + 2.5 S plus 1 sorry S + 1 divided by S square plus 2.5 S + 1. Now based on these values we can calculate D 11 and D 21 which are given here. Now the controller PID controller for U1 and U2 will give U1 decoupled and U2 decoupled, the U1 which we are giving it as an input to the system is U 1 D of S + D 12 times U 2 D of S. So we have one term coming from directly from PID controller which is U1 D of S and then there is another term coming from U2 D of S which goes into D 21 and then added. Notice that there is a negative sign in D 12 which I have translated to this submission block being subtracted; both actions will result in same result.

(Refer Slide Time: 10:42)



The PID controller design I have designed such a way that we will have simple filter along with typical P, I and D values, so you can see that let us run the system for 100 seconds and then we see how the system behaves. So we have given unit step input to the system, as you can see there are initial oscillations and that seems to be some offset which is like 0.2 or something sorry 0.02 or something. Similarly you have minimal oscillatory variation in the 2nd output and there is some offset which is 0.02. So this has to be eliminated through proper design of PID controller and the effect of the model plan mismatch will be more in these cases where we do dynamic decoupling for MIMO systems.

(Refer Slide Time: 11:36)

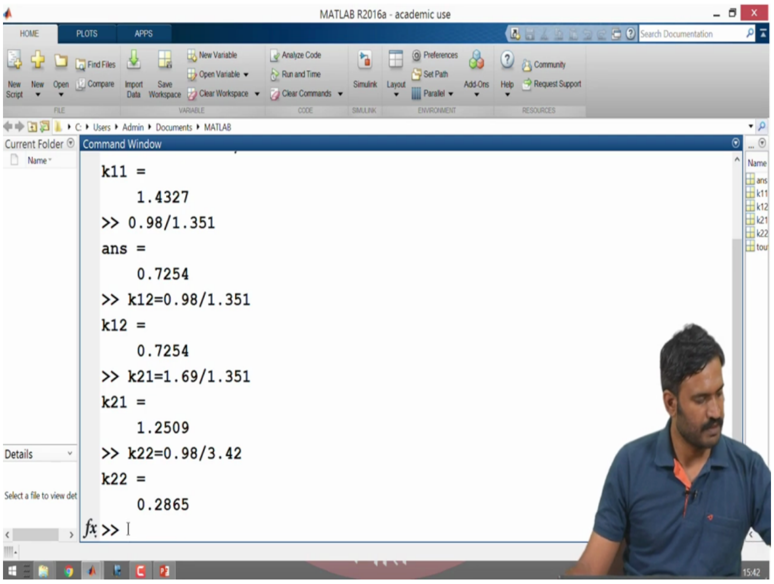


Let us look at the look at another method where we use relative gain array to pair the MIMO systems such that we will have reasonable control with minimal offset and minimal computation also. So the idea is, pairing is done based on static gains and we can compute these static gains values by final value theorem for a given transfer function. So let us suppose we have done that then we will have Y which consist of Y1 and Y2 is equal to K 11 times U1 plus K 12 times U2, similarly Y2 is K 21 times U1 plus K 22 times U2. We define some terminology is relatives gain as K IJ divided by K IJ dash for all I and J values.

Now this K IJ is the gain of the system, K IJ dash is the gain of the system where all other transfer functions, all other loops are in closed loop and only the loop in question is in open loop so in such case what is the gain of the system is computed as K IJ dash. Now we call the term K IJ by K IJ dash as RG related gain IJ, again I runs from 1 to n and J runs from 1 to m, they are integers so they are only integers. Based on the structure of the matrix K here, this is K matrix so based on this structure we can compute relative gain for each and every terminology, so we will have Y1 pairing with U1 and Y1 pairing with U2. Similarly we will have Y2 and U1 and Y2 and U2, so we have 4 terms to calculate.

So if we calculate each terms and then assemble the numbers in matrix, we find that we can reduce the whole operation into K dot K inverse whole transposed, the dot operator is element wise multiplication. So we will look at one simulation example simulink example, where we use RGA Relative Gain Array based tuning to design the controller. So as you can see I have once again configured each and every transfer function individually and then I have used one PID for controlling each block, but we have to find which type of pairing is the best.

(Refer Slide Time: 15:48)



So in order to do that let us see what each final value is and we construct the K matrix and then we will see how each input has to be paired along with output value. Now the final value of this transfer function which is 0.659 divided by S + 0.395 is 0.566 divided by 0.395. Let us calculate what it is K 11 equals to 0.5659 divided by 0.395, so which is 1.4327. Similarly final value of the 2nd system for unit step changes is 0.98 divided by 1.351 which is so 0.7254. The 3rd element is 1.69 divided by 1.351 which is 1.2509, and the final element is 0.984 divided by 3.42.

Now the matrix K is K 11, K 12, K 21 and K 22, so let us compute K dot star inverse of K and transpose. Notice, I have used dot star here in order to compute element these operations and the transpose is computed when we can use the symbol apostrophe. So we find that the best pairing is the pairing which contains nonnegative close to one relative gain. So here you see the first term which is relating Y1 to U1 has negative coefficient, whereas Y1 to U2 has positive coefficient so we have to pair Y1 to U2 and Y2 to U1. So the output of the PID controller is U1 and U2, I have assumed Y2 setpoint to compute U1 so I have given connections to G 11 and G 21. Similarly I have calculated I have used Y1 setpoint and Y1 together to compute U2 so I have given connections to G 12 and G 22.

If the pairing is reverse, this connectivity that is connecting U1 to corresponding transfer function and U2 to corresponding transfer function will change. The PID I have tuned using sales will G desire in direct synthesis method which is 1 by 0.1 S plus 1 and then I have used the corresponding parameters here. Or we can tune it now itself that using trial and error method. So as you can see these parameters are not optimal now so we can do the tuning here, in this case we have seen that Y2 is getting stabilised here, Y1 is still not so we will change that operation or we can as well use the auto tune method but, so this is how you can see, the 1st output reaches 1 fairly quickly within 8 seconds, whereas there is an offset in the 2nd output.

So we can use direct synthesis method of tuning to compute PID parameters but since we are not decoupling base in a dynamic sense, we might have to use starting gains 1st in order to compute direct synthesis method based tuning. With this I will stop this tutorial, we will look at model predictive control based controller design in the next tutorial, thank you.