Controlling pH neutralization

Salvatore Milite

July 16, 2020

A common problem in various chemical engineering task is to quickly bring a solution back to a certain pH value. It often happens in industrial application to have to neutralize large volumes of acids or strong bases in controlled stirred tank reactors (CSTRs).

Generally a large number of industrial wastes produce extremely alkaline waters, bringing them back to a pH around 7 (therefore neutral) has the double benefit of making the water less toxic and promoting the activity of those bacteria that are used for chemical demolition of waste (which would otherwise die in too extreme conditions).

In literature the problem of controlling this neutralization is known to be rather difficult mainly because of its strong non-linearity, especially as regards the oscillations around the reference value, and several different control systems have been proposed to improve its performance.

For this exam project we will first derive a model for pH neutralization in a very simple reactor of strong-base tritation and then we will try to look the performance of a simple PID controller, a neural network based predictive controller and fuzzy-logic controller. The implementation is presented in Matlab-Simulink.

Derivation of the plant

There are several ways to build such a reactor and the attention to the detail of mechanics and dynamics of each component involved in the process can be made very complex. Also from the chemical point of view there are different ways in which this neutralization reaction can be carried out with respect also to the number and type of acids that can be used in the reaction. In our case we will deal with a very simple problem, in which the process is shown in figure 1.

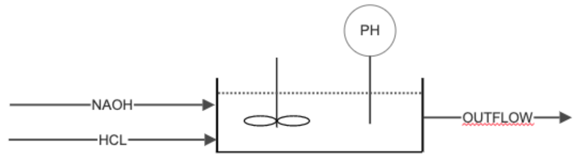

Figure 1: Model of the process

As regards the reaction we have a very simple neutralization of a strong base with a strong acid. In formulas

$$\mathrm{NAOH} + \mathrm{HCL} \rightarrow \mathrm{NACL} + \mathrm{H2O}$$

To have a complete model of the plant we need equations for the dynamics of the system (which basically tells us how the concentrations change with time) and a formula for calculating the pH from these concentrations.

To have a complete model of the plant we need equations for the dynamics of the system (which basically tells us how the concentrations change with time) and a formula for calculating the pH from these concentrations. As for the dynamic model, it can be easily obtained using the principle of material balances [2]:

$$V\frac{du_a}{dt} = F_aC_a - (F_a + F_b)\, u_a$$

$$V\frac{du_b}{dt} = F_bC_b - (F_a + F_b)\, u_b$$

where C_a is the concentration of the inflow acid, C_b the concentration of the inflow base, F_a is the flow of acid, F_b the flow of the base, u_a is the concentration of NA^+ in the tank, is the concentration of CL^- in the tank and V is the total volume.

Now that we have a way to calculate ionic concentration for a given t we need to put this in the context of pH calculation [3]. Remember pH is defined as:

$$\mathrm{pH} = -\log_{10}\left[\mathrm{H}^+\right]$$

Then equilibrium for water is:

$$\mathrm{k_w} = \left[\mathrm{H}^+\right]\left[\mathrm{OH}^-\right]$$

The electroneutrality relation:

$$\left[\mathrm{Na}^+\right] + \left[\mathrm{H}^+\right] = \left[\mathrm{Cl}^-\right] + \left[\mathrm{OH}^-\right]$$

So defining the difference $u_a - u_b = \Delta$ and substituting into the equations we obtain:

$$[\mathrm{H}^+]^2 - \Delta[\mathrm{H}^+] - \mathrm{K_w} = 0$$

which solution is:

$$\mathrm{pH} = -\log_{10}\left(\frac{\mathrm{x}}{2} + \sqrt{\frac{\mathrm{x}^2}{4} + \mathrm{K_w}}\right)$$

Control and verification

The final implementation is done in Simulink, the version with the classical PID is in figure 2. It consist in a submodule for getting the difference in concentration between the two ions in the tank and a pH measurment system. Just note that the inflow of both acid and base is regulated by a valve (with range 0-100%) that inject a percentage of the max flow.

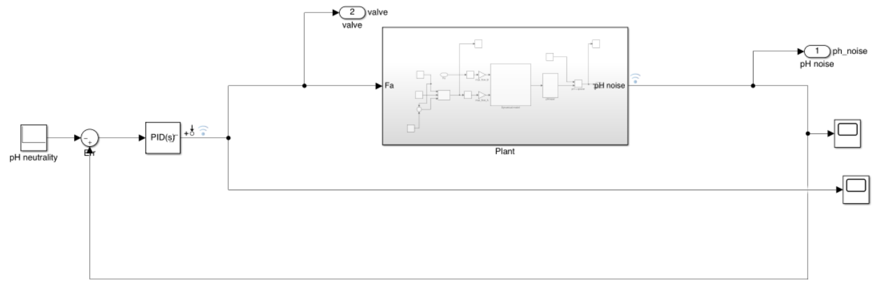

Figure 2: Closed-loop PID control

The neural system uses a simple one layer neural net that predicts the output of the plant and another that acts as a contoller, implemented in the Matlab Deep Learning toolbox.

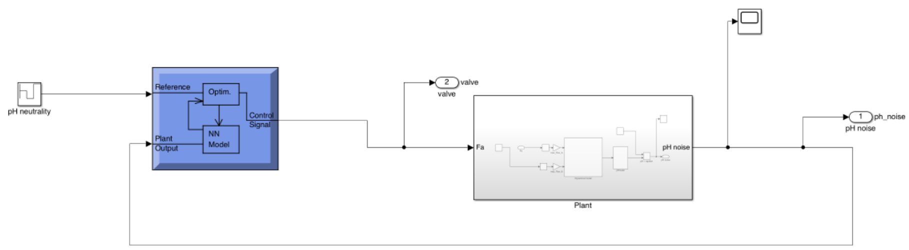

Figure 3: Neural network based controller

We have also included a simple fuzzy-controller for comparison which seems to be the most used in literature.

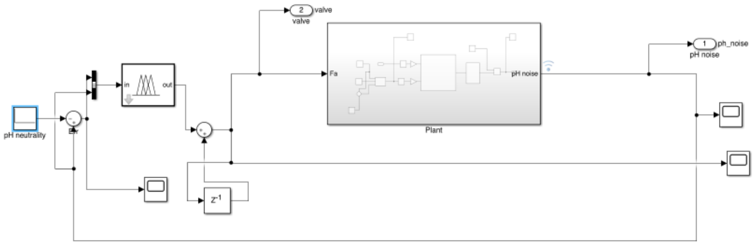

Figure 4: Fuzzy logic controller

However all of these three solutions are far from the state of art.

Simulations

Three inputs

- **Closed-Open Valve** We start from a closed valve and then we open it
- **Closed-Open Valve w/t perturbation** same as before, but we add a perturbation in the middle
- **Servo Problem** we change our PH reference during the simulation

  Parameters for the experiments are in the implementation files.

Apart from the performances we also verify those properties

- pH should never be higher then 8.5 after an initial delay (+-2 than the reference in the servo problem, 9 for the opening valve) $\mathbf{G}_{[delay,final]}\,(pH(t) \leq upper_bound)$
- pH should never be lower then 5.5 (+-2 than the reference in the servo problem, 5 for the opening valve) $\mathbf{G}_{[delay,final]}\,(pH(t) \geq lower_bound)$

The classic PID is tuned manually starting from from lowering the proportional gain while taking the other measures to 0 and then tweaking Kd, finally adjust with Ki.

The neural controller is taken from the Matlab Deep Learning Toolbox and tuned with 2000 samples randomly generated from the plant. The neural part is actually predicting the plant response whether the next move is chosen minimizing this formula .

$$J = \sum_{j=N_1}^{N_2} \left(y_r(t+j) - y_m(t+j)\right)^2 + \rho \sum_{j=1}^{N_u} \left(u'(t+j-1) - u'(t+j-2)\right)^2$$

Where N_1, N_2 and N_u are the horizons over which the tracking error and the control increments are evaluated. u' is the tentative response, ρ is the contribution of the sum of squares of the increments in the formula and y_r and y_m are respectively the reference signal and the network response. More information at [4] or on the Matlab Deep Learning ToolBox documentation

The fuzzy controller has those discourse variables for inputs and for outputs, we have two input discourse variable one (with 3 states) describing where we are in the pH range (0-5,5-9,9-14) and one (with 9 states) describing the error we are receiving (from negative negative error to positive positive error). The output (9 states) describes the strength of opening of the valve. For the rules chekout the file .fip into the implementation.

For the first problem, i.e. simply contrasting a constant flow of base starting from a closed valve results are in figure 5.

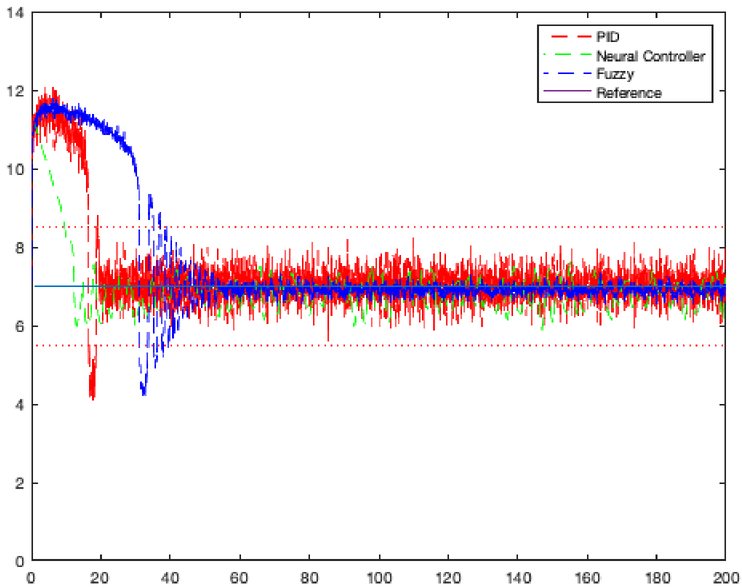

Figure 5: Constant flow of base, starting with the acid valve closed

As we can see the fuzzy controller is more stable but express a particularly oscillatory behavior in the first seconds. The neural controller achieves the fastest response while oscillating a lot , the PID is somewhat similar (if we give a certain delay for the opening valve however all three controllers are able to maintain the reference position)
Things start to get worse when we add a consistent perturbation, such as suddenly close of 40% the inflow valve.

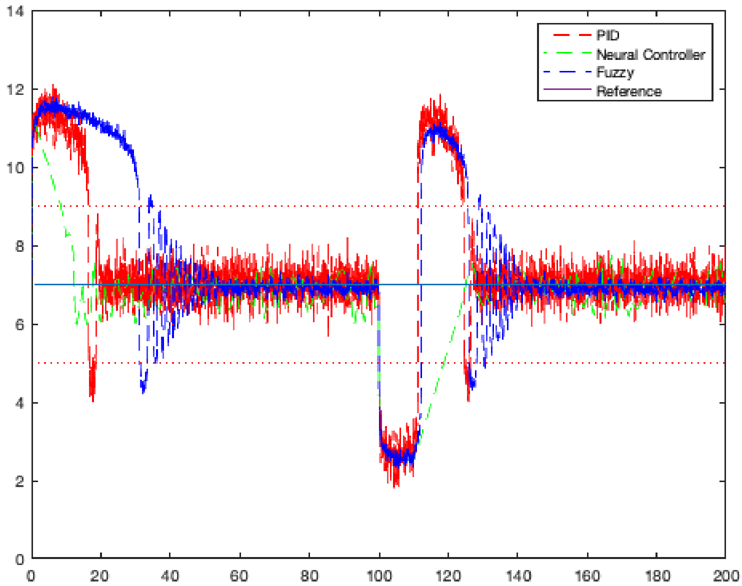

Figure 6: Constant flow of base, with a closing of 40% of the base valve at time 100

Here no controller is able to stay in decent limits, however the neural one is able to recover without oscillations, while the others have an inversion in the pH (to be fair the difference in concentration is kinda big and stays for 10 second, it is realistically very hard to not reach extremely low pHs with these controllers that are indeed constructed to work better around 7).

To conclude if we change the reference point the controller are able to adapt with the same error as before. Even though the neural one seems to underestimate a bit the second reference point.

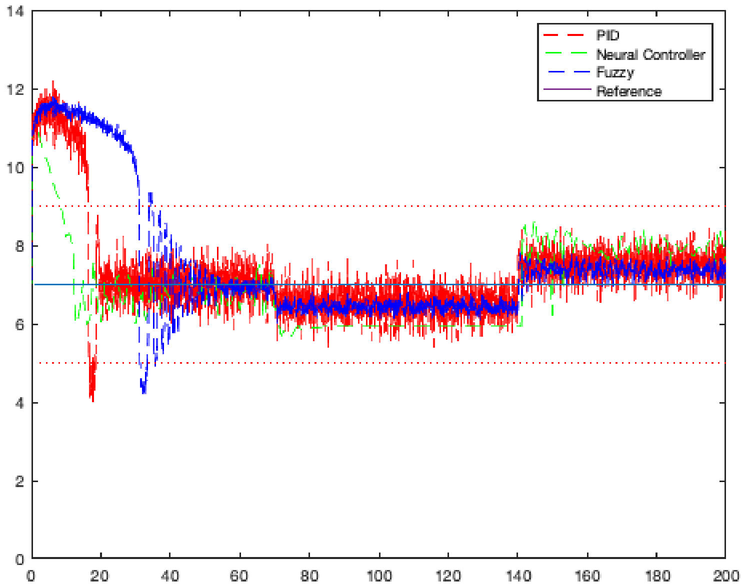

Figure 7: Constant base, varying reference

To conclude this is just a merely demonstration of different very basic controller systems dealing with a non linear model. A lot of things need to be improved, especially the oscillation which are extremely hard to control around 7. There are different directions to explore with more time and skills, for sure a more complex neural network the the simple 2 layers fully connected used here can improve a lot the results, also very popular are also fuzzy controllers with a third variable coding the steepness of the changing in pH [1].

Regarding model checking is pretty obvious that the model in not that robust and it is easy to find counter-examples, what I have implemented is instead trying to search the maximum base valve closing that happen to be neutralizable in the model specification (circa 2% around the steady state value, so if the previous opening is 50% of the base valve the controllers are able to stay in the limitis for a change in 48-52 %).

References

[1] Maria J Fuente, C Robles, O Casado, S Syafiie, and Fernando Tadeo. Fuzzy control of a neutralization process. *Engineering Applications of Artificial Intelligence*, 19(8):905–914, 2006.

[2] David M Himmelblau. Basic principles and calculations in chemical engineering. 1982.

[3] Ralph H Petrucci, Clarence M Cunningham, and Thomas E Moore. *General chemistry*. Macmillan New York, 1989.

[4] Donald Soloway and Pamela J Haley. Neural generalized predictive control. In *Proceedings of the 1996 IEEE International Symposium on Intelligent Control*, pages 277–282. IEEE, 1996.