2. System Identification

Gray-Box Modelling

Gray-Box modelling relies heavily on assessing the physical processes influencing the dynamics of the concerned system. Once these processes are identified, the resulting differential equation is constructed. Most of these processes could be modelled well using a first-order or a second-order differential equation. If such is the case, a relatively smaller number of parameters requires to be estimated, as compared to the parameters involved in the actual differential equation. This section proceeds to explain the justification for the choice of the parameter estimation method, each of the model used and how they are extended to MIMO system analysis.

Motivation for Parameter Estimation

Let us utilize the basic 1st order differential equation used to model single heater-sensor system. Whenever the heater is actuated, it is assumed that some delay is observed until the sensor registers some temperature change. This delay when appended to the 1st order model results in FOPDT as shown in (2.1).

$$\tau_p \frac{dy(t)}{dt} = -y(t) + K_p u\left(t - \theta_p\right) \tag{2.1}$$

We had two ways to handle the equations (2.1) in MATLAB, either numerically, or obtain the analytical solution and substitute the values of input u, and other parameters. Both of them were tested against the time elapsed. The numerical solver, i.e. ode23 took 0.4% of the time engulfed by the analytical solver (see file : `Analytic_run.m`). This is quite an expected observation since MATLAB specializes in its numerical toolbox. For the numerical solver, the resulting difference equation using backward difference operator is given by (2.2) as follows.

$$y_j = e^{\frac{-\Delta t}{\tau_p}}\left(y_{j-1} - y_0\right) + \left(1 - e^{\frac{-\Delta t}{\tau_p}}\right) K_p \left(u_{j-\theta_p-1} - u_0\right) + y_0 \tag{2.2}$$

Any attempt to produce an ARX model and therefore utilize quadratic optimization to estimate the parameters τ_p, K_p and θ_p would be trivial due to the non-linearity induced by 2nd term on RHS of (2.2). Therefore either unconstrained non-linear optimization or random search algorithm, for e.g. genetic algorithm must be prescribed. We opted for the former (MATLAB equivalent *fmincon* with default interior point algorithm) with the cost function defined in equation (2.3).

$$\min_{\tau_p, \theta_p, K_p} \sum_{i=1}^{N} (\hat{y_j} - y_j)^2 \tag{2.3}$$

with $\hat{y}_j$ being the training timeseries dataset. As expected, multiple local minima were obtained for different initial parameter values and optimizer settings. Certain heuristics were utilized with each model to obtain "*hunch*" about the parameters, discovered through experimentation, and are discussed for each model separately.

FOPDT

Since we have two heater-sensor pair in a close proximity to each other, there is bound to be some exchange of energy. Let us assume this exchange to be proportional to the temperature

of the other heater, j. Additionally, our room temperature is not 0. This also has to be appended into the 1st order dynamics of (2.1).

$$\text{Heat Exchange}_i = \alpha_i(T_\infty - \text{T}_j) \tag{2.4}$$

(2.5) shows the 1st order system of differential equation with delay and heat exchange incorporated.

$$\begin{aligned} \tau_{p1}\frac{dT_1(t)}{dt} &= (T_\infty - T_1(t)) + K_{p1}u_1\left(t - \theta_{p1}\right) + \alpha_1(T_\infty - T_2(t)) \\ \tau_{p2}\frac{dT_2(t)}{dt} &= (T_\infty - T_2(t)) + K_{p2}u_2\left(t - \theta_{p2}\right) + \alpha_2(T_\infty - T_1(t)) \end{aligned} \tag{2.5}$$

Random values of parameters were selected as the initial guess for the optimizer. It often happened that the initial optimal parameters obtained were not able to capture the dynamics of the training set, particularly the heat exchange. This was deduced, for e.g., from T_2 response in the regions where $u_2 = 0$ and $u_1 \neq 0$ (see for e.g. Figure 2.1, 5000s-5500s). Before blaming the linear relationship (2.4), we resorted to figuring out a better local minima, in particular, the value of α_i. The initial value for the said parameter was varied to a more "*suitable*" one while keeping the other to the local minima value obtained. The result was an acute trace of the training set. For this reason, we would refrain from providing the training set VAF and figures, as evident in Table (2.4). A curious reader could find the training figures and conclusion in the directed repository [1]. The verification test is shown in figure (2.1) along with the response of the system with an initial guess of parameters.

Parameters	**Values**
K_p	0.437716392601807, 0.283178262143886
τ_p	89.8066689603212, 75.8833530400405
θ_p	38.00015405405, 46.07928346728
α	-0.505754718092448, -0.578899158592154

Table 2.1: Parameters for FOPDT

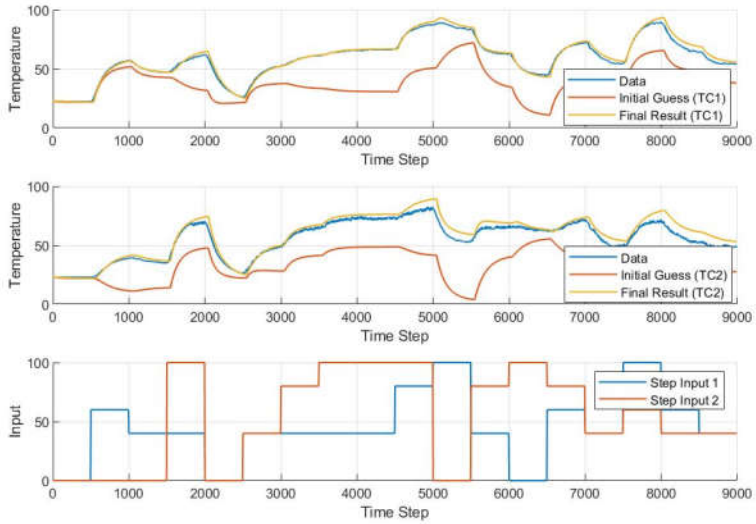

Figure 2.1: Verification test for FOPDT

SOPDT

The standard second order differential equations were modified to embed (2.4) and delay, producing equation (2.6).

$$\begin{aligned}\tau_{s1}^2\frac{d^2T_1(t)}{dt^2} &= -2\tau_{s1}\xi_1\frac{dT_1(t)}{dt} + (T_\infty - T_1(t)) + K_{p1}u_1\left(t-\theta_{p1}\right) + \alpha_1(T_\infty - T_2(t)) \\ \tau_{s2}^2\frac{d^2T_2(t)}{dt^2} &= -2\tau_{s2}\xi_2\frac{dT_2(t)}{dt} + (T_\infty - T_2(t)) + K_{p2}u_2\left(t-\theta_{p2}\right) + \alpha_2(T_\infty - T_1(t))\end{aligned} \tag{2.6}$$

Similar to procedure described for FOPDT, the initial parameter values were carefully selected such as to land in an ideal local minima. The SOPDT model is expected to give a better performance as compared to FOPDT, therefore we utilized the state space model of the equation (2.6) with the states and state transition matrices gives by equation (2.7).

$$x_1 = T_\infty - T_1, \qquad x_2 = T_\infty - T_2, \qquad x_3 = \frac{dT_1}{dt}, \qquad x_4 = \frac{dT_2}{dt}$$

$$\dot{x}(t) = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ \frac{1}{\tau_{s1}^2} & \frac{\alpha_1}{\tau_{s1}^2} & \frac{-2\xi_1}{\tau_{s1}} & 0 \\ \frac{\alpha_2}{\tau_{s2}^2} & \frac{1}{\tau_{s2}^2} & 0 & \frac{-2\xi_2}{\tau_{s2}} \end{bmatrix} x(t) + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{K_{p1}}{\tau_{s1}^2} & 0 \\ 0 & \frac{K_{p2}}{\tau_{s2}^2} \end{bmatrix} \begin{bmatrix} u_1(t-\theta_{p1}) \\ u_2(t-\theta_{p2}) \end{bmatrix} \tag{2.7}$$

Figure 2.2 shows the result on the verification dataset using the derived state space along with the following table stating the optimized values (see file : `statespace.m`). The output is taken as the x_1 and x_2 subtracted out of the ambient temperature.

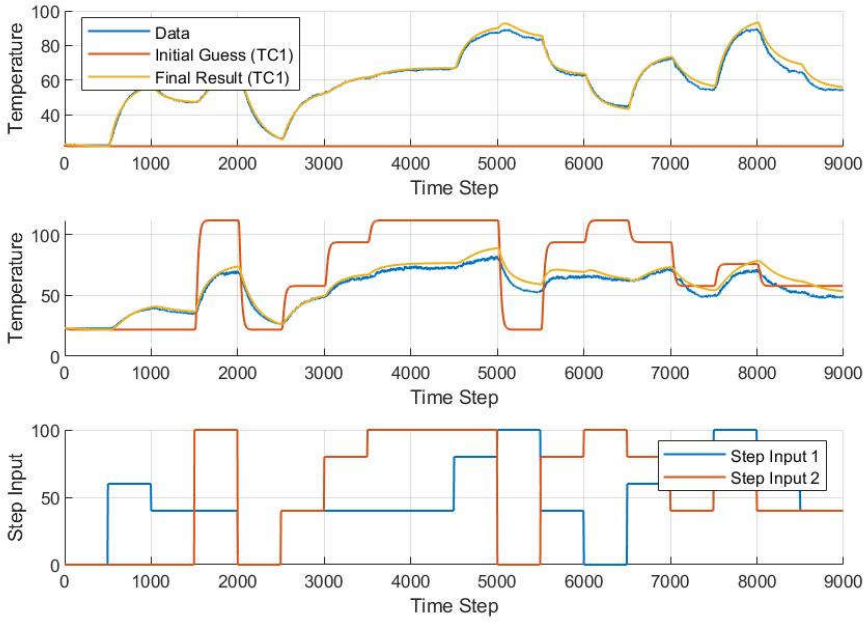

Figure 2.2: Verification test for SOPDT

Parameters	Values
K_p	0.444017081121182, 0.289902615386266
τ_p	6.01205134161293, 24.7626283432986
θ_p	7.78113693111332, 1.97777335094820
ξ	7.78113693111332, 1.97777335094820
α	-0.499258984188171, -0.572779968489432

Table 2.2: Parameters for FOPDT

Physical Non-Linear Model

The physical model is based on heat and mass exchange equations, taking into account various attributes of heater and sensor.The lumped parameter model assumes that the heaters (TH1 and TH2) and temperature sensors (TC1 and TC2) have a uniform temperature. The equation dictating the dynamics of heater and sensor is shown below. The inter-heater radiative heat exchange is ignored.

$$\begin{aligned} mc_p\frac{dT_{H1}}{dt} &= UA\left(T_\infty - T_{H1}\right) + \epsilon\sigma A\left(T_\infty^4 - T_{H1}^4\right) + U_sA_s\left(T_{H2} - T_{H1}\right) + \alpha_1 Q_1 \\ mc_p\frac{dT_{H2}}{dt} &= UA\left(T_\infty - T_{H2}\right) + \epsilon\sigma A\left(T_\infty^4 - T_{H2}^4\right) - U_sA_s\left(T_{H2} - T_{H1}\right) + \alpha_2 Q_2 \end{aligned} \tag{2.8}$$

$$\begin{aligned} \tau_c\frac{dT_{C1}}{dt} &= T_{H1} - T_{C1} \\ \tau_c\frac{dT_{C2}}{dt} &= T_{H2} - T_{C2} \end{aligned}$$

A proper account of derivation could be found on [2] and a brief insight about assumptions evolving from FEA of heat sink could be traced to [4]. Furthermore, all the parameter along with suggested range of U, α_i, τ_c and U_s are documented in the stated reference. This range was utilized for constrained non-linear optimization, as it often turned out that without the constraints, the algorithm crashed for certain initialization. The pre-run for training did not fit the set well. It turned out that certain given ranges (for e.g. α_i) had to be enlarged, the extent of which was determined empirically. The causes might stem from imprecise standardization of hardware as is often the case with mass sensor manufacturing.

$$\begin{aligned} 4.4 \leq & U \leq 4.6 \\ 1\times10^{-4} \leq & \alpha_1 \leq 0.0132 \\ 1.2\times10^{-3} \leq & \alpha_2 \leq 6.6\times10^{-3} \\ 21.2 \leq & \tau_c \leq 23.3 \\ 20.6 \leq & U_s \leq 100 \end{aligned} \tag{2.9}$$

The optimized value of each parameter is shown in table 2.3.

Parameters	Values
U	4.40000000013082
α	0.00983427953031610, 0.00659999999978003
τ_c	23.2999999995173
U_s	35.4999246707500

Table 2.3: Parameters for PHY

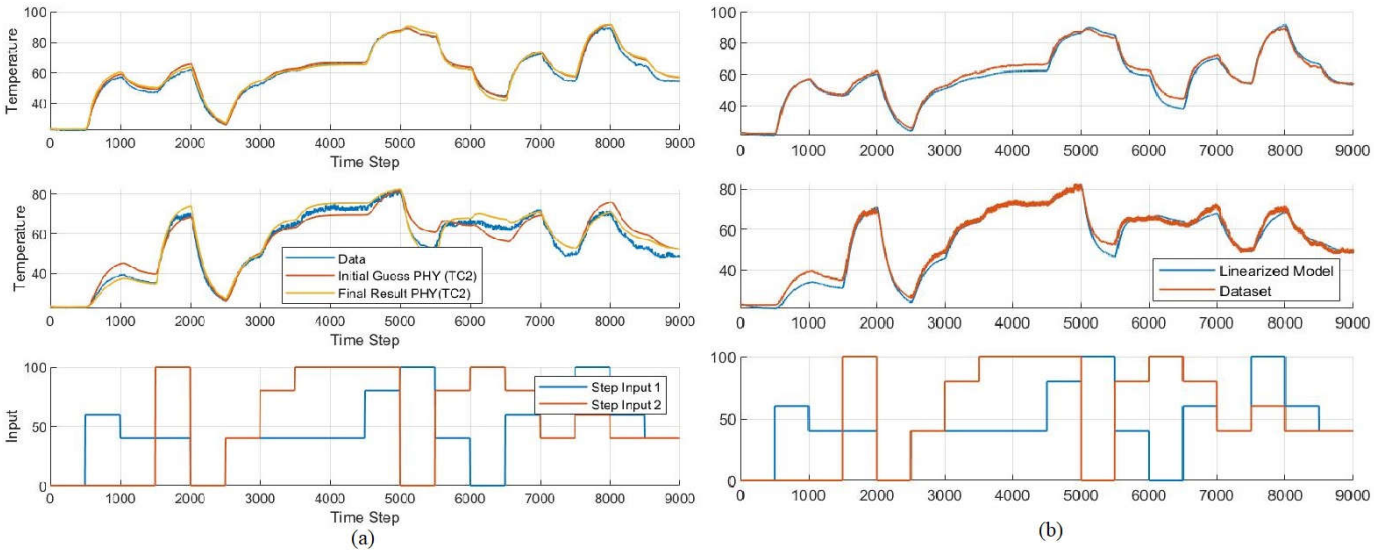

Figure 2.3: Verification test (a) Non-Linear model (b) Linearized model

The physical model had the least number of parameters to be determined and still resulted in the largest VAF. But this alone is not enough to conclusively state that the latter is the best out of all the grey box models concerned. Furthermore, if one were to use this model to design a linear controller, it would also have to be linearized around a working temperature which would, in turn, affect the VAF. To make this claim quantitative and brief, (2.10) and Figure 2.3(b) shows the results of linearization about 50°C.

$$\dot{x}(t) = \begin{bmatrix} (-\frac{UA}{mc_p} - \frac{4\epsilon\sigma A}{mc_p}\bar{x}_1^3 - \frac{U_sA_s}{mc_p}) & \frac{U_sA_s}{mc_p} & 0 & 0 \\ \frac{U_sA_s}{mc_p} & (-\frac{UA}{mc_p} - \frac{4\epsilon\sigma A}{mc_p}\bar{x}_2^3 - \frac{U_sA_s}{mc_p}) & 0 & 0 \\ \frac{1}{\tau_c} & 0 & \frac{-1}{\tau_c} & 0 \\ 0 & \frac{1}{\tau_c} & 0 & \frac{-1}{\tau_c} \end{bmatrix} x(t) + \begin{bmatrix} \frac{\alpha_1}{mc_p} & 0 \\ 0 & \frac{\alpha_2}{mc_p} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \tag{2.10}$$

with $\bar{x}_1$ and $\bar{x}_2$ being the temperature about which linearization is being performed. The output of the system is x_3 and x_4 from which, certain values are subtracted to obtain the actual temperature (see file : `PHY_lin.m`).

Models	VAF for T_1	VAF for T_2
FOPDT	99.8817	99.3012
SOPDT	99.8981	99.3051
Physical Non-Linear Model	99.9953	99.9931
Physical Linearized Model	99.8298	99.8195

Table 2.4: Verification VAF for different models

Bibliography

[1] S.Razdan A. Srivastava. Temperature control lab repository. URL https://github.com/axisrock1997/TCL-TUDELFT-. 3

[2] John D. Hedengren. Temperature control lab, . URL http://apmonitor.com/pdc/index.php/Main/ArduinoTemperatureControl. 1, 5

[3] John D. Hedengren. Overview of identification and control, . URL http://apmonitor.com/pdc/index.php/Main/HomePage. 1

[4] J. D. Kellyb John D. Hedengren J. Park, R.A. Martina. Benchmark temperature microcontroller for process dynamics and control. *Computers and Chemical Engineering*, 2019. doi: https://doi.org/10.1016/j.compchemeng.2020.106736. 1, 5

[5] M. Verhaegen V. Verdult. *Filtering and System Identification: A Least Squares Approach*. Cambridge University Press, May 28,2007. ISBN 9780521875127. 1, 7