Universidade Federal do Ceará
Departament of Teleinformatics
Computer Engeneering
Intelligent Systems in Process Control and Automation

Homework 2

Control of a Dynamic System

Student: Davi Barreto Façanha

Professor: Prof. Dr. Michela Mulas

Fortaleza
2019

Contents

1 Introduction

The objective now is to do a Multiloop and Multivariable Controlling Analyse. But first, we should understand which are the the Control Variables and the Model Variables, and how each one affects the behavior of our system.

1.1 Degrees of Freedom Analysis

First, we must ensure that our model is made up of a solvable system. For this, there are two equations with which we can calculate the degrees of freedom of the model and the control, respectively,

$$N_F = N_V - N_E \tag{1}$$
$$N_{FC} = N_D - N_F \tag{2}$$

where N_V is the total number of process variables (distinct from constant process parameters) and N_E is the number of independent equations.

Recapping our Three-Tanks System, we have:

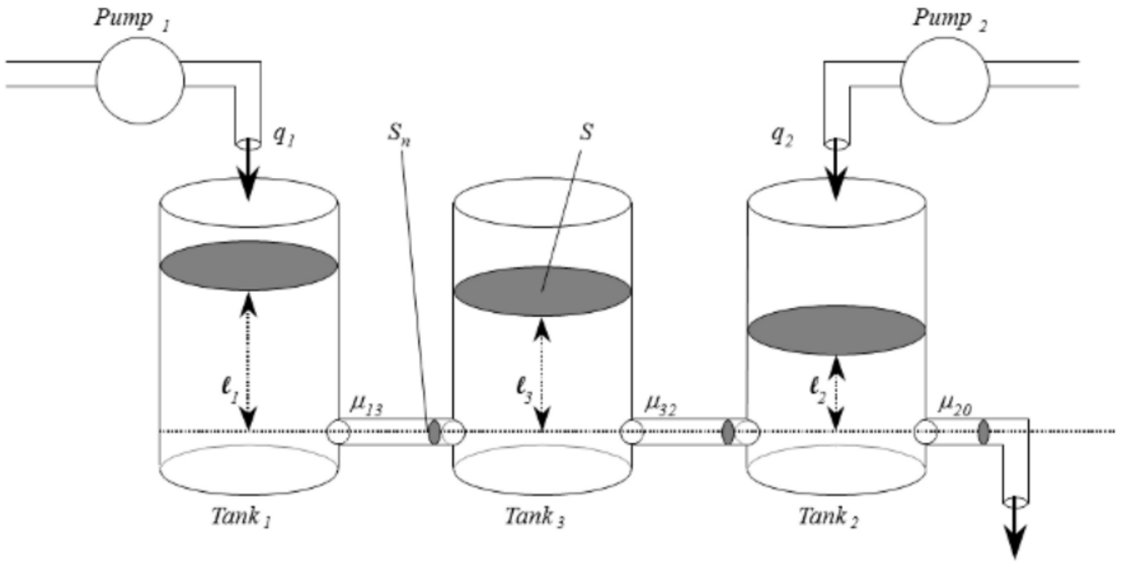


Figure 1: Three-Tanks System

$$\dot{l}_1 = \frac{1}{S} \cdot (q_1 - q_{13})$$

$$\dot{l}_2 = \frac{1}{S} \cdot (q_2 + q_{32} - q_{20})$$

$$\dot{l}_3 = \frac{1}{S} \cdot (q_{13} - q_{32})$$

$$q_{mn} = \mu_{mn} S_p sign(l_m - l_n)\sqrt{2g|l_m - l_n|}$$

$$q_{20} = \mu_{20} S_p \sqrt{2g \cdot l_2}$$

The two input valves(N_E) are above the 1st and 2nd tank, that means the 3rd tank depends on the other's levels. Ergo, the indepent equations(N_V) are only $\dot{l}_1$ and $\dot{l}_2$. So that our $N_F = 0$. This result can be classified according to the following paradigms as an **exactly specified** system (*Process Dinamics and Control: Seborg, Edgar, Mellichamp and Doyle - 4ed*):

1. $N_F = 0$: The process model is **exactly specified**. If N F = 0, then the number of equations is equal to the number of process variables and the set of equations has a solution.
2. $N_F > 0$: The process is **underspecified**. The N_E equations have an infinite number of solutions, because N_F process variables can be specified arbitrarily.
3. $N_F < 0$: The process model is **overspecified**. The set of equations is insoluble.

Also, the disturbances are not defined by any pattern or model. With this, we have $N_D = 0$ and, consequently, $N_{FC} = 0$.

1.2 Correlation and the Gain Matrix

Understood the Degrees of Freedom Analysis, we are dealing with a Multiple-Input-Multiple-Output(MIMO) case whose number of manipulated variables and whose number of controlled variables are the same, so, a Square System. But a MIMO system has some difficulties. The inter-connection between the three tanks makes the amount of some input influences the levels of others tanks beyond itself. We can check the Superposition Principle by the Transfer Functions.

$$\frac{L_1}{U_1} = \frac{64.94s^2 + 2.572s + 0.01905}{s^3 + 0.04956s^2 + 0.0005885s + 9.667 \cdot 10^{-7}} = G_{11}(s)$$

$$\frac{L_1}{U_2} = \frac{0.006428}{s^3 + 0.04956s^2 + 0.0005885s + 9.667 \cdot 10^{-7}} = G_{12}(s)$$

$$\frac{L_2}{U_1} = \frac{0.006428}{s^3 + 0.04956s^2 + 0.0005885s + 9.667 \cdot 10^{-7}} = G_{21}(s)$$

$$\frac{L_2}{U_2} = \frac{64.94s^2 + 1.939s + 0.006428}{s^3 + 0.04956s^2 + 0.0005885s + 9.667 \cdot 10^{-7}} = G_{22}(s)$$

If we express the input-outputs in a vector-notation, the Gp Matrix is going to represent a specie of "Frequency Domain Correlation Matrix".

$$Y(s) = G_p(s)U(s)$$

$$G_p(s) = \begin{bmatrix} G_{11}(s) & G_{12}(s) \\ G_{21}(s) & G_{22}(s) \end{bmatrix}$$

Setting the s = 0, we gonna find the Process Gain Matrix, which means the level of sensitivity of each output for input changes.

$$K = G_p(s) = 10^4 \cdot \begin{bmatrix} 1.9706 & 0.6649 \\ 0.6649 & 0.6649 \end{bmatrix}$$

The Block Diagram of the Superposition Principle works as in Figure .

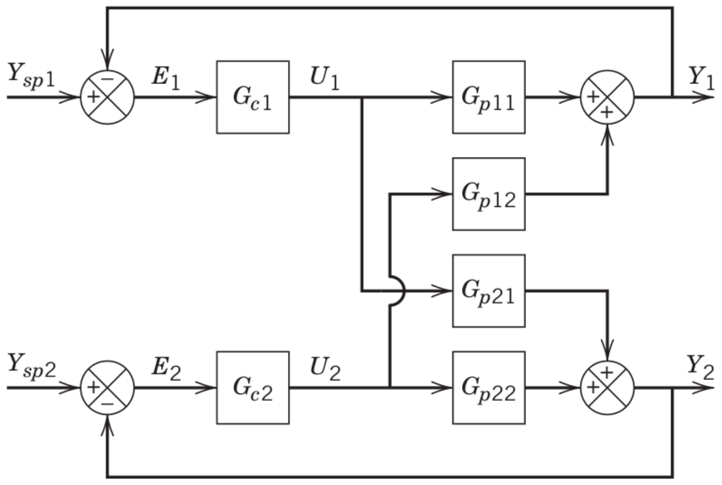


Figure 2: 1-1/2-2 Control Scheme

The inputs U1 and U2 has direct and indirect effects on each output. That implicates in a third extra loop that causes instability. For some cases, the influence of some input affects all the outputs in a very strong way, and beacuse of that, the Process Gain Matrix would have high values of sensitivity. But for other cases too, by calculating the Relative Gain Array, in section 2.1, we could discover that our system works quite independently, and that the inputs and outputs are not very combined. Therefore, would be possible to make a logical decoupling for the Tunning of the Controller.

2 Multiloop and Multivariable Control

Therefore, to solve that dilemma, it will be necessary to first do a pairing of controlled and manipulated variables. Doing so, will be possible to find a controller capable of controlling it.

2.1 Bristol's Relative Gain Array

Using the RGA method, it is possible to measure the process interactions and to achieve some recommendation of the most effective pairing of controlled and manipulated variables.

$$\lambda_{ij} = \frac{\left(\frac{\partial \lambda_i}{\partial u_j}\right)_u}{\left(\frac{\partial \lambda_i}{\partial u_j}\right)_y} = \frac{Open-loop \quad Gain}{Closed-loop \quad Gain} \quad for \quad i = 1,2,3,...,n \quad and \quad j = 1,2,3,...,n$$

Each lambda represents a Relative Gain. However, It is convenient to arrange the relative gains in a relative gain array (RGA):

$$\Lambda = \begin{bmatrix} \lambda_{11} & ... & \lambda_{1n} \\ ... & ... & ... \\ \lambda_{n1} & ... & \lambda_{nn} \end{bmatrix}$$

The RGA has several important properties for steady- state process models (Bristol, 1966; McAvoy, 1983):

1. It is normalized because the sum of the elements in each row or column is equal to one.
2. The relative gains are dimensionless and thus not affected by choice of units or scaling of variables.
3. The RGA is a measure of sensitivity to element uncertainty in the gain matrix K. The gain matrix can become singular if a single element K ij is changed to $K_{ij}(1 - \frac{1}{\lambda_{ij}})$. Thus a large RGA element indicates that small changes in K_{ij} can markedly change the process control characteristics.

The relative gains can easily be calculated from either steady-state data or a process model. So, we would have:

$$y_1 = K_{11}u_1 + K_{12}u_2$$
$$y_2 = K_{21}u_1 + K_{22}u_2$$

Putting in the matrix notation, the K matrix will be the Gain Matrix. Setting $s = 0$, as in section 1.2, we have the Process Gain Matrix. After that, we apply the first Partial Derivative to find the Open-Loop Gain.

$$\left(\frac{\partial y_1}{\partial u_1}\right)_{u_2} = K_{11}$$

For the Closed-Loop Gain Partial Derivative, we first need to solve the system for y_2 constant at its nominal set point value, $y_2 = 0$, and isolating u_2.

$$u_2 = \frac{K_{21}}{K_{22}} u_1 \quad \rightarrow \quad y_1 = K_{11} u_1 + \frac{K_{21}}{K_{22}} u_1 \quad = \quad K_{11}\left(1 - \frac{K_{12}K_{21}}{K_{11}K_{22}}\right) u_1$$

$$\left(\frac{\partial y_1}{\partial u_1}\right)_{y_2} = K_{11}\left(1 - \frac{K_{12}K_{21}}{K_{11}K_{22}}\right)$$

Following the steps, we find:

$$\lambda_{11} = \frac{1}{\frac{K_{12}K_{21}}{K_{11}K_{22}}}$$

Thus, since each row and each column sums 1, the RGA can be expressed as a symmetric matrix for a 2x2 case:

$$\lambda_{12} = \lambda_{21} = 1 - \lambda_{11} \quad \textit{and} \quad \lambda_{22} = \lambda_{11}$$

$$\Lambda = \begin{bmatrix} \lambda & 1-\lambda \\ 1-\lambda & \lambda \end{bmatrix}$$

For higher-dimension processes, the RGA can be calculated from the expression

$$\Lambda = K \otimes (K^{-1})^T$$

$$\Lambda = \begin{bmatrix} 1.9706 \cdot 10^4 & 0.6649 \cdot 10^4 \\ 0.6649 \cdot 10^4 & 0.6649 \cdot 10^4 \end{bmatrix} \otimes \begin{bmatrix} 0.0766 \cdot 10^{-3} & -0.0766 \cdot 10^{-3} \\ -0.0766 \cdot 10^{-3} & 0.2270 \cdot 10^{-3} \end{bmatrix} = \begin{bmatrix} 1.5093 & -0.5093 \\ -0.5093 & 1.5093 \end{bmatrix}$$

The closer to one, the more sensitive is the ratio of input to output. Thus we can see that inputs 1 and 2 influence much more the outputs directly connected to them than any other. In this way, we can disregard relationships outside the diagonal of the matrix, and treat the system as 2 SISOs (Figure 3).

3 Controlling Methods

When treating the system as 2 SISOs, we will need to construct two PID controllers, one for each SISO. To do this, two different methods can be used: **Design the controller based on the IMC tuning relations** and **Design the controller based on Online Controller Tuning methods**.

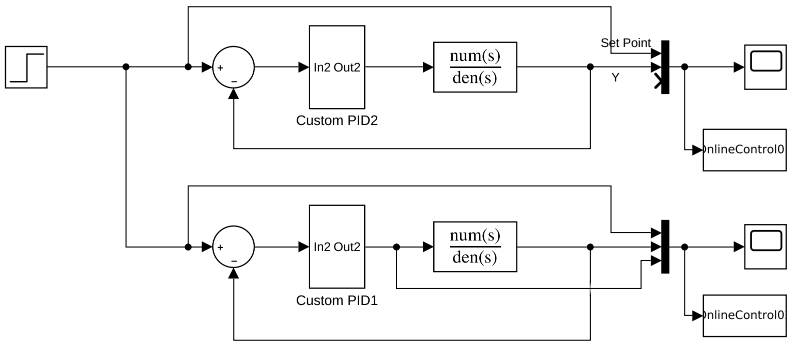


Figure 3: One MIMO into two SISOs

3.1 Internal Model Control Method

3.1.1 Original Internal Model Control Approach

The Internal Model Control is a Model-Based Method, it means that from the model of a system, we can do mathematical manipulations to find the configurations which control our real system. In Figure 4 we can see how the Internal Model Control works in relation to the Conventional Feedback Control.

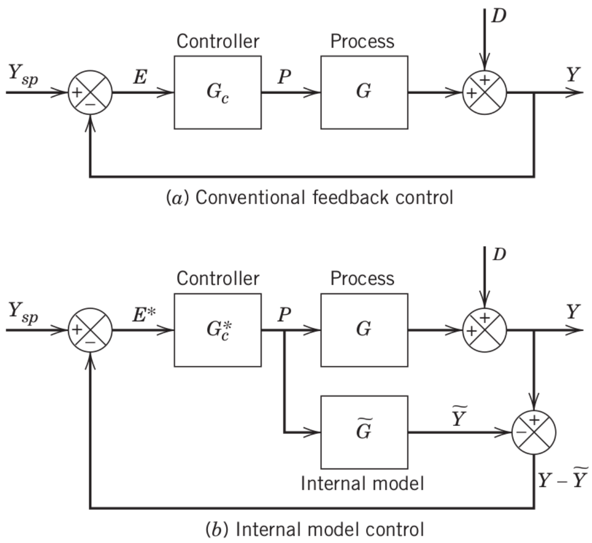


Figure 4: Internal Model Control Scheme

The output of our Gc Controller G_c^* goes through both the Process G (Real System) and the Linear Model (Internal Model). In this way, the difference between the two outputs will be our feedback input. This difference exists because our model will hardly be a perfect approximation of our real system, and therefore will behave differently for the same input.

$$G_C = \frac{G_C^*}{1 - G_C^* \widetilde{G}}$$

The IMC can be design in two steps:

1. First, we factorate our model

$$\widetilde{G} = \widetilde{G}_+ \widetilde{G}_-$$

   Where $\widetilde{G_+}$ contains any time delays $e^{-\theta s}$ or any right-half plane zeros. In the next step is possible to understand why it is important to do so.

2. Than we design the IMC Controller as

$$G_C^* = \frac{1}{\widetilde{G}_-} f$$

   where f is a low-pass filter with a steady-state gain of one. The general low-pass filter equation is like

$$f = \frac{1}{(\tau_c s + 1)^r}$$

Thus, it is necessary to factorize because by inverting the $\widetilde{G}$, the *right-half plane zeros* and the *time delays* would become unstable poles and advances in time (prediction terms) $e^{+\theta s}$. By using only the $\widetilde{G}_-$, we can ensure that our control is physically achievable and stable.

The r value of the filter can usually be chosen as 1. However, depending on the Transfer Function of our model, by inverting $\widetilde{G}_-$ to design the IMC, we could have more poles than zeroes. The filter appears as a way to correct that.

In our case, we choose $\widetilde{G} = \widetilde{G}_-$ because both transfer functions have no *right-half plane zeros* or *time delay*, and $r = 1$.

1. **For our first decoupled system**

$$G_C^* = \frac{1}{\widetilde{G_-}} \frac{1}{(\tau_c s + 1)^r} = \frac{s^3 + 0.04956 s^2 + 58.85 \cdot 10^{-5} s + 96.7 \cdot 10^{-8}}{(\tau_c + 1.0)(64.94 s^2 + 2.572 s + 0.01905)}$$

$$G_C^* = \frac{s^3 + 0.04956 s^2 + 58.85 \cdot 10^{-5} s + 96.7 \cdot 10^{-8}}{64.94 s^3 \tau_C + 2.572 s^2 \tau_C + 0.01905 s \tau_C + 64.94 s^2 + 2.572 s + 0.01905}$$

$$G_C^* = \frac{s^3 + 0.04956 s^2 + 58.85 \cdot 10^{-5} s + 96.7 \cdot 10^{-8}}{(64.94 \tau_C) s^3 + (2.572 \tau_C + 64.94) s^2 + (0.01905 \tau_C + 2.572) s + 0.01905}$$

That way, our controller will have a different impact on our model, depending on the value of τ_C chosen.

```
% 1st Transfer Function Block to Simulink
numIMC1 = [1, 0.04956, 0.000588500000000000473787675758 7856,
   0.000000966999999999999940538449733074] ;
denIMC1 = [64.945*tauC, (2.572*tauC + 64.945), (0.01905*tauC + 2.572),
   0.01905] ;
```

2. **For our second decoupled system**

$$G_C^* = \frac{1}{\widetilde{G_-}}\frac{1}{(\tau_c s+1)^r} = \frac{s^3 + 0.04956s^2 + 58.85\cdot 10^{-5}s + 96.7\cdot 10^{-8}}{(\tau_c + 1.0)(64.94s^2 + 1.939s + 64.28\cdot 10^{-4})}$$

$$G_C^* = \frac{s^3 + 0.04956s^2 + 58.85\cdot 10^{-5}s + 96.7\cdot 10^{-8}}{(64.94\tau_C)s^3 + (1.939\tau_C + 64.94)s^2 + (64.28\cdot 10^{-4}\tau_C + 1.939)s + 64.28\cdot 10^{-4}}$$

Tha same thing for the second one.

```
% 2st Transfer Function Block to Simulink
numIMC2 = [1, 0.04956, 0.0005885000000000000473787675758 7856,
   0.000000966999999999999940538449733074] ;
denIMC2 = [64.945*tauC, (1.939*tauC + 64.945), ((64.28e-4)*tauC + 1.939),
    64.28e-4] ;
```

3. **Integrated system**

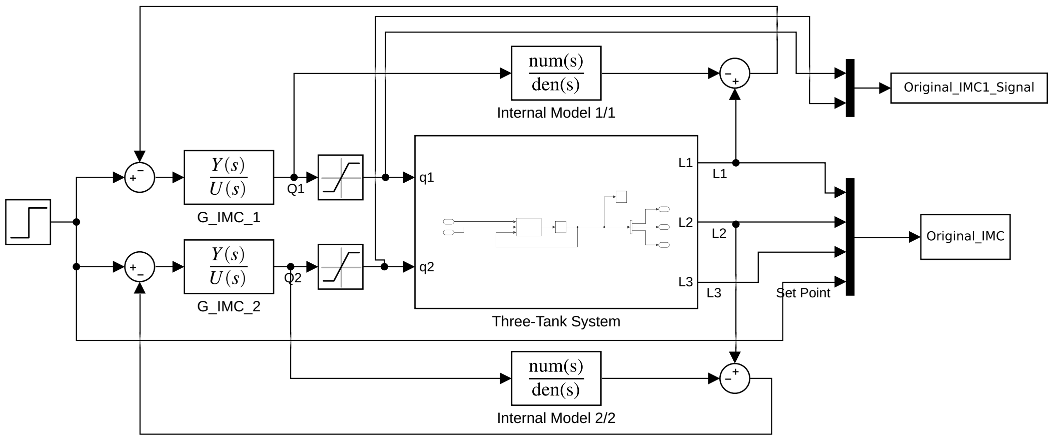
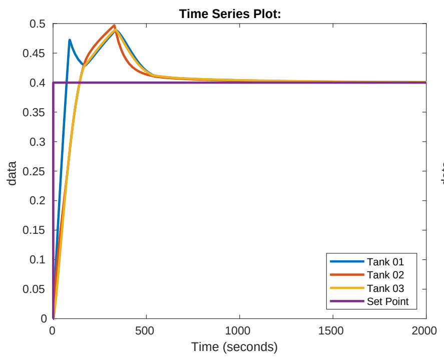


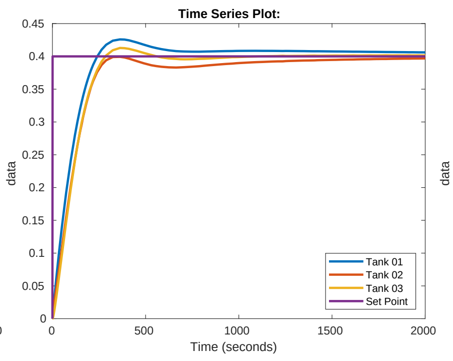
Figure 5: 1-1/2-2 Control Scheme

Now joining everything in the configuration of Figure 4, and by varying the value of τ_C, we can find the results in Figure 6.

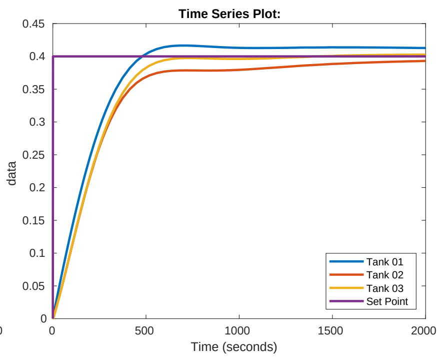
For $\tau_C = 100$, we can see that the answer is already quite satisfactory. However, for $\tau_C = 200$, the PID Signal is much more attenuated and accommodates less abruptly.



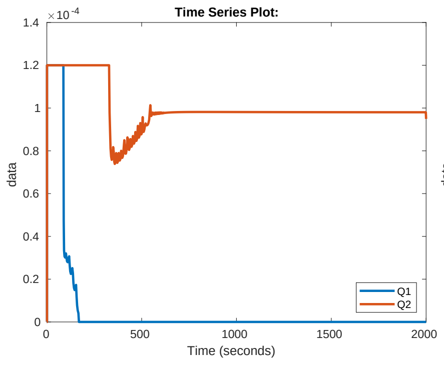
(a) Tank Level $\tau_C = 1$



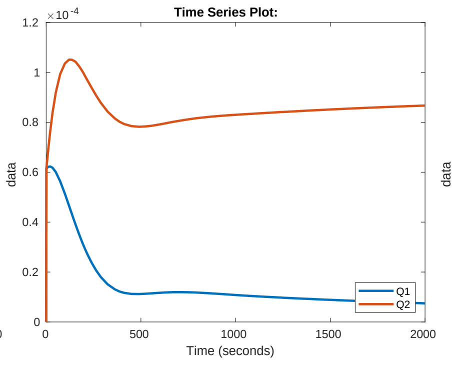
(b) Tank Level $\tau_C = 100$



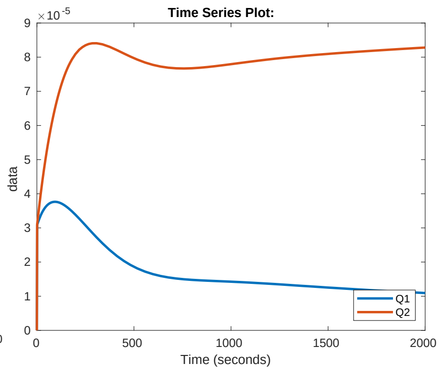
(c) Tank Level $\tau_C = 200$



(d) PID Signal $\tau_C = 1$



(e) PID Signal $\tau_C = 100$



(f) PID Signal $\tau_C = 200$

Figure 6: Levels and Signals for the Online Controlling Method

3.1.2 Equivalence of Internal Model Control Approach

The Tunning of the PID can be made too by the table of Figure 17.

Rewriting our transfer functions, we have

$$\frac{L_1}{U_1} = \frac{(s + 0.0297431)(s + 0.00986273)}{(s + 0.0322325)(s + 0.0153771)(s + 0.0019504)}$$

$$\frac{L_2}{U_2} = \frac{(s + 0.02606)(s + 0.00379829)}{(s + 0.0322325)(s + 0.0153771)(s + 0.0019504)}$$

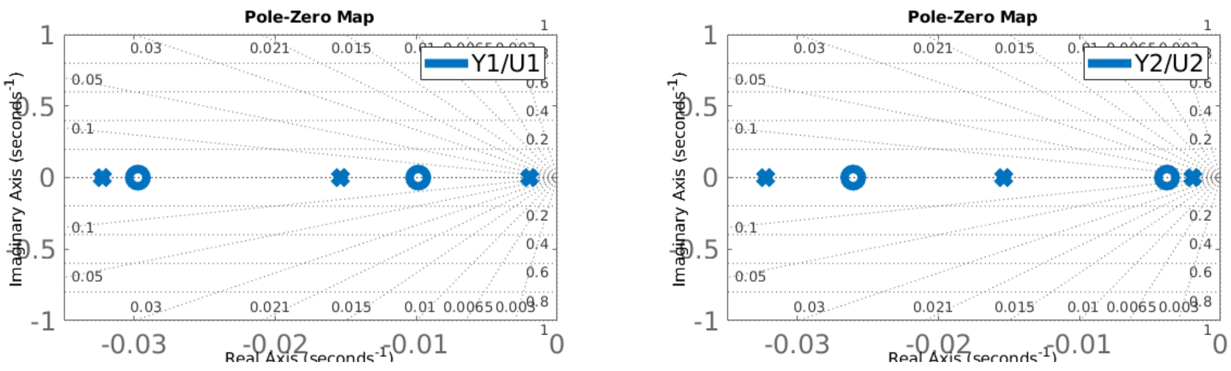


Figure 7: Poles and Zeros Plot

Now, by analyzing the equations and our poles and zeros, we can see that for both subsystems we have a pole and a zero that are very close to each other and relatively far from the center compared to the other poles and zeros. With this, we can assume that they are under the same point and disregard them from the equation.

Therefore, we can now approximate our subsystems to item a) of Figure 4, with $\theta = 0$. I will analyze the controller values using the $\widetilde{G}$(Linear approximation) of each separate state. Only after doing the Tuning of the Linear System will the simulation be performed in the nonlinear one.

1. **For our first decoupled system**

$$\frac{L_1}{U_1} = \frac{0.00986273(101.3918s + 1)}{0.0153771(65.0318s + 1)0.0019504(512.7153s + 1)}$$

$$\frac{L_1}{U_1} = \frac{328.8509(101.3918s + 1)}{(65.0318s + 1)(512.7153s + 1)} = \frac{K(\tau_3 s + 1)}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

$$\begin{cases} K &= 328.8509 \\ \tau_3 &= 101.3918 \\ \tau_1 &= 65.0318 \\ \tau_2 &= 512.7153 \end{cases}$$

By the table, we find $Ki = 0.0021$, $Kd = -31.4033$, and Kp as a function of τ_C. In this way, we can regulate the value of τ_C to find a good control for our system.

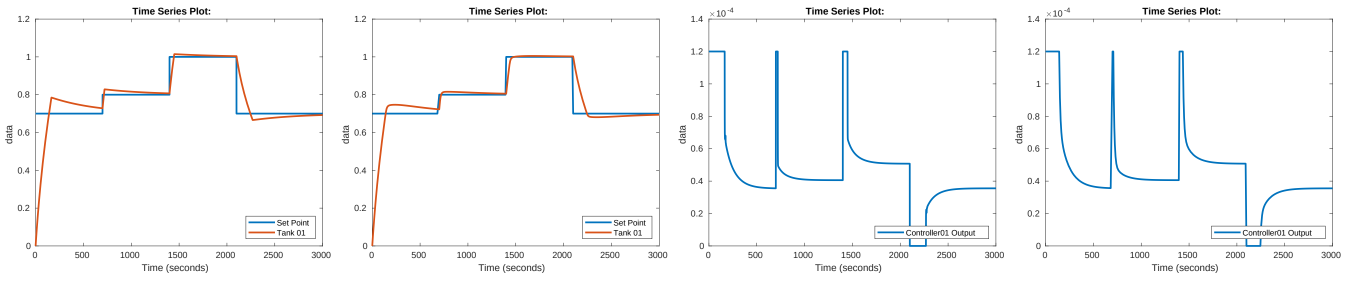


Figure 8: For $\tau_C = 50$ and 1000, respectively

We can see that τ_C does not have much influence on the behavior of the system. This is due to the fact that τ_C, by Figure 17, has only a small influence on just K_P. But K_I and K_D are dependent only of the function coefficients.

Therefore, for the first subsystem, the best option value appears to be $\tau_C = 1000$.

2. **For our second decoupled system**

$$\frac{L_2}{U_2} = \frac{0.00379829(263.2764s + 1)}{0.0153771(65.0318s + 1)0.0019504(512.7153s + 1)}$$

$$\frac{L_2}{U_2} = \frac{126.6456(263.2764s + 1)}{(65.0318s + 1)(512.7153s + 1)} = \frac{K(\tau_3 s + 1)}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

$$\begin{cases} K &= 126.6456 \\ \tau_3 &= 263.2764 \\ \tau_1 &= 65.0318 \\ \tau_2 &= 512.7153 \end{cases}$$

We do the same thing for the second one. By the table, $Ki = 0.0032$, $Kd = -157.2613$, and Kp as a function of τ_C. Ergo, we also need to find good value for τ_C.

Checking Figure 9, it can be seen that the response of the second subsystem is very close to the first subsystem. If we pay close attention, we can see that in the second subsystem, for the same variation of τ_C, we have a slightly larger overshooting for the second simulation.

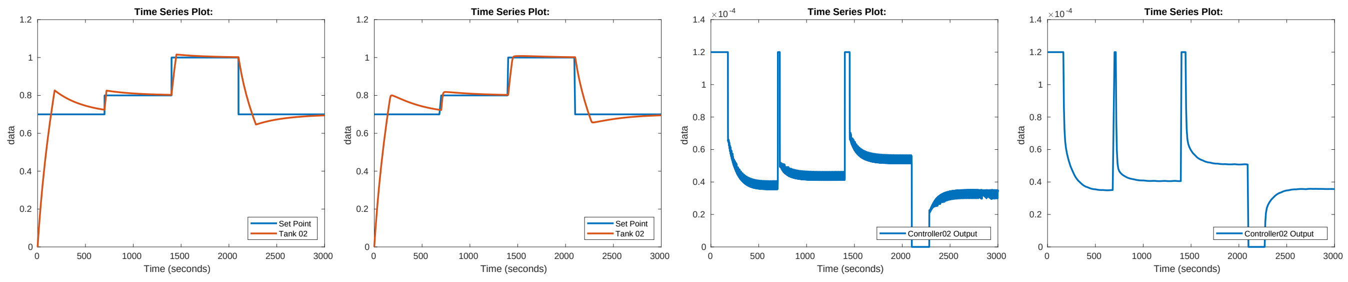


Figure 9: For $\tau_C = 50$ and 1000, respectively

For testing purposes, I even used a much larger τ_C value to test the operation and realized that it is possible to greatly attenuate both the Signal and the Output curve. However, the system response is extremely time consuming.

Also, it is important to note that a Saturator was used, so that it forces the system to respect its limits and, therefore, also makes the response much more time consuming for both subsystems.

3. Application in the Real System

Taking the average of the two best τ_C values

$$\tau_C = \frac{\tau_{C1} + \tau_{C2}}{2} = 125$$

We update the coefficients of our PID and then apply the controller to each input to ensure that each control influences the system as a whole.

In Figure 11, the Set Point chosen is 0.4. Intuitively, we can realize that the accommodation time is a bit long. However, it is worth remembering that the inlet flow rate in both valves is very low.

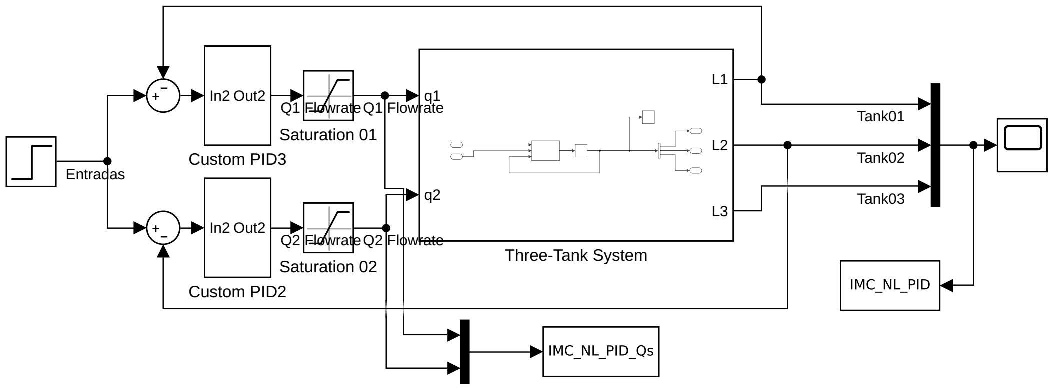


Figure 10: PID by Internal Model Control Applied to a Non-Linear System

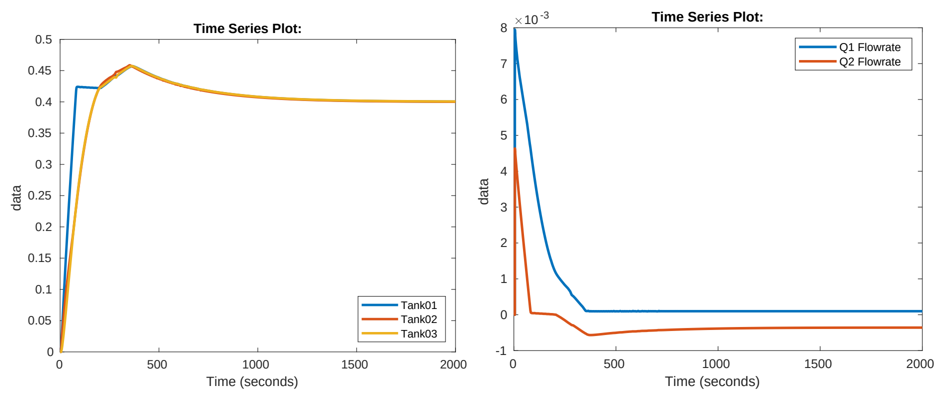


Figure 11: States of a Non-Linear Controlled System for a Set Point equals to 0.4

3.2 Online Controller Tuning Method

The Method used was the Ziegler Nichols Step Test Method.

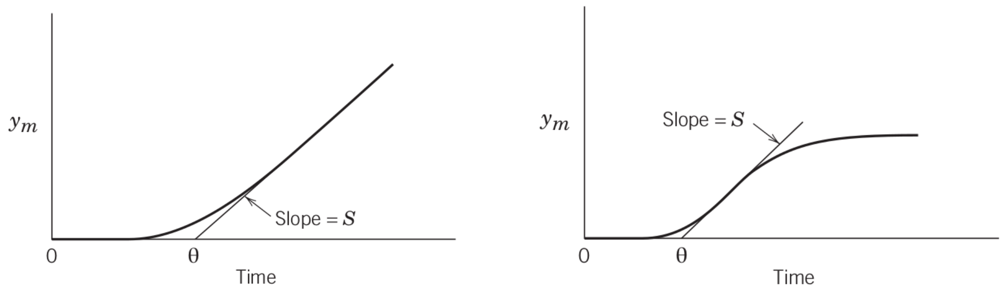


Figure 12: Step Test Method

By that method, after applying the step into the non-linear system, we analyze the output in order to find the values K and τ so that we can approximate (or induce) a model according to Fig XXX.

$$K = \frac{f(\infty) - f(0)}{STEP_{steady}} \quad and \quad \tau = 0.632 \cdot t_{settling}$$

Checking the Table 1, is easy to see that our non-linear system can be approximated to a first order transfer function, as it is shown in the A Case of the Figure 17.

$$G_{ZieglerNichols} = \frac{K}{\tau s + 1}$$

And doing so, we can find the Proportional and the Integrative of the controller, as a function of an variable τ_C. However, for the Online Method, we can see that its control pattern is somewhat

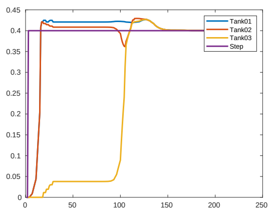
Table 1: Step Response

	Tank 01	Tank 02
Step	$4 \cdot 10^{-5}$	$4 \cdot 10^{-5}$
Y_∞	0.5475	0.2862
K	13387	7155
τ	410.274	371.58
τ_C	-	-

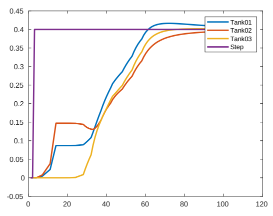
independent of τ_C. This is because, above all, we approach a system of 3 poles and 2 zeros by a system of first order with only 1 pole. In addition, the K values found are extremely large, and therefore, the system is a little more reactive.

If the Transfer Function used to approximate the nonlinear system was a bit more faithful to the real one, we could decrease or increase the τ_C to increase or decrease the Controller's force, since it is inversely proportional.

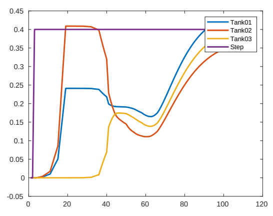
But in fact, this result was already expected, because the method of Ziegler Nichols is already quite archaic.



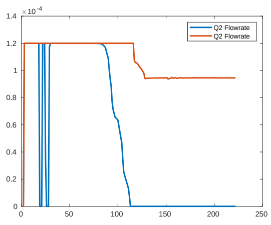
(a) Tank Level $\tau_C =$ 100



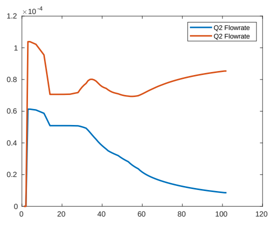
(b) Tank Level $\tau_C =$ 10^5



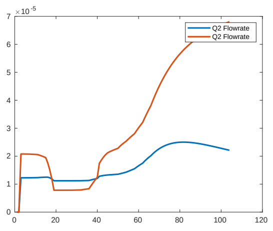
(c) Tank Level $\tau_C =$ 10^8



(d) PID Signal $\tau_C =$ 100



(e) PID Signal $\tau_C =$ 10^5



(f) PID Signal $\tau_C =$ 10^8

Figure 13: Levels and Signals for the Online Controlling Method

4 Genetic Algorithms Applied to Controlling Methods

4.1 Evolutionary Theory

Charles Darwin, the father of the Evolutionary Theory, used to say: "The better an individual adapt to their environment, the greater their chance of survival and generate descendants". So that, all the computational bases used in Genetic Algorithms arises as an analogy, or a replication, of this phrase.

Therefore, the first step is to generate a initial chromosomes population, where each chromosome is a data structure that represents a possible solution for the problem. This population is evaluated with the **fitness function**, so that we are capable of checking an aptitude grade for each one. The greater grades chromosomes are chosen to be the parents of the next new generation of chromosomes. The chosen ones pass through a **mutation** and a **crossing-over** operation from where the sons population is going to “be born”.

This process will keep happening until a good model, for our problem in analyse, be found. Notice that what we have done, in synthesis, is only describing the Neo-Darwinism with a computational language.

4.1.1 Optimization

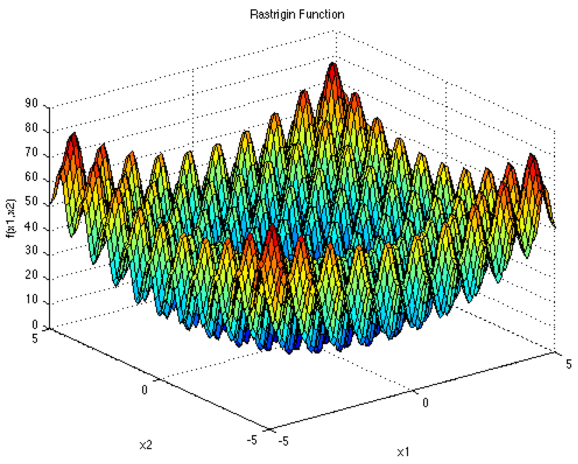


Figure 14: Rastrigin’s function

In a mathematical way, what we are trying to achieve is exactly optimization. However, would not be better to use a classical optimization method instead of this kind of directed stochastic method?

When the problem we desire to optimize is a function that has more than one different peaks and valleys, as Figure 14, would be necessary to analyze all the sample space to find and distinguish the locals minimum and maximum values from the global ones. Thus, is easy to realise that would be very computationally costly.

Meantime, the Genetic Algorithms, because of their stochastic characteristics, form the initial population from random chromosomes. In doing so, it begins by mapping values along all over the sample space. After repeating a couple of times the evolutionary cycle, the values closer to the global maximum, or minimum, starts to converge and, so, pass their "genes" to the next generations. With this, the new "individuals" are going to "be born" even closer to the optimal value, until achieve it.

Therefore, the Evolutionary Computing can optimize the function faster than the usual methods, mainly, because it does not need to check all the inputs availables.

4.2 Tunning PID with Genetic Algorithms

4.3 Matlab Code

```
function [J] = pid_optim_FIT1(x)

Kp = x(1);
Ki = x(2);
%Kd = x(3);

dt = 1 ;
t = 0:dt:500;

set_param("GA/Custom PID3/Kp", 'Gain', num2str(Kp));
set_param("GA/Custom PID3/Ki", 'Gain', num2str(Ki));
%set_param("GA/Custom PID3/Kd", 'Gain', num2str(Kd));
[~, ~, y] = sim('GA', t, [], [t', 0.4*ones(numel(t),1)]);
% u = y(:,4);
y = y(:,1:3);
e = 1 - y;

plot(t,y(:,1));

J = mean(e(:,1).^2) ; %+ mean(u.^2)
```

At first, we define a vector of inputs that is going to be our vector of PID values. Than we set Kp and Ki to receive that vector values.

As the Genetic Algorithm works simulating different combinations of values, applying into the function and checking the cost function, we define a number 't' of iterations and, for each one, we set the new parameters into the Simulink PID, in Figure 15, and next we calculate the error

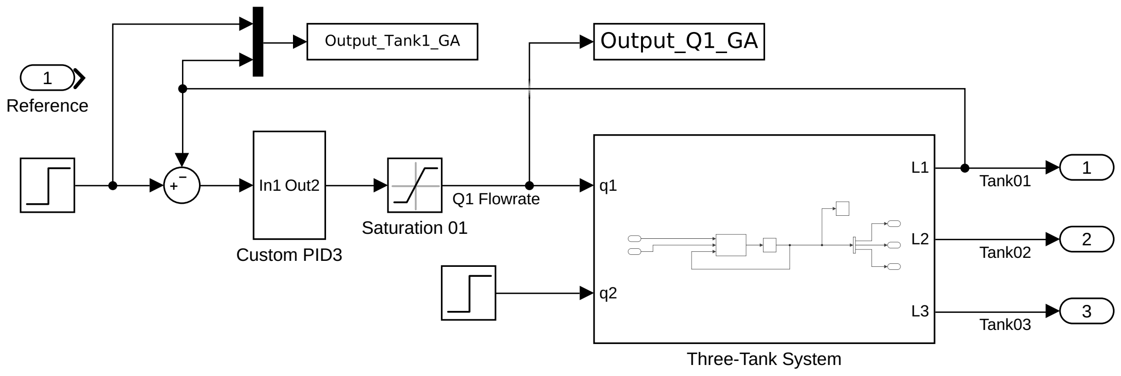
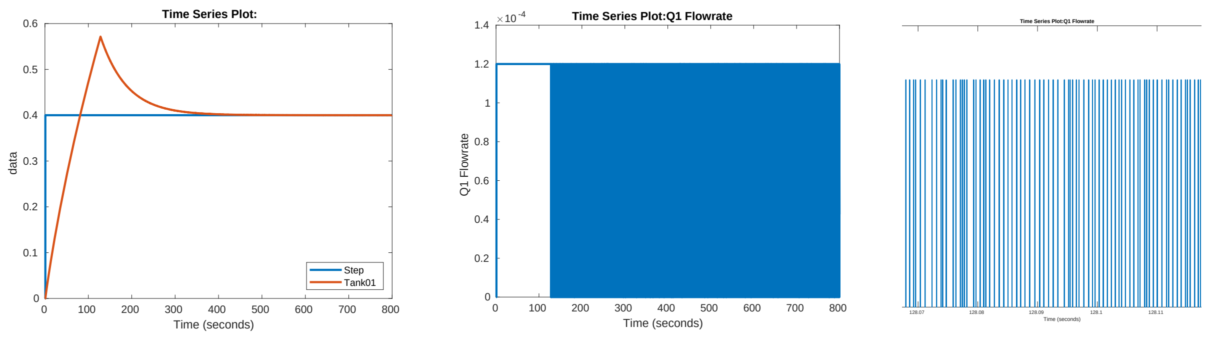


Figure 15: The Simulink Model Tunned by the Genetic Algorithm

as the difference between the Set-point value and the output found for the current configuration.

4.4 Simulation

By the Matlab's Optimasation Toolbox, is possible to simulate our function with all the Genetic Algorithm conditions. Is possible to set the range of values that we want for our parents, and the methods used to **mutating**, **crossing-over**, **fitness scaling**, **migrating**, and more...



(a) Level of Tank 1 (b) PID Signal (c) Zoom of PID Signal

Figure 16: Genetic Algorithm Approach

I just used the default configuration of the Toolbox: 'Constraint Dependent' as Mutation and Crossing-over function; and 'Stochastic Uniform' as Selection Function.

The biggest problem of the Genetic Method, is because is very time-consuming. I left the computer two hours running, with $t = 500$, for achieving only 2 generations (check Table 2).

In Table 2, we can see that despite the two hours, we have managed to reduce the average of the error. So, theoretically, it was converging to a good controller.

However, when we see the PID Signal and Output graph in Figure 16, we can verify that the output is not actually the worst, but the PID Signal is absurdly divergent. In fact, that was the reason why the simulation was so costly computationally.

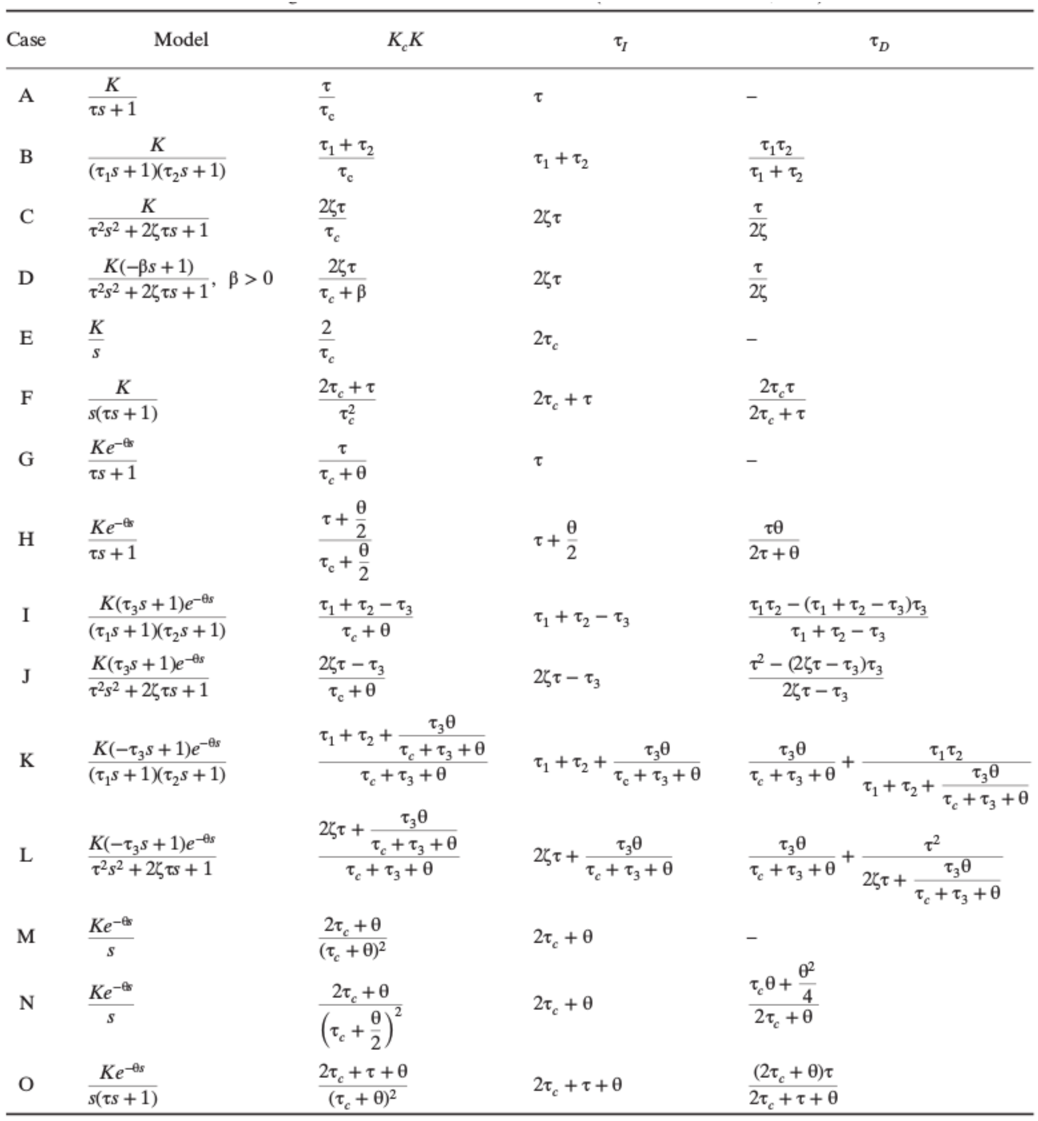
Table 2: Genetic Algorithm's Generations

Generation	Func-count	Best f(x)	Mean f(x)	Stall Generations
1	100	0.371	0.3765	0
2	200	0.3706	0.3759	0

The values of Kp and Ki for that result are $K_P = 408.6516$ and $K_I = 6.7017$.

5 Attachment

5.1 IMC Controller Settings for Parallel-Form PID Controller (Chien and Fruehauf, 1990)



Case	Model	K_cK	τ_I	τ_D
A	$\frac{K}{\tau s+1}$	$\frac{\tau}{\tau_c}$	τ	–
B	$\frac{K}{(\tau_1 s+1)(\tau_2 s+1)}$	$\frac{\tau_1+\tau_2}{\tau_c}$	$\tau_1+\tau_2$	$\frac{\tau_1\tau_2}{\tau_1+\tau_2}$
C	$\frac{K}{\tau^2s^2+2\zeta\tau s+1}$	$\frac{2\zeta\tau}{\tau_c}$	$2\zeta\tau$	$\frac{\tau}{2\zeta}$
D	$\frac{K(-\beta s+1)}{\tau^2s^2+2\zeta\tau s+1},\ \beta>0$	$\frac{2\zeta\tau}{\tau_c+\beta}$	$2\zeta\tau$	$\frac{\tau}{2\zeta}$
E	$\frac{K}{s}$	$\frac{2}{\tau_c}$	$2\tau_c$	–
F	$\frac{K}{s(\tau s+1)}$	$\frac{2\tau_c+\tau}{\tau_c^2}$	$2\tau_c+\tau$	$\frac{2\tau_c\tau}{2\tau_c+\tau}$
G	$\frac{Ke^{-\theta s}}{\tau s+1}$	$\frac{\tau}{\tau_c+\theta}$	τ	–
H	$\frac{Ke^{-\theta s}}{\tau s+1}$	$\frac{\tau+\frac{\theta}{2}}{\tau_c+\frac{\theta}{2}}$	$\tau+\frac{\theta}{2}$	$\frac{\tau\theta}{2\tau+\theta}$
I	$\frac{K(\tau_3 s+1)e^{-\theta s}}{(\tau_1 s+1)(\tau_2 s+1)}$	$\frac{\tau_1+\tau_2-\tau_3}{\tau_c+\theta}$	$\tau_1+\tau_2-\tau_3$	$\frac{\tau_1\tau_2-(\tau_1+\tau_2-\tau_3)\tau_3}{\tau_1+\tau_2-\tau_3}$
J	$\frac{K(\tau_3 s+1)e^{-\theta s}}{\tau^2s^2+2\zeta\tau s+1}$	$\frac{2\zeta\tau-\tau_3}{\tau_c+\theta}$	$2\zeta\tau-\tau_3$	$\frac{\tau^2-(2\zeta\tau-\tau_3)\tau_3}{2\zeta\tau-\tau_3}$
K	$\frac{K(-\tau_3 s+1)e^{-\theta s}}{(\tau_1 s+1)(\tau_2 s+1)}$	$\frac{\tau_1+\tau_2+\frac{\tau_3\theta}{\tau_c+\tau_3+\theta}}{\tau_c+\tau_3+\theta}$	$\tau_1+\tau_2+\frac{\tau_3\theta}{\tau_c+\tau_3+\theta}$	$\frac{\tau_3\theta}{\tau_c+\tau_3+\theta}+\frac{\tau_1\tau_2}{\tau_1+\tau_2+\frac{\tau_3\theta}{\tau_c+\tau_3+\theta}}$
L	$\frac{K(-\tau_3 s+1)e^{-\theta s}}{\tau^2s^2+2\zeta\tau s+1}$	$\frac{2\zeta\tau+\frac{\tau_3\theta}{\tau_c+\tau_3+\theta}}{\tau_c+\tau_3+\theta}$	$2\zeta\tau+\frac{\tau_3\theta}{\tau_c+\tau_3+\theta}$	$\frac{\tau_3\theta}{\tau_c+\tau_3+\theta}+\frac{\tau^2}{2\zeta\tau+\frac{\tau_3\theta}{\tau_c+\tau_3+\theta}}$
M	$\frac{Ke^{-\theta s}}{s}$	$\frac{2\tau_c+\theta}{(\tau_c+\theta)^2}$	$2\tau_c+\theta$	–
N	$\frac{Ke^{-\theta s}}{s}$	$\frac{2\tau_c+\theta}{\left(\tau_c+\frac{\theta}{2}\right)^2}$	$2\tau_c+\theta$	$\frac{\tau_c\theta+\frac{\theta^2}{4}}{2\tau_c+\theta}$
O	$\frac{Ke^{-\theta s}}{s(\tau s+1)}$	$\frac{2\tau_c+\tau+\theta}{(\tau_c+\theta)^2}$	$2\tau_c+\tau+\theta$	$\frac{(2\tau_c+\theta)\tau}{2\tau_c+\tau+\theta}$

Figure 17: IMC Controller Settings for Parallel-Form PID Controller (Chien and Fruehauf, 1990)