NTNU – Department of Chemical Engineering

Verification and Improvements of the SIMC Method for PI Control

Chriss Grimholt

TKP 4550 – Specialization project
Trondheim, december 2010

Acknowledgments

I would like to thank my supervisor professor Sigurd Skogestad at NTNU for his never-ending support and interest in my work. Whenever I came knocking on his office door needing help or guidance, Sigurd would immediately put his own work aside to direct me in the right direction. For this I am profoundly grateful.

I would also like to thank my friend Stian Forselv for proof-reading this report, which must have been tedious work for someone who has not studied control theory.

Abstract

The tradeoff between robustness and performance has been investigated for the SIMC tuning with the help of Pareto-optimal curves, and was found to be good. Though, for a given robustness, there where some room for improving the performance. This resulted in the presented improved-SIMC method which is empirically designed to give the same tradeoff, but with enhanced performances.

Haugen's proposal to increase the integral action for SIMC has also been investigated. It was found that this method gave good tradeoff for delay dominated and slightly lag dominated processes, but terrible tradeoff for very lag dominated and integrating processes.

SIMC was found to perform less than average for processes of higher order and equal lag constants. This was found to be caused by the half-rules approximation of the process lag constants. Two proposed approximations has been proposed for these types of processes, that gives better results.

The two-step setpoint overshoot method was investigated as an alternative method for attaining the open-loop process model. Inconsistencies in the method was found and corrected, and the method gave good results.

Contents

1. Introduction

Though a proportional-integral (PI) controller has only two tuning parameters, it is surprisingly difficult to find a controller setting that gives a satisfactory response. This is due to the tradeoff between performance and robustness. High performance gives low robustness, and vice versa. Where is the optimal compromise? This is up to the control engineers preference, but should be at some middle ground.

One very successfull tuning rule is the Skogestad's SIMC rule, which uses an open-loop model to derive the controller settings[1]. Haugen reported that this rule gave sluggish disturbance response for lag dominated processes and suggested an increase in the integral action to improve the response[2]. With the use of Pareto-optimal curves this report will show that SIMC gives good tradeoff for most processes, but the performance can be improved. This resulted in the presented improved-SIMC, which is empirically designed to give better performance, but with the same tradeoff. Hagen's increased integral action SIMC, is found to give good performance and tradeoff for delay dominated and slightly lag dominated processes, but gave bad tradeoff for very lag dominated and integrating processes, and is thus not recommended.

At the foundation of the SIMC method for deriving PI controller settings, is the half-rule approximation of a higher order open-loop model to a first-order-plus-time-delay-model (FOPTD). This report shows that the half-rules approximation of lag constants for higher order processes with equal lags, results in a deterioration of the SIMC tuning performance. Two approximations, the half-rule addendum and the lag ratio approximation, is presented to give a better approximation for these processes.

The setpoint overshoot method presented by Shamsuzzoha at al. [3] is a method used for determining a close-to SIMC tuning, from a closed-loop setpoint experiment. This was originally a one-step procedure, which gave the controller settings directly from the experiment. An alternative two-step procedure was also proposed, where the setpoint experiment is used to first approximate the open-loop model and, by applying a tuning rule, the controller tuning is determined. The two-step setpoint overshoot methods is proposed as an alternative method for obtaining the process model for SIMC. This method contained inconsistencies that is corrected in this report. The corrected two-step overshoot method gave good results that is very similar to the one-step overshoot method.

2. Background information

The Skogestad's SIMC is a direct synthesis method that uses an open-loop process model to derive the controller settings. Depending on whether a proportional-integral (PI) controller or a proportional-integral-derivative (PID) controller is desired, the model is reduced to a first-order-plus-time-delay(FOPTD) or a second-order-plus-time-delay respectively. This report will mainly concentrate on PI control and PID control will not be described further.

The SIMC method can be divided into a simple two step procedure:

Step 1: Obtain a FOPTD model if the model is of higher order, use the half-rule to approximate the model as a FOPTD model).

Step 2: Derivation of the model-based controller setting.

These steps will be further explained later in the report. A laplace transform of a FOPTD process model is shown in equation 2.1.

$$g_p(s) = \frac{k_p}{(\tau_p s + 1)} \exp(-\theta s) \tag{2.1}$$

Where $g_p(s)$ is the process transfer function, k_p is the process gain, τ_p is the lag time constant and θ is the delay time. The Laplace transform of a PI controller is shown in equation 2.2.

$$g_c(s) = k_c \left(1 + \frac{1}{\tau_i s}\right) \tag{2.2}$$

Where $g_c(s)$ is the controller transfer function, k_c is the process gain and τ_i is the integral time. In figure 2.1, the conventional feedback control structure used in this report is shown, where r is the setpoint, e is the controller error, u is the manipulated variable, d is the load disturbance and y is the process output. Simulink is used for simulations and the block diagram used is shown in appendix A.

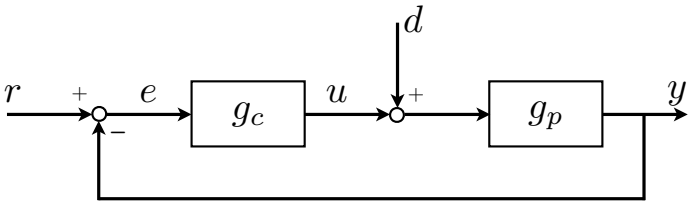


Figure 2.1.: Block diagram of the feedback control system that is used in this report.

2.1. The half-rule

Open-loop experiments does not always give a FOPTD mode, and the model must therefore be reduced to derive the PI controller settings. This can be done by approximating the smaller lag time constants and the inverse responses as time delay. Take the general higher order process model with multiple lag time constants $\tau_{p0,i}$ and negative numerator time constants τ_j^{inv}:

$$g_{p0}(s) = \frac{\prod_j \left(-\tau_{0,j}^{\text{inv}} s + 1\right)}{\prod_i (\tau_{p0,i} s + 1)} \exp(-\theta_0 s) \tag{2.3}$$

The lag time constants $\tau_{p0,i}$ are order by their magnitude and the inverse responce constants $\tau_{0,j}^{\text{inv}}$ are positive. The root of this approximation is the use of a first order Taylor approximation:

$$\exp(-\theta s) \approx 1 - \theta s \qquad \exp(-\theta s) = \tfrac{1}{\exp(\theta s)} \approx \tfrac{1}{(1+\theta s)} \tag{2.4}$$

The inverse response of the general process can by equation 2.4 be approximated as time delay:

$$\left(-\tau_{0,j}^{\text{inv}} s + 1\right) \approx \exp\left(-\tau_{0,j}^{\text{inv}} s\right) \tag{2.5}$$

Similarly, from equation 2.4, a small lag time constant can be approximated as a time delay:

$$\frac{1}{(\tau_{p0,i} s + 1)} \approx \exp(-\tau_{p0,i} s) \tag{2.6}$$

Previously, all except the largest lag time constant where approximated as time delay, but this approximation was to conservative. This is because time delay have a larger negative

effect on performances than a lag constant of equal size. The half-rule compensate for this by only approximating one-half of the second largest time constant as time delay and adding the other half to the retained lag time constant. For summary, the half-rule states that a higher order process can be reduced by the following rules:

$$\tau_p \approx \tau_{p0,1} + \frac{1}{2}\tau_{p0,2} \tag{2.7}$$

$$\theta \approx \theta_0 + \frac{1}{2}\tau_{p0,2} + \sum_{i\geq 3} \tau_{p0,i} + \sum_j \tau_{0,j}^{\text{inv}} \tag{2.8}$$

A process model can also contain positive numerator time constants T_0 as the following process:

$$g_{p0} = \frac{(T_0 s + 1)}{(\tau_{p0} s + 1)} \tag{2.9}$$

For the given process model, it is proposed to cancel out the numerator against a "neighboring" lag constant by the following rules:

$$\frac{(T_0 s + 1)}{(\tau_{p0} s + 1)} \approx \begin{cases} {}^{\tau_0}/_{\tau_{p0}} & \text{if} \quad T_0 \geq \tau_{p0} \geq \tau_c & \text{(Rule T1)} \\ {}^{\tau_0}/_{\theta} & \text{if} \quad T_0 \geq \tau_c \geq \tau_{p0} & \text{(Rule T1a)} \\ 1 & \text{if} \quad \tau_c \geq T_0 \geq \tau_{p0} & \text{(Rule T1b)} \\ {}^{\tau_0}/_{\tau_{p0}} & \text{if} \quad \tau_{p0} \geq T_0 \geq 5\tau_c & \text{(Rule T2)} \\ \frac{\left({}^{\tilde{\tau}_{p0}}/_{\tau_{p0}}\right)}{(\tilde{\tau}_{p0} - T_o)s+1} & \text{if} \quad \tilde{\tau}_{p0} \stackrel{\text{def}}{=} \min\left(\tau_{p0}, 5\tau_c\right) \geq T_0 & \text{(Rule T3)} \end{cases} \tag{2.10}$$

Where τ_c is the SIMC tuning factor. Because the tuning factor is usually decided as a certain factor of the effective time delay (and recommended $\tau_c = \theta$), one does not know this value before the model is approximated. Therefor one will initially have to guess the value for the effective time delay and iterate. When canceling out the positive numerator time constants T_0 against the lag constants τ_{p0}, one usually choose the closest larger lag constant ($\tau_{p0} > T_0$) and use rule T2 or T3. If there is no larger lag constant, then the closest smaller lag constant is chosen and applied to rules T1, T1a or T1b.

2.2. SIMC tuning rules

The SIMC PI controller setting can be found from a FOPDT process (seen in equation 2.1) as follows:

$$k_c = \frac{1}{k_p} \frac{\tau_p}{(\tau_c + \theta)} \tag{2.11}$$

$$\tau_i = \min \left[\tau_p, C \left(\tau_c + \theta \right) \right] \tag{2.12}$$

$$C = 4 \tag{2.13}$$

The closed-loop time constant τ_c is the SIMC method's only tuning parameter, and is selected to give the desired tradeoff between performance and robustness. The recommended setting for fast response with good robustness is:

$$\tau_c = \theta \tag{2.14}$$

The constant $C = 4$ ensures high integral action (i.e. a low value for τ_i) for lag dominated process ($\frac{\tau_p}{\theta} \leq 8$), which gives a better performance for load disturbances. Haugen recommended setting the constant $C = 2$ to further increase performance for lag dominated processes, and reported that this have only a slight negative effect on the gain margin GM.

2.3. Two-step setpoint overshoot method

The setpoint overshoot method recently published by Shamsuzzoha at al. is a closed-loop tuning method, that is designed to give close-to SIMC controller settings. The controller settings is determined by a closed-loop setpoint experiment with a proportional-only (P-only) controller. Controller gain is set sufficiently high to ensure oscillations and a 30% overshoot in the process output.

This method is characterized as a one-step procedure because the controller settings are calculated directly from the setpoint experiment. An alternative two-step procedure is also discussed. First, the closed-loop setpoint experiment is used to determine an open-loop FOPTD process model. Then, the process model is used with a tuning rule like SIMC to determine the PI controller settings.

This report will concentrate on the usage of the two-step setpoint overshoot method as a way of approximating the process model for the SIMC tuning rules. The experimental procedure is as follows:

Step 1: Set the controller to P-only mode

Step 2: Make a set point change. There should be sufficient oscillation in the process output, and the overshoot should be about 30%. If the overshoot is to large or to small, adjust the controller gain k_{c0} until a satisfactory overshoot is obtained.

Step 3: From the closed-loop setpoint experiment, determine the following values:

- Controller gain: k_{c0}
- Time to reach first peak: t_{peak}
- Overshoot: $D = \frac{y_{\max} - y(\infty)}{y(\infty) - y_0}$
- Relative steady state output change, $b = \frac{y(\infty) - y_0}{\Delta r}$

Where $y_{\max}$ is the maximum process output, $y(\infty)$ is the new steady state process output, y_0 is the initial process output and Δr is the change in setpoint.

The setpoint overshoot method gives the following estimate for the process gain:

$$k_p = \frac{1}{k_{c0}} \cdot \left| \frac{b}{1-b} \right| \tag{2.15}$$

The lag time constant can be estimate as follows:

$$\tau_p = 0.86 t_{\text{peak}} \cdot A \cdot \left| \frac{b}{1-b} \right| \tag{2.16}$$

Where the factor A is calculated to be:

$$A = 1.152 \cdot D^2 - 1.607 \cdot D + 1 \tag{2.17}$$

The estimate of the effective time delay is:

$$\theta = 0.305 t_{\text{peak}} \tag{2.18}$$

The setpoint overshoot method uses two different effective delay approximations. The approximation of the effective delay is based on empirical testing, and the value varies from $\theta = 0.305 t_{\text{peak}}$ for lag dominated processes to $\theta = 0.43 t_{\text{peak}}$ for time delay dominated processes. These effective delay approximations are inconsistently used when deriving the lag constant and the effective delay itself. This can easily be seen from the lag constant approximation, when the effective delay approximation is not included:

$$\tau_p = 2\theta \cdot A \cdot \left| \frac{b}{1-b} \right| \tag{2.19}$$

where the time delay is:

$$\theta = 0.43 t_{\text{peak}} \tag{2.20}$$

The approximation for effective time delay is:

$$\theta = 0.305 t_{\text{peak}} \tag{2.21}$$

This inconsistancy leeds to a increase in the controller gain, as demonstrated below:

$$k_c = \frac{1}{k_p} \frac{\tau_p}{(\tau_c + \theta)} = \frac{0.86}{0.61} A \cdot k_{c0} \neq A \cdot k_{c0} \tag{2.22}$$

This inconsistency need to be corrected, and a good approximation for the two regions needs to be found. The one-step overshoot method is used for comparison, and the controller settings can be calculated from the following equations:

$$k_c = A \cdot k_{c0} \tag{2.23}$$

$$\tau_i = \min \left(\; 0.86 A \left| \tfrac{b}{1-b} \right| t_{\text{peak}}, \quad 2.44 t_{\text{peak}} \; \right) \tag{2.24}$$

3. Pareto optimization

Finding a good controller tuning is not as easy as one might think. Though PI controllers only have two tuning parameters, it is hard to find a setting that give both good robustness and good performance. This is because of the tradeoff between robustness and performance. Very robust controllers usually have poor performance, and vice versa. To further complicate the problem, there is also a tradeoff between setpoint performance and disturbance performance. Similar to earlier, a good setpoint response gives usually a poor disturbance response. This tradeoff is small for delay dominated processes, but becomes more pronounced as the process becomes more lag dominated (i.e. large τ_p compared to θ) .

The tuning problem, as mentioned before, contains two conflicting objective functions: minimizing robustness and minimizing performance. One way of illustrating this tradeoff is by constructing a Pareto-optimal (PO) curve, where one of the objective functions are plotted as a function of the other.

A typical example of a PO curve with two conflicting objective functions are shown in figure 3.1. The space above the curve contains feasible solutions, and the space below the curve contains unfeasible solutions. The curve depicts the optimal solutions for each value of the objective functions. When the objective function 1 is low, the objective function 2 is high, and vice versa. These parts of the curve are not interesting, because the goal is to find a good compromise. Usually it is only the part of the curve that is marked with a red line that is of interest. In this region, most solutions gives a good compromise, and the best compromise is based on one's personal judgment.

3.1. Evaluation criteria

As mentioned before, there are two main criteria for a good tuning: performance and robustness. For simplicity, the robustness has been made the independent variable, and performance the dependent variable. In short:

$$\text{perfomance} = f(\text{robustness}) \tag{3.1}$$

Performance and robustness must be quantified to evaluate different tunings.

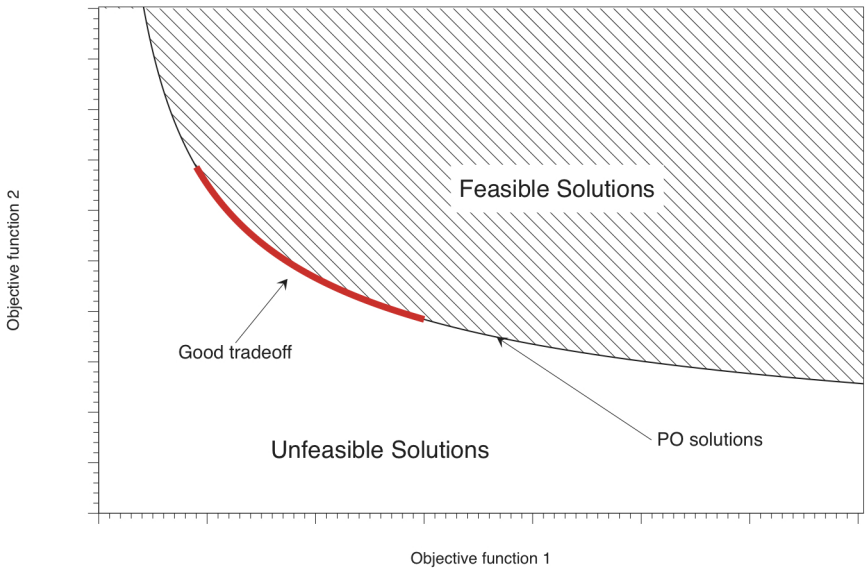


Figure 3.1.: A sketch of a typical Pareto-optimal curve with two conflicting objective functions.

3.1.1. Robustness

The classic measurement of robustness is by the gain margin (GM) and phase margin (PM), where the recommended minimum values for robustness are GM > 1.7 and PM $> 30°$ [4]. The problem with this measurements is that both GM and PM is needed for a complete description of robustness. An alternative robustness measurement is the maximum closed-loop sensitivity (M_s), defined as [5]:

$$M_s = \max |S(j\omega)| \tag{3.2}$$

Where S is the sensitivity function:

$$S(j\omega) = (1 + g_p g_c)^{-1} \tag{3.3}$$

At low frequencies $S \to 0$, and at high frequencies $S \to 1$. At an intermediate frequency range there will be a degradation of the feedback controller performance, resulting in a peaking of the function. M_s is the maximum value of this peak, and represents the worst-case scenario. The M_s is also the inverse of the shortest distance from the loop transfer function $g_p g_c$ to the critical point -1 in the Nyquist plot, and a large distance is desired for robustness (i.e. a small value for M_s). A given M_s always guarantees [5]:

$$GM \geq \frac{M_s}{M_s - 1}; \qquad PM \geq \frac{1}{M_s} \tag{3.4}$$

A Small M_s (i.e. close to one) guarantees a good stability margin. $M_s = 2$ guarantees $GM \geq 2$ and $PM \geq 29.0°$, and represents the recommended upper bounds for robustness.

The report will concentrate mainly on M_s. This because it can simply illustrate the robustness of the system without the use of other measurements.

3.1.2. Performance

To evaluate the performance, the process was subjected to a unit step change in setpoint ($r = 1$) and load disturbance ($d = 1$), and the performance was evaluated for both cases. Performance can be divided into output and input performance. The output performance characterizes the process output signal y, and input performance characterizes the process input signal u.

Output performance

The integrated absolute error (IAE) of the control error ($e = r - y$) is a good indication of the speed and precision of the controller. An IAE closed to one is desired as this is the minimum value. The IAE is defined as follows:

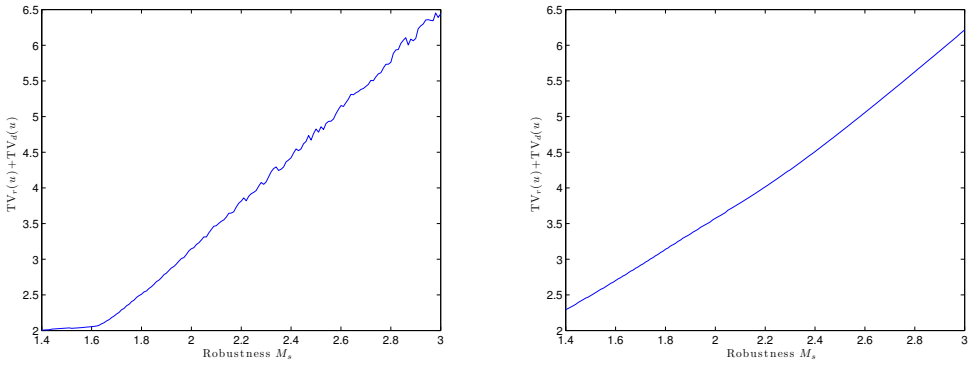
$$\text{IAE} = \int_0^\infty |e| \, dt \tag{3.5}$$

Input performance

The usage of manipulated variable is evaluated by the total variation (TV), which sums up all the changes in the input (u):

$$\text{TV} = \sum_{i=1}^{\infty} |u_{i+1} - u_i| \tag{3.6}$$

A low value for TV is desired and it signals a smooth input signal. The input performance is also an indication of the tuning robustness. If the TV is high, the controller "works actively", indicating an aggressive and thus less robust tuning. Figure 3.2 shows the resulting sum of total variation in setpoint TV_r and disturbance TV_d, from the PO optimal tuning as a function of M_s for a pure time delay process and an integrating process. From the figure, a close correlation between TV and M_s can be seen. Because the objective function should only reflect the tuning performance and not the robustness, the TV is not included.



(a) (b)

Figure 3.2.: Sum in TV from setpoint and disturbance response from the PO optimal tuning for a pure time delay process (figure 3.2a) and an integrating process (figure 3.2b) as a function of M_s.

3.2. The Objective function

There where two main alternatives for constructing the objective function: absolute values and relatives values for IAE. In most cases, disturbance performance is worse than setpoint performance. For the absolute values, a relatively small increase in disturbance performance could then completely overshadow a relatively large decrease in the setpoint performance. This could be adjusted by weighting the performance for the two cases. One problem is that the response can vary greatly from process to process, and consequently the weighting needs to change to achieve good performance tradeoff.

A solution to the weighting problem would be to use relative values for performance, where it is measured against a reference. An increase in disturbance performance can then only be justified by an equal or less decrease in setpoint performance. The problem lies in the selection of a performance reference. Initially, the performance of SIMC was considered as the reference, but the SIMC tuning changes focus from setpoint to disturbance performance for lag dominated processes, which gives a noncontinuous reference. To achieve a continuous reference, the extreme tuning scenarios with best setpoint and best disturbance performance is used. Because the best performance is dependent on the robustness, a reference value need to be selected. When constructing the PO curve, a $M_s = 1.59$ was used.

In short, the reference values for setpoint performance (IAE_r°) and disturbance (IAE_r°) performance are as follows:

$$\mathrm{IAE}_r^\circ = \min\left[\mathrm{IAE}_r(g_{c,i}, M_s = 1.59)\right] \tag{3.7}$$

$$\text{IAE}_d^\circ = \min\left[\text{IAE}_d(g_{c,i}, M_s = 1.59)\right] \tag{3.8}$$

Where IAE_r and IAE_d is the performance for setpoint and disturbance respectively, and $g_{c,i}$ is all the tunings that satisfy the M_s requirement. The objective function then becomes as follows:

$$J = 0.5\left[\frac{\text{IAE}_r}{\text{IAE}_r^\circ} + \frac{\text{IAE}_d}{\text{IAE}_d^\circ}\right] \tag{3.9}$$

The minimum value of the objective function was set to one by multiplying the objective function with 0.5. The minimum objective function will be referred to as the (PO) tuning performance. If the tuning performance is equal to one, then there exist a tuning that can achieve both the best setpoint performance and the best disturbance performance, as is possible for delay dominated processes. This will not be the case for more lag dominated processes, and the tuning performance will be greater than one. The distance from one to the PO tuning performance can be thought off as loss in performance resulting from the tuning compromise, and will be referred to as compromise loss. See figure 3.3. In addition, the difference between the PO tuning performance and the performance for another tuning rule, can be thought of as loss in performance due to non-optimality and will be referred to as non-optimality loss. These losses can only be calculated at the reference M_s for the PO curve. Later in the report, the reference M_s will be the same as the M_s for the controller tuning, making it possible to calculate the controller tunings non-optimality loss.

3.3. Simulations

MatLab and Simulink is used for all calculations and simulation work in this report. The MatLab scripts are not included, but algorithms are given for large and complicated calculations. The Simulink block diagram and two of the most important functions are included in appendix A.

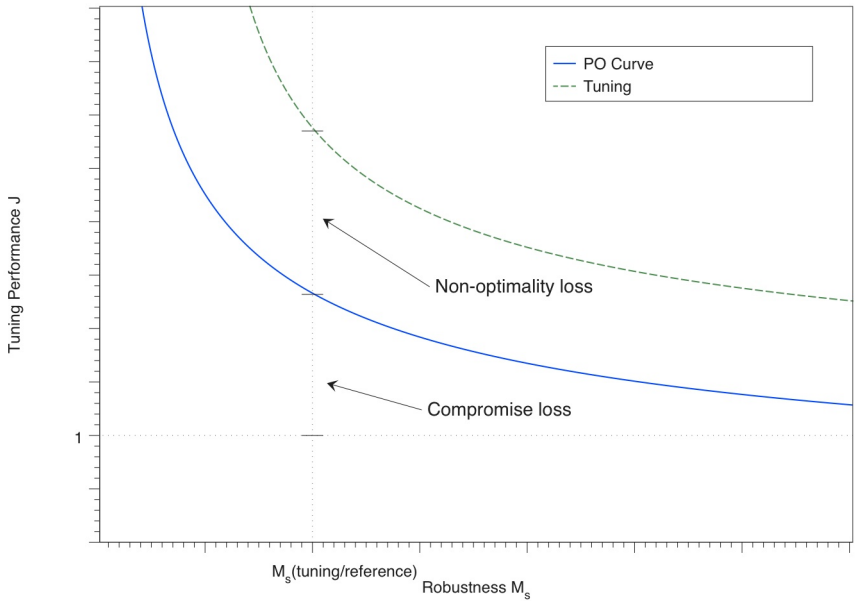


Figure 3.3.: Illustration of the compromise loss and non-optimality loss definition.

4. Comparison between SIMC and the Pareto-optimal

As mentioned in the introduction, there have been inquiries toward the improvability of SIMC tuning. To improve disturbance performance for lag dominated processes, Haugen recommended an increase in integral action (i.e. decreased integral time) by decreasing $C = 4$ to $C = 2$. To investigate the improvability, a comparison between SIMC performance and the PO performance for different processes was done. The chosen processes was:

1. Pure time delay process: $g_p(s) = \exp(-s)$
2. Time delay dominated process: $g_p(s) = \frac{1}{(s+1)} \exp(-s)$
3. Lag dominated process: $g_p(s) = \frac{1}{(8s+1)} \exp(-s)$
4. Integrating process: $g_p(s) = \frac{1}{s} \exp(-s)$

For each process, the PO tuning performance was calculated as a function of M_s. A short calculation algorithm can be seen in algorithm 4.1. The SIMC tuning performance was found as a function of M_s by varying the closed-loop tuning factor τ_c. Also, the original ($C = 4$) and Haugen's improved ($C = 2$) SIMC was compared.

The results for the pure time delay process is shown in figure 4.1a. Because the SIMC tuning for pure time delay is $k_p = 0$ and $\tau_i = 0$, the SIMC results show the tuning results for the following close-to pure time delay process:

$$g_p(s) = \frac{1}{0.01s + 1} \exp(-s) \approx \exp(-s) \tag{4.1}$$

Though the PO and SIMC tuning are not for the same process, the processes are approximately the same and give a good indication for the optimality of SIMC. The SIMC tuning had a substantial non-optimality loss in performance, but the recommended tuning factor $\tau_c = \theta$ give a robust setting with a good tradeoff.

Algorithm 4.1 Calculation summary of the PO-curve

With a given process model g_p:

Step 1: Determine the best response scenarios

For $M_s = 1.59$ do the following:

1. Find the k_c and τ_i relation that satisfy the M_s requirement. (i.e. give a τ_i range and find the corresponding k_c values). See appendix A.2 for MatLab code.
2. From the k_c and τ_i relation determine the controller setting that gives the smallest IAE for a change insetpoint ($\Delta r = 1$) and load disturbance ($\Delta d = 1$), and use these as the references IAE_r° and IAE_d°.

Step 2: Determinine the PO curve

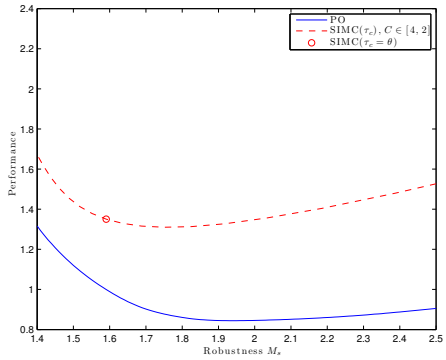
For a given range of M_s values do the following:

1. Find the k_c and τ_i relation that satisfy the M_s requirement .
2. For each controller setting calculate the objective function J.
3. Find the controller setting that minimizes the objective function.
4. Plot the minimum objective function for each M_s value.

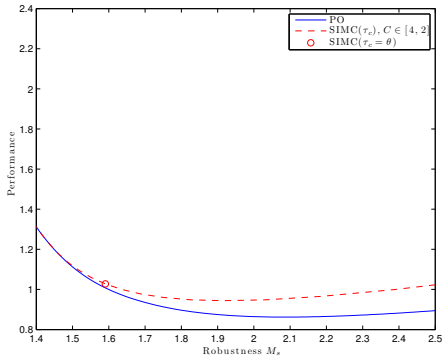
Step 3: Determining the SIMC curve

Give a tuning factor τ_c range and do the following:

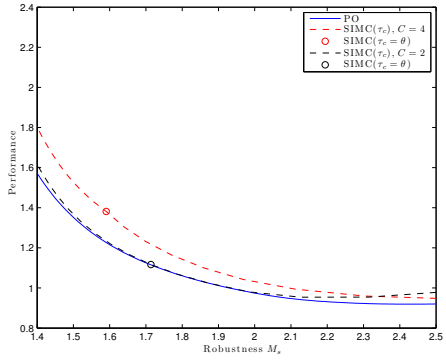
1. Calculate the M_s for each value of τ_c. See appendix A.1 for MatLab code.
2. Calculate the objective function J.



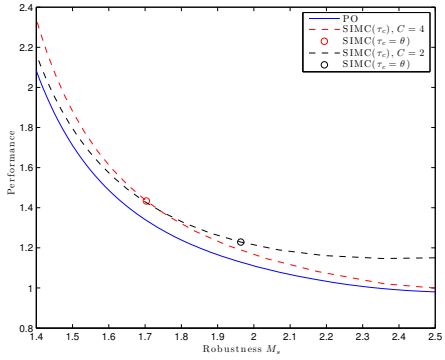
(a) Tuning performance for $g_p = \exp(-s)$.



(b) Tuning performance for $g_p = \frac{1}{(s+1)} \exp(-s)$.



(c) Tuning performance for $g_p = \frac{1}{(8s+1)} \exp(-s)$.



(d) Tuning performance for $g_p = \frac{1}{s} \exp(-s)$.

Figure 4.1.: PO tuning performance for various process as a function of M_s, plotted with SIMC tuning performance, for both $C = 4$ and $C = 2$, as function of the tuning parameter τ_c and the recommended SIMC tuning factor $\tau_c = \theta$.

For the time delay dominated process, SIMC tuning followed the PO tuning closely, resulting in little non-optimality loss, as shown in figure 4.1b. The tuning factor $\tau_c = \theta$ gave good robustness ($M_s \approx 1.59$) and tradeoff.

For the lag dominated process, Haugen's ($C = 2$) SIMC gave a close fit with the PO curve with almost no non-optimality loss, as seen by figure 4.1c. This resulted however in a lower robustness ($M_s \approx 1.7$). The original ($C = 4$) SIMC setting had some non-optimality loss, but had good robustness ($M_s \approx 1.59$).

For the integrating process, both original ($C = 4$) SIMC and Haugen's ($C = 2$) SIMC had some non-optimality loss in performance, as seen by figure 4.1d. The tuning factor $\tau_c = \theta$ gave reduced robustness for both cases. Robustness for Haugen's SIMC was rather poor, bordering on the upper limits of robustness. The original SIMC, though reduced, gave good robustness ($M_s \approx 1.7$). This because the steep incline in the PO curve at $Ms < 1.7$ would give a large deterioration of the performance for a small increas in the robustness. Thus the tuning factor $\tau_c = \theta$ gives a good tradeoff for the original SIMC.

5. Improved-SIMC rule

The previous section showed that there is room for improvement in the SIMC tuning rules, both for close-to pure time delay and lag dominated processes. To find out where the improvements could be made, a comparison between the SIMC tuning and the PO tuning was done. The optimal tuning was found for multiple processes ranging from close-to pure time delay to close-to integrating. The process model can be shown as follows:

$$g_p(s) = \frac{1}{\tau_p s + 1} \exp(-s) \tag{5.1}$$

Where the lag constant was:

$$\tau_p \in \{0.01, 0.02, \ldots, 0.99, 1.0, 1.1, \ldots, 15.9, 16, 17, \ldots 49, 50\} \tag{5.2}$$

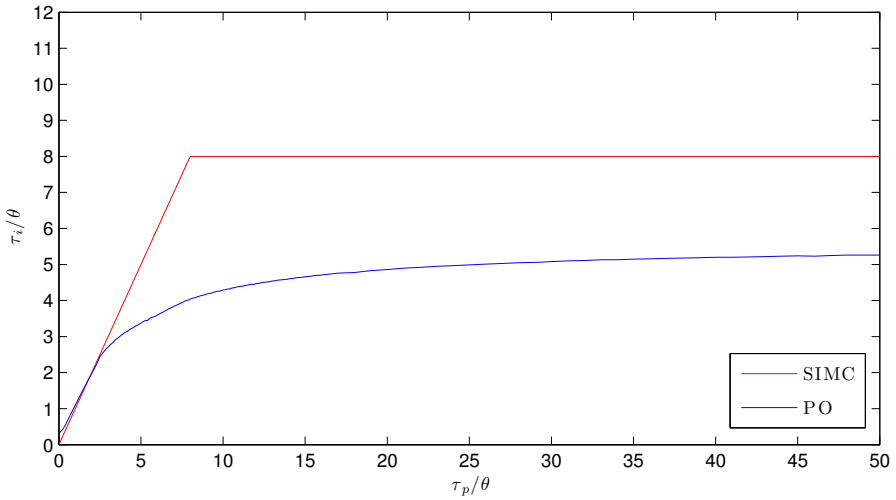
A robustness $M_s = M_s\,(\text{SIMC})$, that is M_s equal to the M_s from the SIMC tuning, was chosen for the PO controller settings. This was done for two reasons:

1. Keeping a constant $M_s = 1.59$ would give poor performance for lag dominated processes.

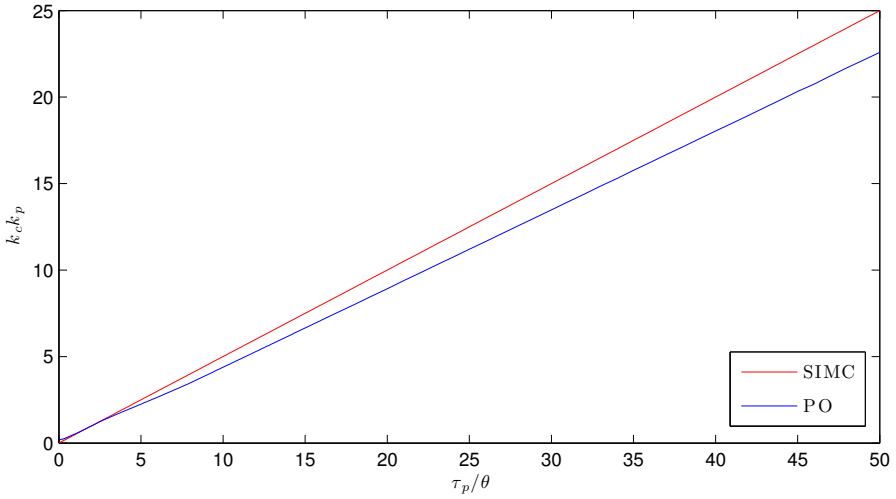
2. $M_s\,(\text{SIMC})$ had a good compromise between performance and robustness for all four cases in the previous section.

To get a representative weighting for the actual process, it was decided that the objective function would also use $M_s\,(\text{SIMC})$ for the reference performance. A summary of the calculation algorithm can be seen in algorithm 5.1, and the results can be seen in figures 5.1a and 5.1b.

From the two figures, three separate regions can be seen: time delay dominated ($\frac{\tau_p}{\theta} < 1$), intermediate region ($1 \leq \frac{\tau_p}{\theta} < 6$) and lag dominated ($\frac{\tau_p}{\theta} \geq 6$). The PO tuning for close-to pure time delay processes does not approach zero when $\frac{\tau_p}{\theta}$ approaches zero, as with SIMC, and the slope of the PO tuning is gentler. For the intermediate region, the PO optimal tuning fits nicely with the SIMC tuning, but due to the gentle transition, there is some deviation when approaching the lag dominated region. The lag dominated region



(a) PO and SIMC τ_i-tuning as a function of τ_p/θ for the process $g_p = \frac{1}{(\tau_p s+1)} \exp(-\theta s)$.



(b) PO and SIMC k_c-tuning as a function of τ_p/θ for the process $g_p = \frac{1}{(\tau_p s+1)} \exp(-\theta s)$.

Figure 5.1.

Algorithm 5.1 Determinim the PO tuning for a process model.

For a given process model g_p do the following:

1. Determine the SIMC controller setting.
2. Determine the M_s value resulting from the SIMC tuning, i.e. $M_s M_s$ (SIMC).
3. Find the k_c and τ_i relation that satisfy the $M_s = M_s$ (SIMC) requirement.
4. From the k_c and τ_i relation determine the controller setting that gives the smallest IAE for a change insetpoint ($\Delta r = 1$) and load disturbance ($\Delta d = 1$), and use these as the references IAE_r° and IAE_d°.
5. From the k_c and τ_i relation calculate the objective funtion J.
6. Find the controller setting that minimizes the objective function J.
7. Plot the controller setting as a function of the process model.

has lower terminal integral time than SIMC and approaches $\tau_i \approx 5.5$. The slope of PO controller gain is reduced to about 90% of the SIMC slope. Considering this results, the following improved-SIMC (I-SIMC) rule has been constructed.

Time delay dominated region: When the $\frac{\tau_p}{\theta} \to 0$, k_c and τ_i approaches $k_c k_p = 0.2$ and $\frac{\tau_i}{\theta} = 0.3$ respectively, as seen in figures 5.1a and 5.1b. Because SIMC controller setting is almost PO when $\frac{\tau_p}{\theta} \to 1$, the controller setting should approach this setting. Therefore the I-SIMC rule for time delay dominated processes becomes:

$$k_c = \frac{\tau_p}{(\tau_c + \theta)} \frac{1}{k_c} + \frac{0.3}{k_p} \left(1 - \frac{\tau_p}{\theta}\right) \tag{5.3}$$

$$\tau_i = 0.3\theta + 0.7\tau_p \tag{5.4}$$

To increase the influence of the tuning factor on the controller settings, the tuning factor τ_c has been introduced into the equation 5.4 . This is done by replacing the time delay with $0.5\,(\tau_c + \theta)$:

$$\tau_i = 0.15\,(\tau_c + \theta) + 0.7\tau_p \tag{5.5}$$

Intermediate and lag dominated region: The SIMC integral time for intermediate region is close to the PO integral time, and no modification of the SIMC rule is necessary. For lag dominant processes the SIMC integral time is too high and needs to be reduced. A good value for the integral time is $\frac{\tau_p}{\theta} = 6$. This is slightly higher than the terminal integral

time seen from figure 5.1a, and was chosen to restrict the increase in M_s that follows from decreasing the integral time from the SIMC rule. The I-SIMC rule for integral time becomes:

$$\tau_i = \min\left[\tau_p, 6\theta\right] \tag{5.6}$$

Implementing the tuning factor τ_c by replacing the time delay for the lag dominated processes with $0.5\left(\tau_c + \theta\right)$:

$$\tau_i = \min\left[\tau_p, 3\left(\tau_c + \theta\right)\right] \tag{5.7}$$

Reducing the integral time without reducing the controller gain leads to an increased M_s value and reduced robustness, as seen by Haugen's reduction of C. Therefore, a compensation is needed in the controller gain rule for the I-SIMC's reduction of integral time. For lag dominated processes, the controller gain should be about 90% of the SIMC controller gain, as seen from figure 5.1b. The controller gain should be equal at the point of switching from the intermediate region to the lag dominated region. After the switching, the slope of the controller gain should be reduced. The intersect a of the I-SIMC controller gain at $\tau_p = 0$ with a slope of n, with both SIMC and I-SIMC equal at point of switching $\tau_p = 3\left(\tau_c + \theta\right)$ is:

$$k_c = a + \frac{n}{k_p} \cdot \frac{3\left(\tau_c + \theta\right)}{\tau_c + \theta} = \frac{1}{k_p} \cdot \frac{3\left(\tau_c + \theta\right)}{\tau_c + \theta} \tag{5.8}$$

$$a = \frac{3\left(1 - n\right)}{k_p} \tag{5.9}$$

The controller gain slope $n = 0.94$ was chosen to give a close-to SIMC robustness, and can be seen from figure 5.3. The controller gain for lag dominated processes becomes:

$$k_c = \frac{3(1 - 0.94)}{k_p} + \frac{0.94}{k_p} \frac{\tau_p}{(\tau_c + \theta)} \tag{5.10}$$

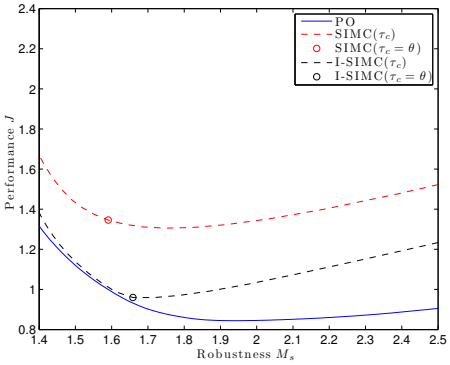
The controller gain for both delay and lag dominated processes will then be:

$$k_c = \min\left[\begin{array}{ccc} \frac{1}{k_p} \frac{\tau_p}{(\tau_c + \theta)} & , & \frac{0.18}{kp} + \frac{0.94}{k_p} \frac{\tau_p}{(\tau_c + \theta)} \end{array} \right] \tag{5.11}$$

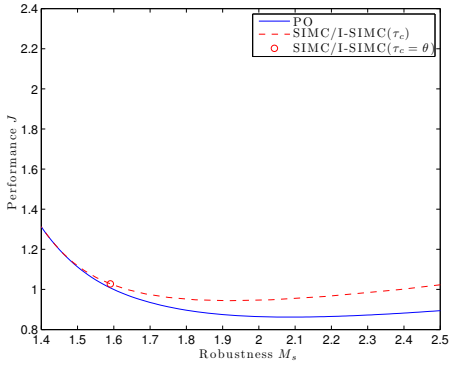
The controller gain for integrating processes, where $\tau_p \to \infty$ and the process can be approximated as $g_p = \frac{k_p}{\tau_p s+1} \approx \frac{k_p}{\tau_p s} = \frac{k'_p}{s}$, gives a controller gain:

$$k_c = \frac{0.94}{k'_p} \frac{1}{(\tau_p + \theta)} \tag{5.12}$$

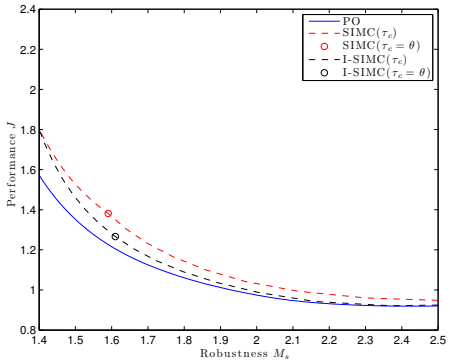
The same comparison conducted in section 4 was done for the I-SIMC rule, and the results are shown in figure 5.2. The I-SIMC rule was closer to the PO tuning for all cases, except for the delay dominated process where it was equal to the SIMC. The pure time delay process had the biggest improvement. The recommended tuning factor gave, in most cases, a M_s value slightly higher for I-SIMC than SIMC, as seen by figure 5.3. There is also a sharp increase in M_s for the delay dominated region, but the increase is well within the limits of robustness.



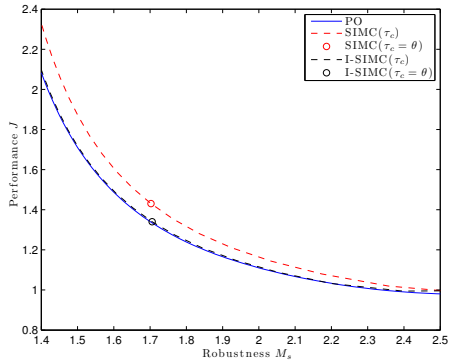
(a) Tuning performance for $g_p = \exp(-s)$.



(b) Tuning performance for $g_p = \frac{1}{(s+1)} \exp(-s)$.



(c) Tuning performance for $g_p = \frac{1}{(8s+1)} \exp(-s)$.



(d) Tuning performance for $g_p = \frac{1}{s} \exp(-s)$.

Figure 5.2.: PO tuning performance for various process as a function of M_s, plotted with SIMC and I-SIMC tuning performance as function of the tuning parameter τ_c and the recommended SIMC tuning factor $\tau_c = \theta$.

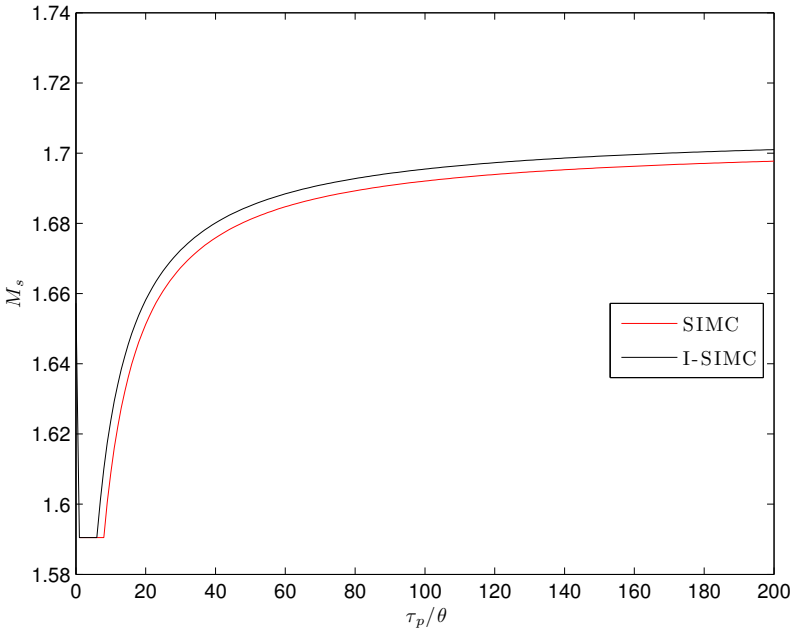


Figure 5.3.: Comparison between the resulting M_s value for SIMC and I-SIMC tuning for the process $g_p = \frac{1}{\tau_p s+1} \exp(-s)$.

6. Evaluation of the improved SIMC rule

The I-SIMC rule was tested on 15 different processes and compared with SIMC, and the results are shown in table 6.1. The integral time that gives the best setpoint response and the integral time that gives the best disturbance response are shown for their respected M_s value. For a good tradeoff, the controller integral time should be between these two best response scenarios. If the controller integral time is not in-between, there is no longer a compromise and there would only be a deterioration of the performance. The controller gain is not included in the results because it is determined by the given integral time and the M_s.

Typical controller gain and integral time relation for delay dominated and lag dominated process are shown in figure 6.1a and 6.1b. The integral time for the best setpoint $\tau$and disturbance response also included. For delay dominated processes, the best response scenarios merges together, resulting in one solution, which is also the optimum. This narrow best response span results in a sharp optimum, as seen in figure 6.1c, and an inaccurate controller setting would greatly affect the performance.

With increasingly lag dominated processes, the best response scenarios are spread further apart. For the integrating case ($\tau_p \to \infty$) the best setpoint scenario gives an integral time $\tau_i = \infty$. This large span gives a flat optimum, as seen from figure 6.1d. As a good disturbance response is, in most cases, more preferred than a good setpoint response, a setting closer to the best disturbance scenario is desired, which is achieved by selecting a shorter integral time. Selecting a too short integral time results in a rapid deterioration of the performance, as seen from the performance curve. Thus some back-off from this point could be appropriate. Selecting a too large integral time has little effect on overall tuning performance (given that the controller gain is selected accordingly), but will deteriorate the disturbance response.

The I-SIMC rule gave for the majority of the processes, excluding the processes from the intermediate region, an increase in the M_s value compared to the SIMC. This is to be excepted since the I-SIMC rule was constructed to give a slightly higher M_s value than SIMC (figure 5.3). If so desired, the M_s value for lag dominated processes can be reduced by reducing the slope of the controller gain.

Many processes from the delay dominated and intermediate regions had a controller setting outside the best response scenarios. This is to be expected, since the best response span for these processes are very narrow. The tunings had generally a too low integral

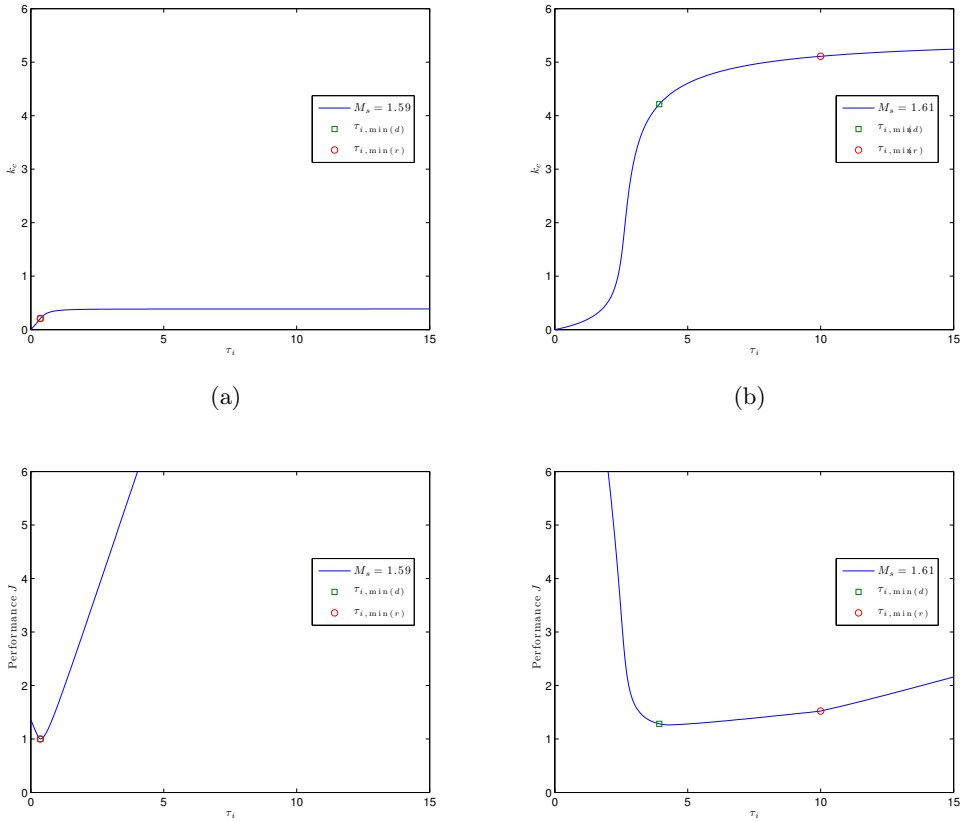


Figure 6.1.: Controller gain k_c and integral time τ_i relation for a given M_s, plotted with the controller setting that gives the best setpoint and disturbance response for the processes $g_p = \frac{1}{(0.1s+1)} \exp(-s)$ (figure 6.1a) and $g_p = \frac{1}{(10s+1)} \exp(-s)$ (figure 6.1b) . The performance objective function is also plotted for the same processes as a function of the integral time (figure 6.1c and 6.1d).

time, but the non-optimality loss was low, ranging from 1 to 4%. Some exceptions are processes E4 and E13, which had a large non-optimality loss of 19 to 51% respectively for SIMC. For the I-SIMC rule the non-optimality loss for process E4 was lower, but still large at 8%.

Because processes from the same region had generally low non-optimality loss, it is suspected that this non-optimality loss is amplified by the half-rules approximation of the lag constants, and not by the SIMC or I-SIMC rule. This will be investigated further in the next section. The large non-optimality loss for process E13, might be caused by the half-rules approximation of the positive numerator time constants. From the best response scenarios, is seems that the current approximation over estimates the size of the lag constants.

Most of the lag dominated processes had an integral time within the best response scenarios. The non-optimality loss for I-SIMC was lower for the majority of the processes, indicating that the I-SIMC rule gives a more "optimal" tradeoff. For the integrating processes, SIMC had the lowest non-optimality loss for processes E6 and E8, and I-SIMC had the lowest non-optimality loss for process E14. For all three processes the I-SIMC rule gave a lower integral time, resulting in a better disturbance response than SIMC.

Table 6.1.: Comparison between SIMC and I-SIMC ($\tau_c = \theta$) performance and robustness for 15 different processes, with non-optimality loss (J/J_{opt}) and integral time for best setpoint ($\tau_{i,\min(r)}$) and disturbance ($\tau_{i,\min(d)}$) response for the respective M_s value.

				Controller settings			Performance					Best response	
Case	Process model g_{p0}	τ_p/θ	Tuning rule	k_c	τ_i	M_s	IAE$_r$ (y)	TV$_r$ (u)	IAE$_d$ (y)	TV$_d$ (u)	J/J_{opt}	$\tau_{i,\min(d)}$	$\tau_{i,\min(r)}$
E1	$\frac{1}{(s+1)(0.2s+1)}$	11	SIMC(l)	5.5	0.8	1.56	0.355	12.7	0.145	1.55	1.01	0.71	1.32
			I-SIMC(l)	5.35	0.6	1.67	0.4	13.1	0.112	1.74	1.1	0.66	1.26
E2	$\frac{1}{(2s+1)(1s+1)(0.4s+1)(0.12s+1)(0.05s+1)^3}$	1.7	SIMC(d)	0.85	2.5	1.66	3.57	1.9	2.97	1.26	1.01	2.54	3.1
			I-SIMC(im)	0.85	2.5	1.66	3.57	1.9	2.97	1.26	1.01	2.54	3.1
E3	$\frac{1}{(s+1)(0.1s+1)^2}$	7	SIMC(d)	2.33	1.05	1.55	0.464	4.98	0.452	1.3	1.04	0.61	1.17
			I-SIMC(l)	2.31	0.9	1.58	0.478	5.03	0.391	1.36	1.01	0.61	1.18
E4	$\frac{1}{(s+1)^4}$	0.6	SIMC(d)	0.3	1.5	1.46	5.59	1.15	5.4	1.1	1.19	2.22	2.31
			I-SIMC(d)	0.38	1.8	1.46	5.1	1.14	4.93	1.08	1.08	2.22	2.32
E5	$\frac{1}{(s+1)(0.2s+1)(0.05s+1)(0.008s+1)}$	7.43	SIMC(d)	3.72	1.1	1.59	0.451	8.15	0.296	1.41	1.02	0.71	1.21
			I-SIMC(l)	3.67	0.888	1.65	0.48	8.37	0.242	1.52	1	0.71	1.25
E6	$\frac{(0.17s+1)^2}{s(s+1)^2(0.028s+1)}$	(i)	SIMC	0.296	13.5	1.47	6.5	0.67	45.7	1.55	1	10.7	∞
			I-SIMC	0.278	10.1	1.52	6.59	0.663	36.5	1.68	1.03	10	∞
E7	$\frac{-2s+1}{s(s+1)^3}$	0.429	SIMC(d)	0.214	1.5	1.66	7.28	1.05	8.33	1.29	1.04	1.73	1.75
			I-SIMC(d)	0.329	2.1	1.79	6.39	1.04	7.53	1.34	1.01	1.94	1.99
E8	$\frac{1}{s(s+1)^2}$	(i)	SIMC	0.333	12	1.77	6.43	0.854	36	1.78	1	9.4	∞
			I-SIMC	0.313	9	1.86	6.7	0.862	28.7	1.97	1.03	8.9	∞
E9	$\frac{1}{(s+1)^2}\exp(-s)$	1	SIMC(d)	0.5	1.5	1.61	3.38	1.32	3.14	1.15	1.04	1.68	1.87
			I-SIMC(im)	0.5	1.5	1.61	3.38	1.32	3.14	1.15	1.04	1.68	1.87

Table 6.1 Continued

Case	Process model g_{p0}	τ_p/θ	Tuning rule	Controller settings		M_s	Performance				J/J_{opt}	Best response	
				k_c	τ_i		$\text{IAE}_r\,(y)$	$\text{TV}_r\,(u)$	$\text{IAE}_d\,(y)$	$\text{TV}_d\,(u)$		$\tau_{i,\min(d)}$	$\tau_{i,\min(r)}$
E10	$\frac{1}{(20s+1)(2s+1)}\exp(-s)$	10.5	SIMC(l)	5.25	16	1.72	6.34	12.3	3.05	1.49	1.03	9.7	21.3
			I-SIMC(l)	5.12	12	1.79	6.98	12.6	2.35	1.64	1	9.5	21.5
E11	$\frac{-s+1}{(6s+1)(2s+1)^2}\exp(-s)$	1.4	SIMC(d)	0.7	7	1.63	11.5	1.59	10.1	1.2	1.01	7.2	8.5
			I-SIMC(i)	0.7	7	1.63	11.5	1.59	10.1	1.2	1.01	7.2	8.5
E12	$\frac{(6s+1)(3s+1)}{(10s+1)(8s+1)(s+1)}\exp(-0.3s)$	3.33	SIMC(d)	7.41	1	1.66	1.07	18.3	0.148	1.39	1.01	0.85	2.04
			I-SIMC(im)	7.41	1	1.66	1.07	18.3	0.148	1.39	1.01	0.85	2.04
E13	$\frac{2s+1}{(10s+1)(0.5s+1)}\exp(-s)$	3.6	SIMC(d)	2.88	4.5	1.74	2.86	6.56	1.61	1.2	1.51	1.19	2.17
			I-SIMC(im)	2.88	4.5	1.74	2.86	6.56	1.61	1.2	1.51	1.19	2.17
E14	$\frac{-s+1}{s}$	(i)	SIMC	0.5	8	2	3.59	2.04	17.3	3.4	1.27	3.5	∞
			I-SIMC	0.47	6	1.89	3.71	1.85	14	3.31	1.1	3.8	∞
E15	$\frac{-s+1}{(s+1)}$	1	SIMC	0.5	1	2	2	1	2.85	3	1.09	0.8	0.85
			I-SIMC	0.5	1	2	2	1	2.85	3	1.09	0.8	0.85

(i) Integrating process $g_p = \frac{1}{s}$,
(l) Lag dominated process: SIMC: $\tau_p/\theta \geq 8$, I-SIMC: $\tau_p/\theta \geq 6$
(d) Delay dominated process: SIMC: $\tau_p/\theta < 8$, I-SIMC: $\tau_p/\theta < 1$

(im) intermediate region: I-SIMC: $1 \leq \tau_p/\theta < 6$

7. Improved lag approximation

As previously shown, the SIMC tuning had a higher than average non-optimality loss for higher order processes with equal lag time constants. It is thought that this was caused by a weakness in the half-rules approximation of the lag constants, and not by the SIMC tuning rule. An investigation was conducted to substantiate this claim and the objective of the investigation was:

1. To determine if the half-rule was the cause of the high non-optimality loss.

2. To find an easy approximation for higher order processes with equal lag constants, that give better results that the half-rule.

To confirm that the half-rule is the cause of the non-optimality loss, a comparison between a higher order process with equal lag constants and processes from the same delay dominated region was conducted, and the results are shown in table 7.1.

For Processes with several equal lag time constants, there is generally an non-optimality loss larger than the average. For process E4, the half-rule non-optimality loss is 18%. The half-rule approximation gives $\tau_p/\theta = 0.6$. Processes from the same τ_p/θ region without equal lag constants are shown in process E4-B and E4-C, and the half-rule results in an non-optimality loss of 6%. Because the processes are in the same delay dominated region, the SIMC rule should give approximately the same results. Thus, if the non-optimality loss was cause by the SIMC rule, it would expected that the loss was approximately equal for the three processes. Because process E4, has a much higher loss than the other processes, it is reasonable to conclude that the increase in non-optimality loss for process E4 is cause by the half-rule approximation.

Table 7.1.: Comparison between the SIMC non-optimality loss for process models with equal τ_p/θ.

		Approximated model, g_p			
Case	Process model, g_{p0}	τ_p	θ	τ_p/θ	J/J_{opt}
E4	$\frac{1}{(s+1)^4}$	1.5	2.5	0.6	1.18
E4-B	$\frac{1}{(0.6s+1)}\exp(-s)$	0.6	1	0.6	1.06
E4-C	$\frac{1}{(0.5s+1)(0.2s+1)}\exp(-0.9s)$	0.6	1	0.6	1.06

As a first step, an easy addendum to the half-rule approximation was desired to compensate for the half-rules shortcoming in approximating equal lag constants. An approximation was to be found for the process:

$$g_{p0} = \frac{1}{(s+1)^n} \tag{7.1}$$

Where

$$n \in \{2, 3, 4, 5\} \tag{7.2}$$

The objective was to find a simple approximation that reduced the IAE $= \int |e|\, dt$ between the original process and the approximated process. The error e was then defined as:

$$e = y\left(g_{p0}\right) - y(g_p) \tag{7.3}$$

Where $y\left(g_{p0}\right)$ is the process output for the original process model and $y\left(g_p\right)$ is the process output for the approximated process model. Two approximation scenarios where tested:

1. The sum of the lag time constants and the time delay was to be equal for both the original and approximated model (*constant sum optimalization*).

2. The approximated lag time constant and the time delay could be chosen freely.

The results are shown in table 7.2. For the second order process, the differences between the half-rule approximated model and the two optimized model approximations where small, and the half-rule seems valid. However, this difference increased for higher order processes, with 80 to 100% decrease in error for the 5th order process. Thus an addendum for this kind of processes could drastically improve the accuracy of the model. The difference between the two optimized scenarios was small. Though this difference also increased with the process order, the increase was not significant compared to the half-rule. Because of this, and to achieve simple relations, the constant sum approximation is chosen for further work.

Table 7.2.: Comparison between the half rule and the minimum error optimized approximations.

		$n=2$	$n=3$	$n=4$	$n=5$
Half-rule approximation	τ_p	1.5	1.5	1.5	1.5
	θ	0.5	1.5	2.5	3
	IAE	0.135	0.300	0.511	0.702
Constant τ_p and θ sum	τ_p	1.44	1.77	2.04	2.27
	θ	0.56	1.23	1.96	2.73
	IAE	0.130	0.232	0.317	0.393
Freely choosen τ_p and θ	τ_p	1.45	1.78	2.06	2.29
	θ	0.61	1.34	2.10	2.91
	IAE	0.120	0.213	0.292	0.360

The first order regression gives an increase in the approximated lag equal to 0.276 for higher than second order models. To simplify, the increase can be approximated as 0.3. Because of the constant sum optimization, the increase in time delay becomes 0.7. The approximated lag time constant and time delay can be expressed as:

$$\tau_p \approx \tau_{p0,1} + 0.5\tau_{p0,2} + \sum_{i\geq 3} 0.3\tau_{p0,i} \tag{7.4}$$

$$\theta \approx \theta_0 + 0.5\tau_{p0,2} + \sum_{i\geq 3} 0.7\tau_{p0,i} \tag{7.5}$$

This approximation works best for processes with close-to equal lag time constants, and the validity of this approximation is determined by two factors: the order of the process and the relative difference in size between the lag constants. To illustrate this, a constant sum optimization was coducted for processes of different orders, with varying sizes of the dominating lag constant. The result is shown in figure 7.1, and two trends may be observed for minimizing IAE.

1. With increasing order, more of the smaller lags should be added to the approximated lag constant.

2. With increasing dominant lag constant, more of the smaller lags should be approximated as time delay.

As the largest lag constant becomes more dominating, the approximated lag constant will approach the same size. Thus, the half-rule addendum is only valid if all the lag constants are almost equal. Table 7.3 shows the value of the largest lag constant when

the half-rule addendum gives the same IAE as the original half-rule. For third, fourth and fifth order processes this occurs when the largest lag constant deviates more than 40, 60 and 80% respectively compared to the smaller lag constants. If the dominating lag constant is larger than this, the half-rule will give a better approximation.

Table 7.3.: The absolute error for the half-rule, the constant sum and the half-rule with addendum approximations for the process $g_{p0} = \frac{1}{\prod_{i=1}^{n}(\tau_{p0,i}+1)}$, where $\tau_{p0,(i\geq 2)} = 1$.

		$n = 3$ $\tau_{p0,1} = 1.4$	$n = 4$ $\tau_{p0,1} = 1.6$	$n = 5$ $\tau_{p0,1} = 1.8$
IAE	Half-rule approximation	0.267	0.404	0.525
	Constant τ_p and θ sum	0.253	0.348	0.431
	Half-rule with addendum	0.276	0.400	0.529

Also seen in table 7.3, is that the constant sum approximation is still better than the half-rule. Therefore, a more accurate approximation can be made, which also takes into account the two trends seen in figure 7.1. To investigate how the optimal approximation behaves with the relative size of the lag constants, the constant sum approximation was conducted for the process:

$$g_{p0} = \frac{1}{(\tau_{p0,1}s + 1)} \frac{1}{(s + 1)} \tag{7.6}$$

Where the largest lag constant was:

$$\tau_p \in \{1, 2, \ldots, 19, 20\} \tag{7.7}$$

The larger lag constant $\tau_{p0,1}$ is the basis of the approximated lag constant τ_p. When reducing the order of the process model, the smaller lag constants are divided between the larger time constant and the effective time delay in a certain ratio. For the smaller lag constant in this process ($\tau_{p0,2} = 1$), the fraction approximated as lag by the constant sum approximation is plotted as a function of the ratio $\frac{\tau_{p0,2}}{\tau_{p0,1}}$, and shown in figure 7.2. The following linear relationship gave the best correlation with the optimum ratio:

$$\frac{\tau_p - \tau_{p0,1}}{\tau_{p0,2}} = 0.4722 \cdot \frac{\tau_{p,i+1}}{\tau_{p0,i}} - 0.01544 \tag{7.8}$$

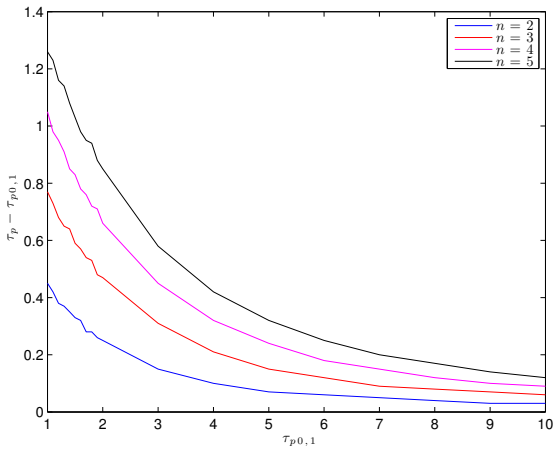


Figure 7.1.: The difference between the constrant sum approximated lag time constant τ_p and the largest lag time constant $\tau_{p0,1}$ for the process $g_{p0} = \frac{1}{(\tau_{p0,1}s+1)(s+1)^{n-1}}$.

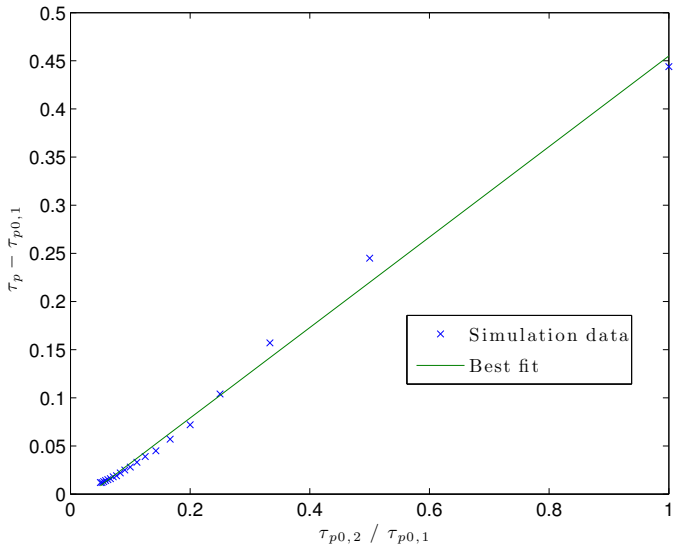


Figure 7.2.: Constant sum optimization result for the process $g_{p0} = \frac{1}{(\tau_{po,1}s+1)(s+1)}$, where the ratio of smaller $\tau_{p0,2}$ added to the approximated FOPTD τ_p is plotted as a function of the size ratio between $\tau_{p0,2}$ and $\tau_{p0,1}$. The best fit linear correlation is also shown.

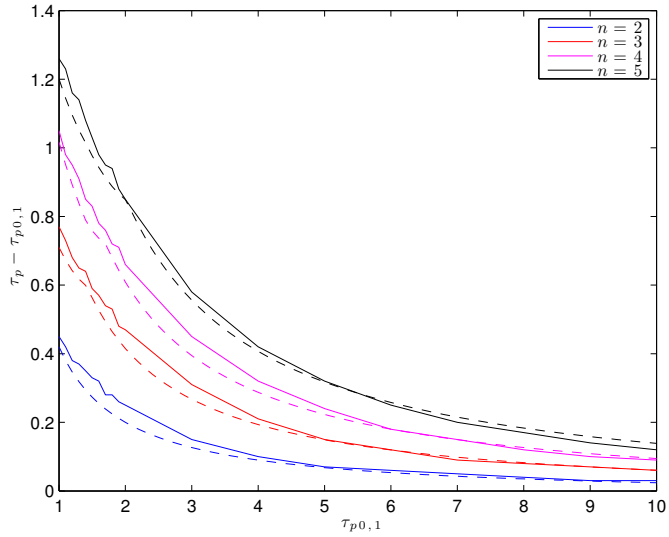


Figure 7.3.: Comparison between the FOPTD approximated lag constant τ_p from lag ratio approximation (striped line) and the constant sum optimization (solid line) for the process $g_{p0} = \frac{1}{(\tau_{p0,1}s+1)(s+1)^{n-1}}$.

This was then adjusted by eye from figure 7.3 to give a more balanced relation between high and low ratio:

$$\frac{\tau_p - \tau_{p0,1}}{\tau_{p0,2}} = 0.44 \cdot \frac{\tau_{p,i+1}}{\tau_{p0,i}} - 0.02 \tag{7.9}$$

A s-curve correlation could give a better fit than the linear for the ratio relation, but it would result in a more complex expression. The s-curve fit could probably handle unsorted processes better than a linear correlation. A s-curve that gives a good correlation is shown as follows:

$$\frac{\tau_p - \tau_{p0,1}}{\tau_{p0,2}} = 1 - \frac{1}{1 + \exp\left(0.77 \cdot \left(\frac{\tau_{p,i+1}}{\tau_{p0,i}} - 1.5\right)\left(\frac{\tau_{p,i+1}}{\tau_{p0,i}}\right)^{-0.5}\right)} \tag{7.10}$$

For simplicity, a linear correlation was chosen. For large differences in the lag constants, the linear correlation approaches zero while the constant sum approximation does not. This will have little effect on the approximation, because the small fraction added from the smallest time constant to the large would be negligible. If the order of the large and small lag constants would change, and the difference between lag constants are big, the

approximated lag constant would be overestimated. To avoid this, a sorting of the lag constants must be done between each order reductions.

With this correlation, a higher order process can be approximated to a lower one by the following *lag ratio approximation*. Given a process:

$$g_{p0} = \frac{1}{\prod_i (\tau_{p0,i} s + 1)} \exp(-\theta_0 s) \tag{7.11}$$

Where the lag time constants are ordered by magnitude, from high to low. The smaller lag constants could be approximated to the second largest time constant by the following relation:

$$\tau_{p,i} \approx \tau_{p0,i} + \left(0.44 \cdot \frac{\tau_{p,i+1}}{\tau_{p0,i}} - 0.02\right) \tau_{p,i+1} \tag{7.12}$$

In the case where i is the index for the smallest lag constant in the original process model, $\tau_{p,i+1}$ becomes $\tau_{p0,i+1}$. This process order reduction is performed for the smaller time constants until the desired order is achieved. The time delay then becomes as follows:

$$\theta \approx \theta_0 + \sum_{i=1}^{n} (\tau_{p0,i}) - \sum_{i=1}^{m} (\tau_{p,i}) \tag{7.13}$$

Where n is the process order for the original process, and m is the order of the approximated process model.

Example: The following process will be approximated as FOPTD model:

$$g_{p0} = \frac{1}{(s+1)^4} \tag{7.14}$$

The first step is the approximation of the fourth order process into a third order, as follows:

$$\tau_{p1,1} = 1 \qquad \tau_{p1,2} = 1 \qquad \tau_{p1,3} \approx 1 + 0.44 - 0.02 = 1.42 \tag{7.15}$$

This new approximated lag constant is larger than the other lags, and the lag constants are re-sorted by magnitude. The process order reduction is then repeated from a third order to second order process:

Table 7.4.: Comparison between the half-rule approximation, lag ratio approximation and the constant sum optimization for 5 processes.

			Approximated model g_p		IAE
Case	Process model g_{p0}	Method	τ_p	θ	$y(g_{p,0}) - y(g_p)$
E1	$\frac{1}{(s+1)(0.2s+1)}$	half-rule	1.1	0.1	0.060
		lag ratio	1.02	0.186	0.022
E2	$\frac{\exp(-0.3s)}{(2s+1)(1s+1)(0.4s+1)(0.12s+1)(0.05s+1)^3}$	half-rule	2.5	1.47	0.230
		lag ratio	2.23	1.74	0.186
E3	$\frac{1.5}{(s+1)(0.1s+1)^2}$	half-rule	1.05	0.15	0.049
		lag ratio	1.01	0.19	0.021
E4	$\frac{1}{(s+1)^4}$	half-rule	1.5	2.5	0.511
		lag ratio	2.02	1.98	0.317
E5	$\frac{1}{(s+1)(0.2s+1)(0.05s+1)(0.008s+1)}$	half-rule	1.1	0.148	0.060
		lag ratio	1.02	0.23	0.023

$$\tau_{p2,1} = 1.42 \qquad \tau_{p2,2} \approx 1 + 0.44 - 0.02 = 1.42 \tag{7.16}$$

The final reduction from second order to first order process is as follows:

$$\tau_{p3,1} \approx 1.42 + 0.44 \cdot 1.42 - 0.02 \cdot 1.42 = 2.02 \tag{7.17}$$

And the time delay becomes as follows:

$$\theta \approx 4 - 2.02 = 1.98 \tag{7.18}$$

The FOPTD approximation would be:

$$g_p = \frac{1}{(2.02s + 1)} \exp(-1.98s) \tag{7.19}$$

The lag ratio approximation gives a very good correlation with the process model, which greatly reduces the IAE compared to the half-rule, as seen from table 7.4.

Simulations where conducted for a number of process models, where the SIMC tuning was applied to the lag ratio method, and compared to the half-rule. An excerpt of the results

Table 7.5.: Comparison between the SIMC tuning performance for the half-rule approximation and the lag ratio approximation.

Case	Process model g_{p0}	Method	M_s	Perfromance		
				$\text{IAE}_r\,(y)$	$\text{IAE}_d\,(y)$	J/J_{opt}
E1	$\frac{1}{(s+1)(0.2s+1)}$	half-rule	1.56	0.355	0.145	1.01
		lag ratio	1.29	0.435	0.373	1.88
E2	$\frac{\exp(-0.3s)}{(2s+1)(1s+1)(0.4s+1)(0.12s+1)(0.05s+1)^3}$	half-rule	1.65	3.57	2.97	1.01
		lag ratio	1.52	3.93	3.55	1.16
E3	$\frac{1.5}{(s+1)(0.1s+1)^2}$	half-rule	1.55	0.401	0.45	1.03
		lag ratio	1.40	0.443	0.582	1.25
E4	$\frac{1}{(s+1)^4}$	half-rule	1.46	5.59	5.4	1.19
		lag ratio	1.57	4.57	4.22	1.07
E5	$\frac{1}{(s+1)(0.2s+1)(0.05s+1)(0.008s+1)}$	half-rule	1.59	0.451	0.296	1.02
		lag ratio	1.36	0.539	0.468	1.43
E4-C	$\frac{1}{(0.5+1)(0.2s+1)}\exp(-0.9s)$	half-rule	1.63	2.18	2.13	1.06
		lag ratio	1.62	2.21	2.16	1.08

are shown in table 7.5. For the majority of the cases, there where a drastic decrease in the M_s values. The exception was for processes with equal time lag constants, as for the E4 case, where the M_s value increased. The increased M_s value lead to a deterioration of the performance and, compared with the optimal tuning performance J_{opt} for the respective tuning, a deterioration in the tuning performance J.

The lag ratio method gave for most process a very robust setting with sluggish response. The worst case was E1 where there was a non-optimality loss ($J/J_{\text{opt}} \cdot 100$) of 88%. Because the SIMC rule should give a fast and robust tuning when applied to a process model (with standard tuning factor), the lag ratio method is not recommended for processes when there is a large difference in the lag constants.

The increase in M_s value gives a tuning performance J that is closer to the optimal at the respective M_s. For process E4, the non-optimality loss decrease from 19% deviation to 7%. As mentioned earlier, there was a strong indication that less than optimal performance for these processes was caused by the half-rule. The lag ratio approximation gives an non-optimality loss similar to the E4-C process, as would be expected from a good aproximation. One can then conclude that there is a weakness in the half-rule approximation for processes with equal lag constants, and one could use the lag ratio approximation or the half-rule addendum to correct for this.

The lag ratio approximation takes the the relative size difference of the lag constants into account. It will therefore give a better approximation than the half-rule addendum when

there is small deviations in the close-to equal lag constants. The lag ratio approximation will also give a better approximation when the order of the process is high.

8. Setpoint Overshoot Method

The new setpoint overshoot method developed by Shamsuzzoha at al. has offered fast closed-loop alternative to model approximation. There is great interest in the possibility of using this method to approximate FOPTD model for the SIMC method. As mentioned before, the presented method inconsistently uses two different time delay approximation in the derivation of the process model. The objective was then defined as:

- Find a good solution for approximating time delay for both delay dominated and lag dominated regions.
- Determine if the setpoint overshoot method give good results when used with the SIMC tuning rules.

To determine how the relation between the effective time delay θ and the time to reach first peak t_{peak} varies with the process model, a setpoint overshoot experiment was conducted for the process:

$$g_p(s) = \frac{1}{\tau_p s + 1} \exp(-s) \tag{8.1}$$

Where the lag time constant was:

$$\tau_p \in \{0.1, 0.2, \ldots, 0.9, 1, 2, \ldots 99, 100\} \tag{8.2}$$

The overshoot was adjusted to the recommended value of 0.30 for all processes. The results can be seen in figure 8.1, and a good correlation was found to be:

$$\frac{\theta}{t_{\text{peak}}} = 0.309 + 0.209 \exp\left(-0.610 \cdot \frac{\tau_p}{\theta}\right) \tag{8.3}$$

From the figure, it can be seen that Shamsuzzoha's time delay approximation $\theta/t_{\text{peak}} = 0.305$ fits nicely with it's respective lag dominated region ($\tau_p/\theta \geq 8$). For the delay dominated region ($\tau_p/\theta < 8$), there is a large variation in the θ/t_{peak} relation, and Shamsuzzoha's time delay approximation $\theta/t_{\text{peak}} = 0.43$ is not very accurate. Two alternative methods can then be used for the time delay approximation:

1. Use the continuous time delay correlation in equation 8.3

2. Determine the process domain and use the appropriate Shamsuzzoha time delay approximation.

The setpoint overshoot method was conducted for the different time delay approximations with process models ranging from time delay dominated to lag dominated. The overshoot was adjusted to 0.3 and the IAE was calculated for the difference in response between the real and approximated process models. The results are summarized in table 8.1. The continuous correlation gave the best result for all processes except the $\tau_p/\theta = 1$ case where it was slightly less accurate than the best approximation $\theta = 0.43t_{\text{peak}}$. Shamsuzzoha's time delay approximations gave good results for their respected regions.

The models can be approximated by using Shamsuzzoha's time delay approximations or the continuous time delay correlation. For a more precise model approximation in the delay dominated region, use the continuous time delay correlation from equation 8.3 and determine the model by the following steps:

Step 1 Perform the setpoint overshoot experiment and obtain the key parameters.

Step 2 Determine the process domain by solving equation 2.19 with respects to $\frac{\tau_p}{\theta}$:

- $\frac{\tau_p}{\theta} = 2 \cdot A \cdot \left|\frac{b}{1-b}\right|$

Step 3 Determine the time delay with the continuous time delay correlation (equation 8.3):

- $\frac{\theta}{t_{\text{peak}}} = 0.309 + 0.209 \exp\left(-0.610 \cdot \frac{\tau_p}{\theta}\right)$

Step 4 Determine the process gain k_p and the lag time constant τ_p (equation 2.15 and 2.16 repectivly):

- $\tau_p = 2\theta \cdot A \cdot \left|\frac{b}{1-b}\right|$
- $k_p = \frac{1}{k_{c0}} \cdot \left|\frac{b}{1-b}\right|$

To avoid additional complicated expressions, Shamsuzzoha's time delay approximation can be used to determine the model by replacing step 3 in the method above with:

Step 3 Determine the time delay:

- if $\frac{\tau_p}{\theta} < 8$ use $\theta = 0.43t_{\text{peak}}$

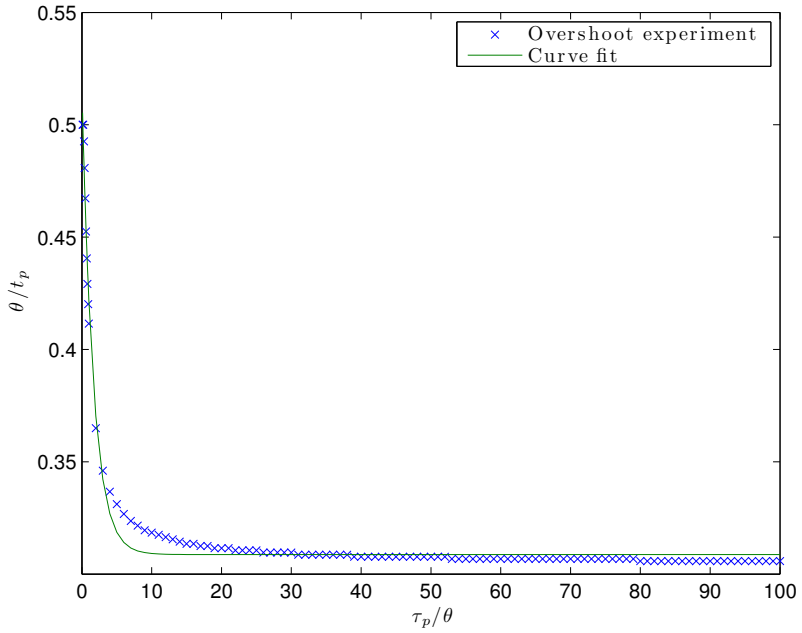


Figure 8.1.: Ratio of the time delay(θ) and the time to reach firs peak(t_{peak}) as a function the process $g_p(s) = \frac{1}{\tau_p s+1}\exp(-s)$ with an overshoot of 0.3. The solid line is the best fit exponential curve.

Table 8.1.: The IAE between the original model g_{p0} and the overshoot approximated model g_p for the three time delay approximations.

	IAE $= \int \lvert y(g_{p0}) - y(g_p)\rvert$		
Process model $g_{p0}(s)$	Continuous correlation	$\theta = 0.43t_{\text{peak}}$	$\theta = 0.305t_{\text{peak}}$
$\frac{1}{(0.1s+1)}\exp(-s)$	0.01	0.16	0.43
$\frac{1}{(s+1)}\exp(-s)$	0.19	0.17	0.46
$\frac{1}{(8s+1)}\exp(-s)$	1.63	5.68	1.41
$\frac{1}{(100s+1)}\exp(-s)$	35.6	89	33.8
$\frac{1}{(1000s+1)}\exp(-s)$	134	261	129

- if $\frac{\tau_p}{\theta} \geq 8$ use $\theta = 0.305 t_{\text{peak}}$

The continuous time delay correlation was used with the two-step overshoot method for 13 processes, and the results are shown in table 8.2. The overshoot model approximation is compared with the half-rule approximation, and the resulting controller performance and robustness are compared when the SIMC rule is applied. The one-step controller tuning is also included in the table.

The model approximations was similar to the half-rule approximations, but in the majority of processes tested, the two-step procedure seemed to overestimate the process lag constants. The time delay approximation was for most cases good, and there was little deviation. The worst approximations compared to the half-rule was the E4 and E10 case. The E4 approximation of the time delay $\theta = 2.05$ is probably good, because the lag-ratio approximation indicates that the time delay approximation should be $\theta = 1.98$.

The one- and two-step controller settings where very similar, giving almost the same performance and robustness. This is to be expected because the two methods uses the same overshoot correlations. Some small deviations between the one- and two-step procedures can be seen, but these are most likely caused by the use off different time delay approximations.

Table 8.2.: Comparison between SIMC tuning performances when applied to two-step and half-rule approximation, and one-step tuning rules for 13 processes.

									Preformance			
			Approximation $g_p(s)$			Controller setting			Setpoint response		Disturbance response	
Case	Process model $g_{p_0}(s)$		Half-rule	Two-step		k_c	τ_i	M_s	IAE$_r(y)$	TV$_r(u)$	IAE$_d(y)$	TV$_d(u)$
E1	$\frac{1}{(s+1)(0.2s+1)}$	k_p	1	1	half-rule	5.5	0.8	1.56	0.355	12.7	0.145	1.55
		τ_p	1.1	2.04	Two-step	8.12	1	1.67	0.304	20.4	0.124	1.7
		θ	0.1	0.126	One-step	8.12	0.991	1.68	0.305	20.4	0.122	1.71
E2	$\frac{(-0.3s+1)(0.08s+1)}{(2s+1)(1s+1)(0.2s+1)(0.05s+1)^3}$	k_p	1	1	half-rule	0.85	2.5	1.66	3.57	1.9	2.97	1.26
		τ_p	2.5	3.12	Two-step	0.926	3.12	1.61	3.45	1.86	3.38	1.15
		θ	1.47	1.69	One-step	0.926	3.57	1.56	3.85	1.75	3.86	1.09
E3	$\frac{2(15s+1)}{(20s+1)(s+1)(0.1s+1)^2}$	k_p	1	2	half-rule	2.33	1.05	1.55	0.464	4.97	0.453	1.29
		τ_p	1.05	1.97	Two-step	3	1.31	1.68	0.436	6.97	0.44	1.46
		θ	0.15	0.164	One-step	3	1.3	1.69	0.435	6.98	0.434	1.46
E4	$\frac{1}{(s+1)^4}$	k_p	1	1	half-rule	0.3	1.5	1.46	5.59	1.15	5.4	1.1
		τ_p	1.5	3.17	Two-step	0.768	3.17	1.6	4.13	1.55	4.12	1.13
		θ	2.5	2.06	One-step	0.768	3.48	1.55	4.54	1.47	4.54	1.09
E5	$\frac{1}{(s+1)(0.2s+1)(0.04s+1)(0.0008s+1)}$	k_p	1	1	half-rule	3.72	1.1	1.59	0.45	8.16	0.296	1.41
		τ_p	1.1	1.6	Two-step	4.16	1.54	1.59	0.464	9.23	0.371	1.39
		θ	0.148	0.193	One-step	4.16	1.52	1.6	0.461	9.24	0.365	1.4
E6	$\frac{(0.17s+1)^2}{s(s+1)^2(0.028s)}$	k'_p	1	0.652	half-rule	0.296	13.5	1.48	6.5	0.67	45.6	1.55
				(i)	Two-step	0.495	12.4	1.76	4.75	1.29	25	1.8
		θ	1.69	1.55	One-step	0.495	12.2	1.77	4.75	1.29	24.7	1.8
E7	$\frac{-2s+1}{(s+1)^3}$	k_p	1	1	half-rule	0.214	1.5	1.66	7.28	1.05	8.34	1.28
		τ_p	1.5	1.37	Two-step	0.244	1.37	1.96	7.28	1.36	8.24	1.62
		θ	3.5	2.79	One-step	0.244	1.27	2.12	7.67	1.55	8.6	1.81
E8	$\frac{1}{s(s+1)^2}$	k'_p	1	0.729	half-rule	0.33	12	1.76	6.47	0.843	36.4	1.78
				(i)	Two-step	0.354	15.5	1.74	6.31	0.886	43.8	1.7
		θ	1.5	1.94	One-step	0.354	15.3	1.74	6.31	0.887	43.3	1.7
E9	$\frac{1}{(s+1)^2}e^{-s}$	k_p	1	1	half-rule	0.5	1.5	1.61	3.38	1.32	3.14	1.15
		τ_p	1.5	1.89	Two-step	0.59	1.89	1.59	3.21	1.29	3.2	1.06
		θ	1.5	1.6	One-step	0.59	1.98	1.57	3.35	1.24	3.35	1.03

Table 8.2 Continued

Case	Process model $g_{p_0}(s)$		Approximation $g_p(s)$			Controller setting			Preformance			
									Setpoint response		Disturbance response	
			Half-rule	Two-step		k_c	τ_i	M_s	IAE$_r(y)$	TV$_r(u)$	IAE$_d(y)$	TV$_d(u)$
E10	$\frac{1}{(20s+1)(2s+1)}e^{-s}$	k_p	1	1	half-rule	5.25	16	1.72	6.34	12.3	3.05	1.49
		τ_p	21	26	Two-step	4.96	20.9	1.62	5.92	11	4.22	1.33
		θ	2	2.62	One-step	4.96	20.6	1.62	5.92	11	4.16	1.34
E11	$\frac{(-s+1)}{(6s+1)(2s+1)^2}e^{-s}$	k_p	1	1	half-rule	0.7	7	1.63	11.5	1.59	10.1	1.2
		τ_p	7	8.58	Two-step	0.787	8.58	1.61	11.1	1.61	10.9	1.12
		θ	5	5.45	One-step	0.787	9.49	1.56	12.1	1.52	12.1	1.07
E12	$\frac{(6s+1)(3s+1)}{(10s+1)(8s+1)(s+1)}e^{-0.3}$	k_p	0.225	1	half-rule	7.4	1	1.66	1.07	18.3	0.148	1.39
		τ_p	1	4.84	Two-step	9.22	2.1	1.74	0.919	21.5	0.236	1.26
		θ	0.3	0.263	One-step	9.22	2.07	1.74	0.919	21.5	0.233	1.26
E13	$\frac{2s+1}{(10s+1)(0.5s+1)}$	k_p	0.625	1	half-rule	2.88	4.5	1.74	2.86	6.56	1.61	1.2
		τ_p	4.5	4.08	Two-step	2.95	4.08	1.77	2.77	6.9	1.45	1.23
		θ	1.25	0.692	One-step	2.95	5.37	1.76	2.88	6.61	1.84	1.2

(i) k_p' is the process gain for an integrating process $g_p(s) = \frac{k_p}{\tau_p s} = \frac{k_p'}{s}$

9. Summary and discussion

9.1. Tuning rules

Three varieties of SIMC has been investigated in this report, but how can one choose the best rule? A good tuning rule should fulfill most of the following criterias: give good robustness (small M_s), give little non-optimality loss, give good disturbance response (setpoint are usually constants, and is therefore no as important) and be simple to use and calculate.

The SIMC rule generally gives good robustness. Though the M_s increases for lag dominated processes, it is necessary for good performance. Haugen's ($C = 2$) SIMC results in a high M_s for lag dominated process, boarding on the upper limits of robustness. This test was conducted for a integrating process, without model approximation. Thus this increase in M_s is most likely to be amplified with the uncertainty in the model approximations. Haugen reported that this had only a minor effect on the gain margin. But, as Haugen did not report, the increase in integral action results in a severe deterioration of the delay margin (see appendix B for calculation). This decrease in delay margin is then reflected in the increased M_s value.

For I-SIMC the M_s value is increased for almost all processes, as is to be expected, because it was designed to give a slightly higher M_s value than SIMC. Though the M_s is increased, it is still lower than the M_s value resulting from Haugen's ($C = 2$) SIMC. Though this is not shown explicitly in the report, it can be justified by two arguments: The integral action is not increased as much as Haugen's proposal (thus the M_s is not increasing as much), and the controller gain is reduced (reducing the M_s).

Both SIMC and I-SIMC gave little non-optimality loss for the majority of processes, and usually ranging between 1 and 4% loss. The I-SIMC had lower or equal non-optimality loss for for all processes except processes E1, E6 and E8. Here SIMC had little or no non-optimality loss, and achieved the best compromise. SIMC and I-SIMC had both large non-optimality loss for the processes E4, E13 and E14, but compared to SIMC, I-SIMC reduced the non-optimality loss for processes E4 and E14 by 11 and 17 percentage points receptively. The large non-optimality loss for E4 is caused by a weakness in the half-rule approximation of the lag constants, and not by the SIMC. When SIMC was applied to a better approximation of E4, the non-optimality loss was reduced from 19 to 7%.

In most situations, a good disturbance response is prioritized more than a good setpoint response. This is because for most processes the setpoint is fixed and, for setpoint changes, a prefilter can be added to give excellent setpoint response. It is also not desired to have a controller that is tuned solely for the best disturbance response scenario. This is because there can be disturbances close to the process output (can be thought of as a unpredicted setpoint change, and a prefilter will not help), and one would want a satisfactory response. For delay dominated processes, the optimal disturbance and setpoint response merges, and thus the tradeoff issue is not relevant. For lag dominated processes, the I-SIMC rule gives better disturbance response, and generally have an integral time closer to the integral time for the best disturbance response.

The simplicity of the tuning rule is also important. The SIMC rule is very simple, easy to use and remember. The I-SIMC rule is complicated, and contains more empirical relations. Therefore the rule is not as simple as SIMC and is much harder to remember. The rule itself is easy to use, and one need only to insert the model approximations, as for SIMC.

9.2. Improved lag approximation

A good model approximation must retain the main characteristics of the original model, resulting in good performance and robustness when the tuning rule is applied. The approximation methods presented in this report: the half-rule addendum and the lag ratio approximation, works both well for processes with equal lag constants, and gives better tuning results than the half-rule. For processes with unequal lag constants, the half-rule performs better than the two proposed methods.

The ratio lag approximation gives smaller deviation between the model and the approximation than the half-rule. Nevertheless, when the tuning rule is applied to processes with equal lag constants, there is a deterioration of the performance. Thus minimizing the error between the process model and the approximation does not guarantee good tuning results.

Simplicity is also a key factor when approximating models, and the half-rule addendum uses a simple relation with only two calculations. The lag ratio reduction uses a more complex empirical correlation and requires a number of calculations equal to the reduction in order.

9.3. Setpoint overshoot model approximation

The model approximation resulting from the proposed corrected two-step overshoot procedure, in combination with SIMC, yields very similar results as the one-step overshoot

method. Thus, if the opinion is that the one-step overshoot method gives good tuning results, the two-step method will also give good results.

The one-step method yields the controller tuning directly from the setpoint experiment. This resembles a "black box" operation, and without a thorough review of the experimental foundation for the method, the tuning parameters appears like "magic".

By first finding an approximated model and then applying the tuning rules, the two-step procedure gives the user gets an indication of the process dynamics, and the option to apply a familiar or even favorite tuning rule. Thus the user gets more insight and influence over the tuning process.

The one-step procedure is designed to give close to SIMC tuning, and because the two-step procedure are derived from the same relations, it will be reflected in the approximated model. Therefore, an overshoot approximated model might not give the same good tuning results when used with a different tuning rule, as when used with SIMC.

10. Conclusion

The SIMC rule has been found to give close-to optimal tuning for most processes, but potential improvements could be made to enhance performances for delay dominated and lag dominated processes. These improvements resulted in the improved-SIMC rule mentioned below.

Haugen's ($C = 2$) SIMC rule is not recommended, as it results in a high M_s for very lag dominated and integrating processes that borders on the upper levels of robustness.

A new improved-SIMC rule is proposed for PI-tuning. This improved rule is designed to give better performance for delay dominated processes ($\tau_p/\theta < 1$), and better disturbance response for lag dominated processes($\tau_p/\theta \geq 6$). For intermediate processes ($1 \geq \tau_p/\theta < 6$), the I-SIMC gives the same results as the original SIMC. The I-SIMC rule is as follows:

Delay dominated region $\tau_p/\theta < 1$	intermediate and lag dominated region $\tau_p/\theta \geq 1$	Integrating process $\tau_p \to \infty (i)$
$k_c = \frac{\tau_p}{(\tau_c+\theta)} \frac{1}{k_c} + \frac{0.3}{k_p}\left(1 - \frac{\tau_p}{\theta}\right)$	$k_c = \min\left[\ \frac{1}{k_p}\frac{\tau_p}{(\tau_c+\theta)}\ \ ,\ \ \frac{0.18}{kp} + \frac{0.94}{k_p}\frac{\tau_p}{(\tau_c+\theta)}\ \right]$	$k_c = \frac{0.94}{k'_p}\frac{1}{(\tau_c+\theta)}$
$\tau_i = 0.15\left(\tau_c + \theta\right) + 0.7\tau_p$	$\tau_i = \min\left[\tau_p, 3\left(\tau_c + \theta\right)\right]$	$\tau_i = 3\left(\tau_c + \theta\right)$

(i) Integrating process ($\tau_p \to \infty$): $g_p = \frac{k_p}{(\tau_p s+1)} \approx \frac{k_p}{\tau_p s} = \frac{k'_p}{s}$

Where τ_c is the tuning factor, with a recommended value of $\tau_c = \theta$.

The cause of the poor SIMC tuning performance for higher order processes with equal lags is found to be caused by the half-rule approximations of the lag constants. Two methods are proposed for improving the approximation: The half rule addendum and the lag ratio approximation.

The half-rule addendum is a simple extension of the original half-rule and are computed as follows. Given the process model, where the lag constants are ordered by magnitude from large to small:

$$g_{p0} = \frac{1}{\prod_i (\tau_{p0,i} s + 1)} \exp(-\theta_0 s)$$

The approximated lag constant τ_p and effective delay θ can be calculated:

$$\tau_p \approx \tau_{p0,1} + 0.5\tau_{p0,2} + \sum_{i \geq 3} 0.3\tau_{p0,i}$$

$$\theta \approx \theta_0 + 0.5\tau_{p0,2} + \sum_{i \geq 3} 0.7\tau_{p0,i}$$

The lag ratio approximation gives little deviation between the original model and the approximated one, but the resulting tuning have poor performance if used on a process with unequal lag constants. The lag ratio approximation can be calculated as follows. From the process given above, the model can be reduced one lag constant at the time with the following approximation:

$$\tau_{p,i} \approx \tau_{p0,i} + \left(0.44 \cdot \frac{\tau_{p,i+1}}{\tau_{p0,i}} - 0.02\right) \tau_{p,i+1}$$

In the case where i is the index for the smallest lag constant in the original process model, $\tau_{p,i+1}$ becomes $\tau_{p0,i+1}$. Between each reduction, the lags are sorted by size. After the model has been reduced to the desired order, the effective delay can be calculated by:

$$\theta \approx \theta_0 + \sum_{i=1}^{n} (\tau_{p0,i}) - \sum_{i=1}^{m} (\tau_{p,i})$$

Where n is the process order for the original process, and m is the order of the approximated process model. Both the half-rule addendum and the lag ratio approximation can only be used for higher order processes with equal lag constants.

A correction to the inconsistency in the two-step setpoint overshoot method has been proposed, with a new continuous correlation for the effective time delay approximation. This corrected two-step method give very similar tuning results to the one-step overshoot method when the SIMC rule is applied. The corrected method is as follows:

Step 1 Perform the setpoint overshoot experiment and obtain the key parameters.

Step 2 Determine the process domain by solving: $\frac{\tau_p}{\theta} = 2 \cdot A \cdot \left| \frac{b}{1-b} \right|$

Step 3 Determine the time delay with the continuous time delay correlation:

$$\frac{\theta}{t_{\text{peak}}} = 0.309 + 0.209 \exp\left(-0.610 \cdot \frac{\tau_p}{\theta}\right)$$

Step 4 Determine the process gain k_p and the lag time constant τ_p:

$$\tau_p = 2\theta \cdot A \cdot \left|\frac{b}{1-b}\right| \qquad k_p = \frac{1}{k_{c0}} \cdot \left|\frac{b}{1-b}\right|$$

Bibliography

[1] Skogestad S. Simple Analytic Rules for Model Reduction and PID Controller Tuning. Modeling, Identification and Control. 2004;25(2):85–120.

[2] Haugen F. Comparing PI Tuning Methods in a Real Benchmark Temperature Control System. Modeling, Identification and Control;31(3):79–91.

[3] Shamsuzzoha M, Skogestad S. The Setpoint Overshoot Method: A Simple and Fast Closed-Loop Approach for PID Tuning. Journal of Process Control. 2010;20:1220–1234.

[4] Seborg DE, Edgar TF, Mellichamp DA. Process Dynamics and Control. New York: John Wiley & Sons; 1986.

[5] Skogestad S, Postlethwaite I. Multivariable Feedback Control – Analysis and Design. 1st ed. New York: John Wiley & Sons; 1996.

A. Simulink block diagrams and MatLab codes

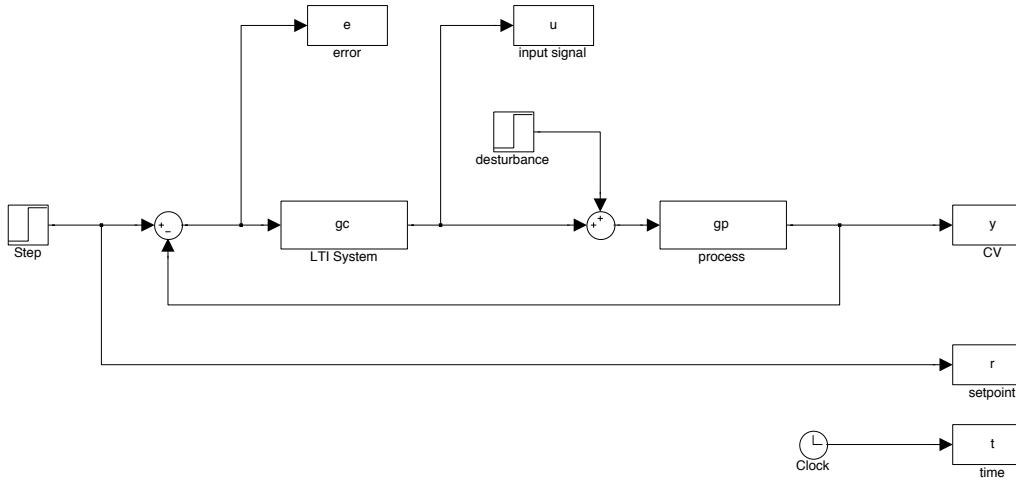


Figure A.1.: The Simulink block diagram

A.1. Function for calculating M_s

```
% ------------------------------------------------------------------------ %
% Function for calculaing the M_s for a process and tuning.                %
%                                                                          %
%                            Original function made by: Sigurd Skogestad   %
%                            Modified by: Chriss Grimholt, to handel       %
%                                         transfer function inputs.        %
%                                                   Made: September 2010   %
% ------------------------------------------------------------------------ %

function MS_S = ms(gp, kc, ti)

% gp is the process transfer function, kc is the controller gain
% and ti is the integral time

[gp_n gp_d] = tfdata(gp,'v');

delay = totaldelay(gp);

w = logspace(-4,4,4000);

Gol_n_s = conv([ti*kc kc],gp_n);
Gol_d_s = conv([ti 0],gp_d);

Gol_n_w = polyval(Gol_n_s,w*1i);
Gol_d_w = polyval(Gol_d_s,w*1i);
delay_w = exp(-delay*w*1i);

Gol = Gol_n_w./Gol_d_w.*delay_w;

S = 1./abs(1+Gol);
MS_S = max(S);
end
```

A.2. Determining k_c and τ_i relation for a given M_s

```
% ------------------------------------------------------------------- %
% Function for finding the controller kc and the integral time ti for a  %
% process transfer function gp with a given ms.                          %
%                               Made by Chriss Grimholt, September 2010  %
% ------------------------------------------------------------------- %


function [ti_sim kc] = optimaltuning(gp, ti_sim, kc0, ms_s)

kc = [];

for n = 1:size(ti_sim,1)

    ti=ti_sim(n);

    [kc_temp, fval] = fsolve(@(kc) myfun(gp,kc,ti,ms_s),kc0);

    kc0=kc_temp;

    kc=[kc; kc_temp];

end

end


% ------------------------------------------------------------------- %
% Function for calculating the controller gain and integral time that %
% satisfies the ms requirement.                                       %
%                             Made by Chriss Grimholt, September 2010  %
% ------------------------------------------------------------------- %


function fun = myfun(gp, kc, ti, ms_s)

ms_s = ms(gp,kc,ti)
fun = ms_s-max(S);

end
```

B. Calculation of delay margin for Haugen's experiment

The process model in Haugen's experiment:

$$g_p = \frac{5.7}{(60s+1)} \exp(-4s) \tag{B.1}$$

The open-loop system

$$y = g_c g_p \cdot r$$

The controller:

$$g_c = k_c \left(1 + \frac{1}{\tau_i s}\right) = \frac{k_c}{\tau_i s} (\tau_i s + 1)$$

The process:

$$g_p = \frac{k_p}{\tau_p s + 1} e^{-\theta s}$$

The phase angle for the system:

$$\phi = -90 + \tan^{-1}(\tau_i \omega) + \tan^{-1}(-\tau_p \omega) - \theta\omega$$

At the phase margin frequency (ω_{pm}), the aplitude ratio (AR) is 1

$$\mathrm{AR} = \frac{k_c k_p}{\tau_i \omega_{pm}} \cdot \frac{\sqrt{\tau_i^2 \omega_{pm}^2 + 1}}{\sqrt{\tau_p^2 \omega_{pm}^2 + 1}} = 1$$

Solving with regards to ω_{pm}

$$\left(\frac{\tau_i\tau_p}{k_ck_p}\right)^2\omega_{pm}^4+\left(\frac{1}{(k_ck_p)^2}-1\right)\tau_i^2\omega_{pm}^2-1=0$$

$$\omega_{pm}^4+\left(\frac{1-k_c^2k_p^2}{\tau_p^2}\right)\omega_{pm}^2-\left(\frac{k_ck_p}{\tau_i\tau_p}\right)^2=0$$

substituting ω_{pm}^2 with ω_{pm}^*

$$\omega_{pm}^2=\omega_{pm}^*$$

$$\omega_{pm}^{*2}+\left(\frac{1-k_c^2k_p^2}{\tau_p^2}\right)\omega_{pm}^*-\left(\frac{k_ck_p}{\tau_i\tau_p}\right)^2=0$$

Solving with regards to ω_{pm}^* and choosing only the positive answer

$$\omega_{pm}^*=-\frac{a}{2}+\frac{1}{2}\sqrt{a^2-4b}$$

Using the following controller parameters for k_c and τ_i:

Table B.1.: The controller and process parameters

k_c	$\frac{\tau_p}{k_p(\tau_c+\theta)}$	k_p	$5.7\,\mathrm{degC/V}$
τ_i	$\min\left[\tau_p, c\left(\tau_c+\theta\right)\right]$	τ_p	$60\,\mathrm{s}$
τ_c	θ	θ	$4\,\mathrm{s}$

Calculating the values for ω_{pm}^* for values of $C=[2,4]$ and finding the values for ω_{pm}

$$\omega_{pm}=\sqrt{\omega_{pm}^*}$$

Finding the phase angle at the frequency ω_{pm}

$$\phi_{pm} = -\frac{\pi}{2} + \tan^{-1}(\tau_i \omega_{pm}) - \tan^{-1}(\tau_p \omega_{pm}) - \theta \omega_{pm}$$

The phase magin (PM) is defined as:

$$\text{PM} = \phi_{pm} + \pi$$

And the delay margin (DM) is defined as:

$$\text{DM} = \frac{\text{PM}}{\omega_{pm}}$$

The final calculated results becomes:

Table B.2.: The results from using Skogestad's SIMC tuning rules with the factor $C = 4$ and Haugen's improved desturbance compansating factor $C = 2$.

	$C = 2$	$C = 4$
ω_{pm}	0.063	0.034
ϕ_{pm}	-2.024	-1.995
PM	1.118	1.147
DM	17.89	33.37
M_s	1.80	1.63