Automatic Fuzzy Tuning of Proportional-integral Controllers Based on Time-domain Specifications

Emad Ali

Chemical Engineering Department,
King Saud University, P.O. Box 800, Riyadh 11421, Saudi Arabia
Email: amkamal@ksu.edu.sa, fax:++(9661)467-8770

Abstract. In this paper, an online tuning method based on time-domain performance specification is proposed to determine the parameters of standard PI controllers. The method uses a process model to predict the future output and to detect the specification violation. The maximum performance violation along with general tuning guidelines formulated as fuzzy sets and rules are used to find the new PI parameter values. The procedure is repeated each sampling time to provide continuous automatic tuning of the PI controller so that it can preserve good performance over a wide range of operating conditions. The algorithm allows the usage of any form of models and uses four simple fuzzy sets with eleven simple fuzzy rules to maintain simplicity and minimum computational effort. Numerical testing of the algorithm on a CSTR and on an evaporator example shows that better performance than what can be achieved by standard PID controllers for both set point change and load disturbance.
Keywords: PI controller, Self-tuning, Fuzzy rules, Performance specification.

Nomenclature

a	Coefficient for the linear model
A	Scaled measure for the upper bound violation
$A_{i,j}$	Pre-calculated constant matrix
b	Coefficient for the linear model
B	Scaled measure for the lower bound violation
C	Scaled measure of the violation rate, also a constant matrix
C_A, C_B, C_{Af}	Concentration of the reactant A, product B, and the feed, kmole/m^3
C_1, C_2	Feed and Product composition in percentage, respectively
C_p	Heat capacity for cooling water, kW/kg min
d	Estimated of model-plant mismatch
e	Error signal
F	Feed flow rate m^3/min
F_1, F_2	Feed and product flow rates, kg/min

F_3, F_4, F_5	Recycle, vapor and condensate flow rate, kg/min
F_{200}	Cooling water flow rate, kg/min
K	Sampling time
k_1, k_2, k_3	Reaction rate constants for the CSTR, 1/hr, 1/hr, 1/kmole hr.
k_c, k_I	PI parameters; controller gain and reciprocal integral time, respectively
K_c, K_I	Diagonal matrices of controller gain and reciprocal integral time
M	Constant
n_1, n_2, n_3	Dimension of the three zones of the performance envelope
n_R, n_f	Number of fuzzy rules and number of fuzzy functions
P_w	Closed-loop prediction horizon
P_{100}, P_2	Steam and product pressure, respectively, kPa
T	Time
T_1, T_{200}	Feed and Cooling water temperature, ^{o}C
U	Manipulated variable
UA	Product of heat transfer coefficient and the transfer area, kW/K.
V	Reactor volume, m^3
y, y_r	Vector of outputs and set points, respectively
y_P	Plant output
y^l, y^u	Lower and upper bounds on y
W	Multiplication factor for the tuning parameters
z, z_e	State space vectors

Greek letters

δ	Pre-calculated constant factor
μ	Membership function
τ_I	Integral time constant
λ_{s1}	Latent heat of steam at saturation condition, kW/Kg min.
λ_{s2}	Latent heat of evaporation of water, kW/kg min.

Introduction

Conventional PID controllers have been and are still widely used in the chemical industry. PID controllers are simple, easy to understand and implement in hardware and software, and do not require a process model for initialization or operation. However, frequent fine-tuning of the PI controller for good performance and/or stability is essential especially for highly non-linear and/or coupled processes. In real applications, tuning of a PID is usually a cumbersome experience. Many efforts dealing with tuning of such controllers for better performance have been reported. A review of these methods can be found in earlier research works [1-4] or in survey papers [5, 6]. These methods can be classified under three categories; reaction curve method, continuous oscillation method and frequency domain method. Despite the variation between these methods, they are all implemented offline and applied for SISO control loops. In real practice, most of the chemical processes are multivariable with strong cross-loop interaction. In this case, if

each SISO loop in the process is tuned individually, the overall process performance may deteriorate due to interaction brought in by closing all loops simultaneously. Moreover, the process may operate at different operating conditions either voluntarily due to grade changeover or inevitably due to severe load changes. In this case, a SISO loop tuned for a specific operating condition may degrade for another condition. Therefore, controllers tuned using the aforementioned methods may not necessarily provide stability or good performance over wide range of operating conditions.

To deal with the tuning problem, a self-tuning procedure is required. For this reason, Ali [1-4] has developed an automatic (continuous) self-tuning PI controller. The PI controller parameters are tuned such that the resulted closed-loop response satisfies a certain time-domain specification. The proposed tuning method is model based. The model is used for two purposes. One purpose is to predict the future dynamics of the process and to detect whether violation of the time-domain specification occurs or not. Another purpose is to obtain the sensitivity of the closed-loop response with respect to the PI settings. The sensitivity expression is the heart of the tuning method. In this paper, the sensitivity expressions are eliminated. Instead, fuzzy sets and rules [7] are used. This modification will add some advantages. First, the reliance on the process model is reduced because the model will be used only for prediction and detection. Secondly, the design effort will be reduced knowing that the sensitivity expressions are sometime difficult to obtain and/or that the solution of the sensitivity increases computational load.

Many efforts dealing with tuning the PI controllers using fuzzy logic have been reported. Bandyopadhyay and Patranabis [8] proposed an autotuner for PI controllers based on fuzzy logic. The method computes each sampling time new PI parameters using a scaling factor to provide dead-beat performance. The scaling factor depends on expert knowledge and thus, it is not clear how to extend the method to different type of processes. Xu *et al*. [9] and Rajani *et al*. [10] developed a tuning scheme based on fuzzy logic. However, the control law and structure are based on PID-type fuzzy controller and not on a standard PI algorithm formulation. Miler *et al*. [11] proposed a PID tuning strategy based on fuzzy logic to control a specific bioreactor. The proposed method tunes the PID parameters online each sampling time using fuzzy set and rules. However, the fuzzy rules are generated from expert knowledge of the oxygen consumption in the reactor. Therefore, this method can not be generalized easily to other processes. Fuzzy logic theory is also used to design a gain scheduler for PID controllers [12, 13]. In these methods, fuzzy rules are used to switch or interpolate between pre-calculated values for the PI parameters when the process moves from one operating condition to another. The drawback of this approach is that different operating regions for the process should be defined and subsequently the PID controller should be well designed for each region beforehand. Chan [14] developed an automatic tuner for PID controllers based on fuzzy rules and sets. However, the proposed method requires training the fuzzy rules using different closed-loop responses generated at different sets of PI parameter values. Ramkumar and Chidambaram [15] suggested another fuzzy self-tuning PI controller. The

tuning algorithm is based on the output error signal and not on a certain performance specification. Moreover, the rules base is generated from iterative simulation procedure implemented on the given process. Therefore, it may not work well for another process. He *et al*. [16] proposed another fuzzy self-tuning method. The method is based on parameterizing the Ziegler-Nichols tuning formula by a single parameter. A fuzzy inference mechanism is used to self-tune the new parameter. The method is tested only on simple linear models. Moreover, the parameters are tuned to speed up the response, but no specific time-domain performance is used to shape the whole response. Pfeiffer and Isermann [17] have also proposed a self-tuning PI controller using fuzzy logic. The two parameters of the PI controller are adapted based on two fuzzified variables of the measured overshoot and settling ratio. The method requires the current signal of the control error to estimate the two fuzzified variables. In this case, the method lacks prediction capability. Consequently, the PI parameters are adapted only when some loss in the performance has occurred.

The proposed tuning method in this paper differs from those reported methods in three aspects. First, the proposed method adapts the PI parameters to obtain a certain *time-domain* specification, which is more appealing to a practitioner than other sophisticated specifications such as gain/phase margins, frequency-domain specifications, pole placement, ISE, complicated objective functions, etc. The time-domain performance envelope also allows for easy visualization of the goodness of the response and thus the operator can adjust his performance envelope online for possible trade-off. Secondly, the tuning method is based on general and well-known tuning guidelines rather than heuristic and/or expert knowledge. Thirdly, the tuning method uses future predictions of the output response, which permits advance correction of the PI parameters. This feature provides enhanced tuning especially for processes with dead-time or inverse response.

The objective of this paper is to develop an online automatic tuning method for the conventional PI controllers. The online feature of the proposed method allows the continuous adaptation of the PI settings to preserve its performance when the process undergoes through different operating conditions. The new method combines model-based and rule-based tools to develop a self-tuning scheme. The process model is helpful in determining the process dynamics due to a disturbance or a set point change. The rule base is useful to transform the general tuning guidelines into a specific inference engine. The conclusion of the inference engine fixes the PI parameters properly so that good performance is achieved.

On-line Tuning of the PI Parameters

The tuning technique adapts on-line the PI control parameters in order to steer the closed-loop response to satisfy preset time-domain desired specifications. Examples of time-domain specifications for set point tracking and disturbance rejection applications are shown in Fig. 1. The shape of the performance specifications can be designed to

reduce overshoot, reject disturbances, and maintain proper speed of the response. The user should provide his nominal performance specifications in the form of vectors of upper and lower bounds, y^u and y^l respectively. The performance envelope should have a specific window size. Within this window size, the envelope consists of several zones. The first zone is the startup, which spans n_1 time units, the second zone is the settling zone, which spans n_2 time units, and finally the steady zone, which spans n_3 time units. The size of the bound in each zone should be designed by the user according to his understanding of the process dynamics. Also, the user should define a value for the closed-loop prediction horizon, P_w. More elaboration on the utilization of P_w will be discussed later. The tuning algorithm consists of two phases namely: observation and triggering phases.
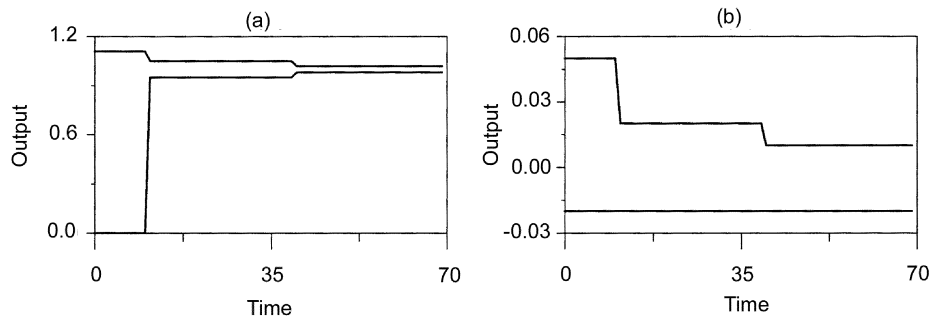

Fig. 1. Example of the performance specification envelope, (a) set point change; (b) step disturbance.

Observation phase

In the observation phase, the algorithm monitors the closed-loop prediction of the output. If the prediction violates the preset performance specifications or if the output set point is changed, then the algorithm switches into the triggered phase. In the same time, the algorithm adjusts the nominal performance specifications automatically to match the actual process behavior. The method of performance specifications adjustments is discussed elsewhere [1-4]. The closed-loop predictions are obtained by numerical integration of a process model over P_w. In this paper, state space models are used. However, the tuning algorithm is not limited to this type of models. In fact, any form of model can be used to predict the future outputs. In this paper, the process is modeled as follows:

$$\dot{z} = f(z,u,t) \tag{1}$$
$$y = Cz \tag{2}$$

and the following decentralized PI control law is considered:

Emad Ali

$$\dot{z}_e = y_r - y_p \equiv e \tag{3}$$

$$u = K_c e + K_c K_I z_e \tag{4}$$

where K_c and K_I are diagonal matrices with their diagonal elements being the controller gain (k_{ci}) and the reciprocal integral time (τ_{Ii}) for each control loop, respectively. Solution of Eqs. (1-2) along with (3-4) gives the closed-loop simulation of the process under a standard PI control structure. Note that in Eq. (3), the process measurement, y_p is required. Because future values for y_p are not available, the model outputs obtained from Eq. (2) are used instead. To dampen the effect of model-plant mismatch, a correction factor is added to the model outputs. The correction factor is defined as follows:

$$d(k_) = y_p(k) - y(k) \tag{5}$$

As mentioned earlier, during prediction, future values for y_p are not available. Therefore a constant value for d at the current sampling time k is used to correct the model outputs. The above control law (Eqs. 3 & 4) are presented in a continuous-time format. However, in this paper the control law is implemented in a discrete-time fashion to simulate real practice.

Triggering phase

In the triggering phase, the magnitude of performance violation and its rate are calculated from which a new value for the PI tuning parameters is determined. The new values of the PI parameters will be determined by a fuzzy logic system. The fuzzy logic system consists of three consecutive stages. Fuzzification is the first stage followed by the Base rules (inference engine) stage and finally the Defuzzification stage. Each stage is explained in the following section. Here, it is assumed that the reader is familiar with fuzzy logic terminology and concepts.

Fuzzification:

In this stage, a specific measured variable is transformed into a member of a set of fuzzy membership functions. Three different input sets of membership functions and one output set of membership functions are used in this paper. The first input set is shown in Fig. 2. The set consists of two membership functions namely: (G)ood denoted as μ_G and (H)igh denoted as μ_H. The universe of discourse of this set spans the possible values for its specific input, which is the scaled value for the bound violation. The scaled bound violation is defined as follows:

If upper bound is violated:

$$A \equiv \frac{y_j(k+m) - y_j^u(k+m)}{y_j^u(k+m)} \tag{6}$$

If lower bound is violated:

$$B \equiv \frac{y_j^l(k+m) - y_j(k+m)}{y_j^l(k+m)} \tag{7}$$

where y_j is the predicted value of the jth output. y^u and y^l are the upper and lower bound for y as described earlier. Also, k is the sampling time and m is the instant at which maximum violation occurs for the jth output. The jth output is determined such that it is the one with the maximum violation over all other outputs. The definition of A and B in the above equations guarantees positive value when the corresponding bound is violated and negative otherwise. Note that the upper and lower bounds can not be violated in the same time by one specific output. Therefore, if the upper bound is violated, then A belongs to the H membership function and B to the G membership function and *vice versa*. If neither bound is violated, then A and B belongs to the G membership function. This argument applies for the case of set point change. It also applies to disturbance case with the exception that the belonging of B to the membership functions is reversed. The reason for reversing is that the lower bound in the disturbance case is always negative because deviation variables are used in this paper.
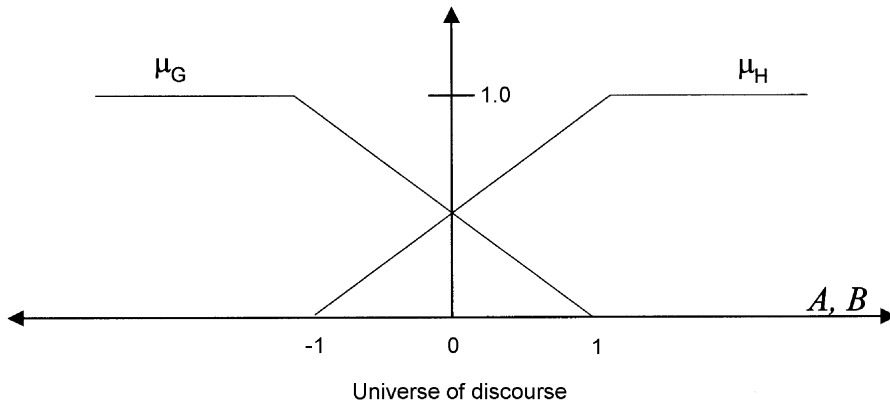

Fig. 2. Input fuzzy set for bound violation A & B.

The second input set of membership functions is shown in Fig. 3. The set consists of three functions namely: (P)ositive, (Z)ero and (N)egative. These functions are labeled μ_P, μ_Z and μ_N, respectively. The input to this fuzzy set is the scaled violation rate, which is defined as follows:

$$C \equiv \frac{y_j(k+m) - y_j(k+m-1)}{y_j(k+m)} \tag{8}$$

Emad Ali

The above expression for violation rate applies for both upper and lower bound violation. Note that scaled values are used for the bound violation measure (A, B) and its rate (C). The reason for that is to simplify the determination of the possible range for the universe of discourse in Figs. 2 and 3. The third input fuzzy set is shown in Fig. 4. The input to this fuzzy set is the simulation (or real) time, which is denoted as T. The corresponding membership function becomes active when the time exceeds half of the startup time. This fuzzy set is labeled μ_T and will be used to activate certain fuzzy rules as will be discussed later.
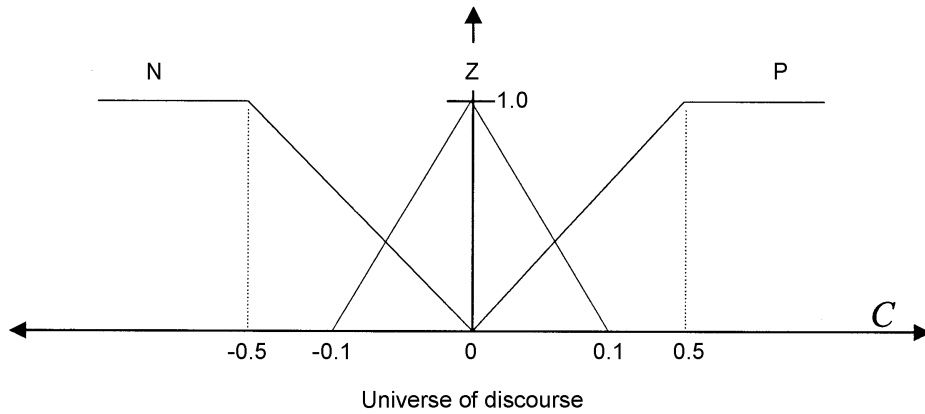

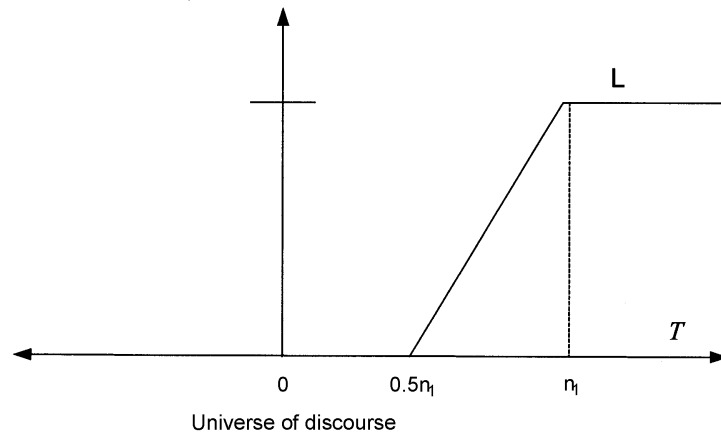**Fig. 3. Input fuzzy set for bound violation rate C.**

Fig. 4. Input fuzzy set for time T.

It should be noted that the predicted value of the output y is obtained from solving Eqs. (1) and (2). The calculated value is then corrected by adding the disturbance estimates, d (Eq. 5), to it before being used in Eqs. (6-8). This also provides some feed back from the process.

Inference Engine:

The rule base governing the tuning guidelines is given in Table 1. In the table, μ, represents the rule output. These rules apply for both k_c and k_I and for set point change and disturbance. Exceptions are that R1, R2, R4 and R7 are reversed in the disturbance case. The result of each rule given in Table 1 activates a specific output fuzzy set. The output membership functions (i.e. output fuzzy sets) are denoted as LN, SN, ZE, SP and LP as shown in Fig. 5. To simplify the numerical treatment, these functions are labeled μ_5, μ_4, μ_3, μ_2 and μ_1 respectively. The rules given in Table 1 formulate the general understanding of the effect of k_c and k_I on the closed-loop response. The general effect of these parameters is explained next.

Table 1. The base rules

No.	Rule	Result	Definition
R1	If A is H and B is G	Then □ is SN	Upper bound is violated, reduce k_c & k_i
R2	If A is G and B is H	Then □ is SP	Lower bound is violated, increase k_c & k_i
R3	If A is G and B is G	Then □ is ZE	Neither bound is violated, no change
R4	If A is H and C is P	Then □ is SN	Upper bound is violated and its rate is increasing, reduce k_c & k_i
R5	If A is H and C is Z	Then □ is LP	Upper bound is violated and its rate is stationary, increase k_c & k_i
R6	If A is H and C is N	Then □ is ZE	Upper bound is violated and its rate is decreasing, wait and see
R7	If B is H and C is P	Then □ is SP	Lower bound is violated and its rate is increasing, increase k_c & k_i
R8	If B is H and C is Z	Then □ is LP	Lower bound is violated and its rate is stationary, increase k_c & k_i
R9	If B is H and C is N	Then □ is ZE	Lower bound is violated and its rate is decreasing, wait and see
R10	If A is H and T is L	Then □ is LN	Upper bound is violated and time exceeded the startup time, reduce k_c & k_i
R11	If B is H and T is L	Then □ is LN	Lower bound is violated and time exceeded the startup time, reduce k_c & k_i

To illustrate the general effect of k_c and k_I on the closed-loop response, several simulation tests were carried out as shown in Fig. 5. The simulation shown in Fig. 6 is based on the CSTR example discussed later. The influence of varying k_c at fixed k_I and the influence of varying k_I at fixed k_c for set point change is shown in Fig. 6a and 6b, respectively. It is clear that k_c and k_I have similar effect on the response. In this case, increasing one of these parameters leads to a faster response. However, this will be at the expense of higher overshot and oscillation. Therefore, if fast response is sought, then the

PI parameters should be increased. On the other hand, if less overshoot is sought, then the PI parameters should be decreased. Moreover, to eliminate oscillation and/or provide stability at low frequency (i.e., asymptotically), the PI parameters should be decreased.
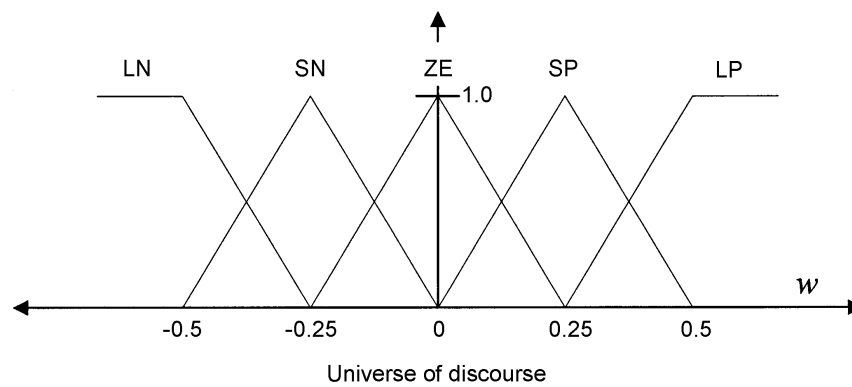

Fig. 5. Output fuzzy set for *w*.

The influence of varying k_c at fixed k_I and the influence of varying k_I at fixed k_c for rejecting a load disturbance is shown in Fig. 5c and 5d, respectively. It is observed that increasing one of the PI parameters creates faster response with less overshoot. However, excessive increase of the PI parameters leads to oscillatory response. Therefore, if a faster response and/or a less overshoot is necessary, then the PI parameters should be increased. On the other hand, to stabilize the response ultimately (at low frequency), the PI parameters should be decreased.

The knowledge gained from the above analysis is the basis for the tuning rules listed in Table 1. Note that the rules in Table 1 are developed for set point change and they are self-explained. In case of disturbance, R1, R2, R4 and R7 are reversed to accommodate the knowledge gained from the simulation shown in Fig. 5c and 5d. It should be noted that rules 10 and 11 are designed to provide stability at low frequency (i.e. after the dominant time constant of the process has elapsed). These two rules are based on the idea that any lower or upper bound violation that occurs after the startup time has elapsed is due to oscillation caused by large values for the PI parameters. Therefore, an intuitive reaction is to decrease the PI parameters. On the other hand, rules 5 and 8 are designed to eliminate offset. These rules are based on the idea that when the output violates the performance specifications and in the same time the output rate of change is very small, then the response suffers from an offset. This happens because the PI parameters became very small. In this case, increasing the PI parameters is an insightful reaction. The reasoning behind the rest of the rules given in Table 1 is straightforward. It should be noted that the possible tuning rules are not limited to those listed in Table 1. Any well-known knowledge or field experience can also be incorporated.
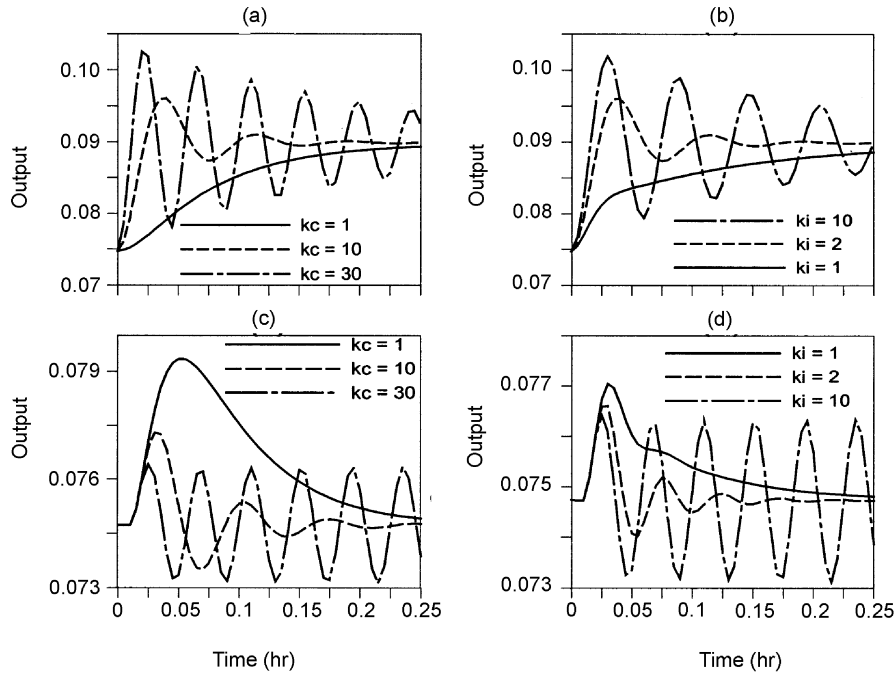
Fig. 6. Closed-loop response to different values for k_c and k_I.

Defuzzification:

In this stage, the results of the second stage (inference engine) are combined in a special way to produce a crisp value for the output. The obtained crisp output is the factor that will be used to adjust the PI tuning parameters. This procedure of combining the results of the inference engine is known as defuzzification. The defuzzification procedure is similar to finding the weighted average. Here, we adopt the center of area (COA) defuzzification method[18].   Before discussing the COA method, some terminology will be explained. The base rules of Table 1 are in script (linguistic) form. They can be cast in mathematical form so that they can be directly used to calculate the output crisp value. For example, the results of rule 1 in Table 1 can be written as follows:

$$\mu_{4,1}(w) = \min\,(\mu_H(A), \mu_G(B)) \tag{9}$$

where $\mu_H(A)$ is the membership grade of A to μ_H function or the degree of membership of A in the fuzzy set H. The same definition applies to $\mu_G(B)$ with respect to input B and fuzzy set G. On the other hand, $\mu_{j,i}(\bullet)$ denotes the membership degree of $(\bullet)$ to the jth

output membership function with respect to rule *i*. For example, $\mu_{4,1}$ denotes the membership degree to the 4th output membership function (i.e., SN) with respect to rule 1. Note that in the above equations, the AND command is transformed into minimum operation. In this case, a common fuzzy rule operation [18] is followed. Using the above criteria, the entire rules listed in Table 1 can be written mathematically as in Eq. 9.

Having transformed the rules into mathematical expressions, the COA can be applied to find the output (factor) for k_c and k_I as follows:

$$w(z) = \frac{\sum_{j=1}^{n_R} \sum_{i=1}^{n_f} \mu_{j,i}(z) \delta_i A_{j,i}}{\sum_{j=1}^{n_R} \sum_{i=1}^{n_f} \mu_{j,i}(z) A_{j,i}} \tag{10}$$

where n_R is the number of rules and equals 11 in this paper, n_f is the number of output membership functions and equals 5 in this paper, and δ_i is value for the location of the center of μ_i. The value of δ_i is pre-calculated and fixed as shown in Fig. 5. *A* is n_R x n_f pre-calculated matrix, which identifies which membership function is included in each rule. For example, row 1 of matrix *A,* which is assigned for rule 1, contains 1 at the first column and zeros elsewhere. The same logic is carried out over the remaining rows. The argument *z* denotes either k_c or k_i. However, since k_c and k_I have exactly the same directional effect on the closed-loop response, only a single factor is computed for both parameters for each control loop.

Triggering the tuning algorithm

As mentioned earlier the tuning method operates in two modes: observation and triggering. The triggering phase is entered when the tuning algorithm is triggered. The tuning method is activated either due to a set point change or to a load disturbance. In case of set point change, the tuning method is triggered automatically at the point where set point is modified. In due course, all outputs with set point change will be assigned a performance envelope for set point tacking, while the output with zero set point change will be assigned a performance envelope for disturbance rejection as shown in Fig. 1. When the time exceeds the window size of the performance envelope, the tuning method is disabled and returned to the observation mode. In case of load disturbance, the tuning method can not be triggered automatically because the point at which a disturbance is injected in the process is usually unknown. Hence, the algorithm will be operating in the observation phase during which the predicted output is checked against pre-defined threshold value. If the threshold is violated, then the process is assumed to be under the influence of disturbances. Therefore, the tuning algorithm is triggered and all outputs are assigned performance envelope for disturbance rejection. Whenever the time exceeds the window size of the performance envelope, the algorithm is disabled and returned to the

observation mode. The threshold value should be set by the user. It should be designed such that any process variation within the threshold is considered tolerable, e.g. due to measurement noise.

PI Parameters Adaptation Algorithm:

The adaptation algorithm can be clearly understood by the following algorithm: At any on-line sampling point k, and before computing the control action:

Step 1: Predict the closed-loop response over the prediction horizon (P_w) using Eqs. (1-4) for fixed values of the tuning parameters and constant disturbance variables.

Step 2: Check whether the closed-loop prediction violates the specifications. If it does not, then go to step 5.

Step 3: Determine the output index and the sampling point at which the maximum violation of the specification occurs. Let this be for output j and point $k+m$.

Step 4: Calculate the bound violation measure (A, B) and its rate (C) using Eqs. 6-8.

   Step 4.1: Determine the degree of membership of A and B with respect to membership functions μ_G and μ_H. Also, determine the degree of membership of C & B and C & A with respect to membership functions μ_P and μ_Z, μ_N. Determine the degree of membership of T (time) with respect to μ_T.

   Step 4.2: Calculate the adjustment factor w using Eq. (10).

   Step 4.3: Set $k_{c,j} = k_{c,j}\,(1+w)$, $k_{I,j} = k_{I,j}\,(1+w/10)$.

Step 5: Compute and implement the control action. Advance to the next sampling time in real-time operation and set $k=k+1$. Go to step 1.

In step 4.3 above, both PI parameters are adjusted by the same correction factor, w. The two PI parameters have similar influence on the closed-loop response. For example, increasing both k_c and k_I simultaneously and equally may lead to aggressive control actions. Similarly, decreasing both k_c and k_I simultaneously and equally may lead to a sluggish control action. On the other hand, changing k_c and k_I in different directions may not necessarily provide improved performance because the effect of each parameter will counteract the other. For this reason, both parameters are changed in the same direct, but at different rate to avoid over exploitation. Specifically, the rate of adjusting k_I was made intentionally ten times slower than that of k_c. This formulation does not necessarily result in optimal values for k_c and k_I. However, this is not the main objective of the proposed method. It is only meant to establish *automatic online* adjustment of the parameters to provide acceptable response.

The algorithm has one parameter, namely P_w. The prediction horizon is an important design parameter as it provides advance prediction of the behavior of the closed-loop response, which may result in earlier correction of the PI parameters. The larger the value of P_w, the more robust is the tuning algorithm, but the more the computational load is. On the other hand, a small value for P_w delays triggering and/or delays detection of bound violation. This will result in delayed correction of the controller parameters. A straightforward guideline for selecting the value of P_w is given elsewhere [1-4]. Since

the tuning algorithm works automatically, the manual adjustment of the PI tuning parameters is replaced by the manual adjustment of a single parameter, which is P_w. Here, P_w is fixed so that the tuning procedure becomes fully automatic.

It should be emphasized that the proposed tuning system retains the standard structure of the simple PI algorithms. The pure function of the tuning method is to modify the PI settings each sampling time for the sake of a better performance. In addition, to avoid complexity the method is kept as simple as possible where the user is only required to perform the following tasks:

- Design a performance envelope for his control objective.
- Come up with some sort of model needed for prediction.

If these pieces of information are provided by the user, then the tuning method will be triggered and start adapting automatically. Moreover, the inference engine has to be designed once based on available tuning guidelines and/or simulations. If abnormal dynamics that is not covered by the inference engine is observed, then the fuzzy base rule can be simply modified to include the additional knowledge.

Isothermal CSTR Example

This example is adopted from Wu [19]. It demonstrates a series/parallel reaction taking place in a CSTR. The non-linear model equation is given by:

$$\dot{C}_A = -k_1 C_A - k_3 C_A^2 + (F/V)(C_{Af} - C_A) \tag{11a}$$

$$\dot{C}_B = k_1 C_A - k_2 C_B - (F/V)C_B \tag{11b}$$

where F is the inlet flow rate of component A, V is the reactor volume, C_A and C_B are concentration of the reactant and an intermediate, respectively. The reaction rate constants are assumed to be $k_1 = 50$ 1/hr, $k_2 = 100$ 1/hr and $k_3 = 101$ 1/kmole hr. The plant operating conditions are $C_{Af} = 10$ kmol/m^3, $F/V = 1$, $C_{Ass} = 0.151$ kmol/m^3 and $C_{Bss} = 0.075$ kmol/m^3. The control objective is to regulate the intermediate product, C_B by manipulating the dilution rate, F/V. A linear model in the form of $\dot{z} = az + bu$ is obtained by step testing the plant. Specifically, 0.1 step change is applied in the dilution rate from which the model coefficients are determined to be $a = 40$ and $b = 2.32$.

To implement the proposed tuning method, the performance specification is designed first. The nominal performance specification for set point change is designed such that it limits the overshoot to 10% in the first 10 samples, to bring the response within ±5% of the final steady state for the following 15 samples and eventually to within ±2% for the last 16 samples. The nominal performance specification for disturbance is designed such that it limits the overshoot to 3% in the first 10 samples, to bring the response within ±1% of the final steady state for the following 31 samples. The

threshold value is set to ±2% of the output steady state value. A sampling rate of 0.005 hr is used in all simulations. Note that in all the following simulation, $\tau_I = 1/k_I$ is shown in the figures and P_w is fixed at 10 for fair comparison. It should be noted also that the PI controller is implemented on the non-linear plant model, i.e. Eqs. (10) and (11). The linear model is used only in the tuning algorithm to predict the output over P_w. The discrepancy between the non-linear plant and linear model creates model-plant mismatch, which makes the control problem more challenging. In this example, the integral time is limited between 0.5 and 10 and the controller gain is limited between 1 and 50. The manipulated variable, i.e. the dilution rate is limited between 0 and 2.5.

Figure 7 demonstrates the process response to a series of set point changes of +0.0152, 0.0252, -0.0252 as deviation from initial steady state. Aggressive initial PI settings of k_c (0) =30 and k_I (0) = 1 were used. The initial values for the PI settings are chosen such that they create oscillatory feedback response and thus testing the ability of the tuning method to smooth out the transient response. The light solid lines in Fig. 7 represent the performance envelope. The dashed line in Fig. 7 shows the feedback performance for fixed PI settings at the given initial values. The solid line in the same figure shows the feedback response using the self-tuned PI algorithm. Since P_w is chosen equal to the size of the startup zone, the tuning algorithm started the adjustment at the beginning of the simulation. As a result, k_c is decreased and τ_I is increased. This correction helped in providing closed-loop response with a good decay ratio for the first set point change. During the second set point change, the PI settings were kept changing resulting in a perfect response. At the beginning of the third set point change, k_c has already reached a small value and τ_I a large value, a situation that could have caused a sluggish response if k_c and τ_I were not changed. However, because of the prediction feature of the tuning algorithm, the PI settings were adapted properly to deliver a response with acceptable speed. Overall, the adapted response is superior to that with fixed PI settings.

To test the opposite situation, the above set point change test is repeated with smaller initial values for the PI settings. Specifically, Fig. 8 shows the result for this case with k_c (0) =10 and k_I (0) = 0.1. The idea is to start with sluggish control performance. As expected sluggish response especially for the third set point change is observed when the PI controller is applied with fixed settings. However, the PI control performance improved substantially with the aid of the proposed tuning algorithm. In this case, k_c was continuously increased accompanied with continuous decrease in τ_I. A situation led to improved control performance. Note that the feedback response for the first set point change was not enhanced as much as for the second and third set point changes. This is because the PI settings were modified gradually, which delayed their effect.

Figure 9 shows the result of testing the tuning algorithm for disturbance rejection using small initial values for the PI settings of k_c (0) =10 and k_I (0) =0.1. The disturbance is taken as 20% increase in the reactant feed concentration, C_{Af}. The disturbance takes place after two sampling instants from the beginning of the simulation.

As   Fig.   9

Fig. 7: CSTR feedback response to Set point change using k_c=30 and τ_I=1.

demonstrates, the proposed tuning method managed to increase k_c and equivalently
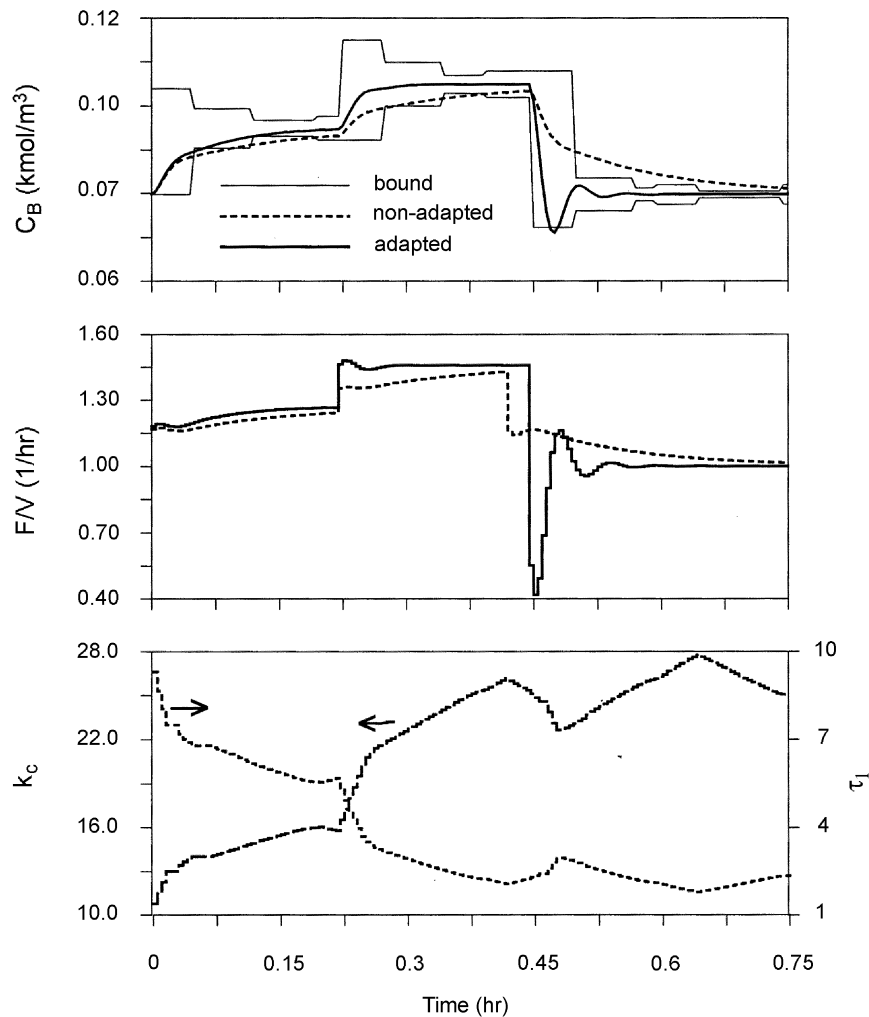
decrease τ_I gradually. This modification helped in delivering somewhat faster recovery to the set point. The disturbance test is repeated using large initial values for the PI settings such as k_c (0) =30 and k_I (0) =1. The simulation result is depicted in Fig. 10. There is no doubt that the closed-loop response obtained at fixed PI settings is unstable. However, with the aid of the controller tuning method, the feedback response was stabilized. It is true that the initial oscillation lasted for some long period before it settles down. This is attributed to the infinitesimal change in the PI settings. The tuning

algorithm can be modified to allow larger adjustments in the PI settings. However, this is avoided because it creates sudden kick in the control action and oscillation may creep in.

In the last simulation and even in the earlier ones, it is observed that the PI setting were continuously updated even that the closed-loop response was completely inside the performance envelope. This means that the model was detecting bound violation during that period. There are two explanations to this behavior. One is that the model makes wrong prediction due to modeling errors, another is that the true plant dynamics will violate the bound in the future if the PI settings were kept fixed at their last values. However, these violations disappear as time goes on because the PI settings change continuously.   It is found that the second explanation is more common.
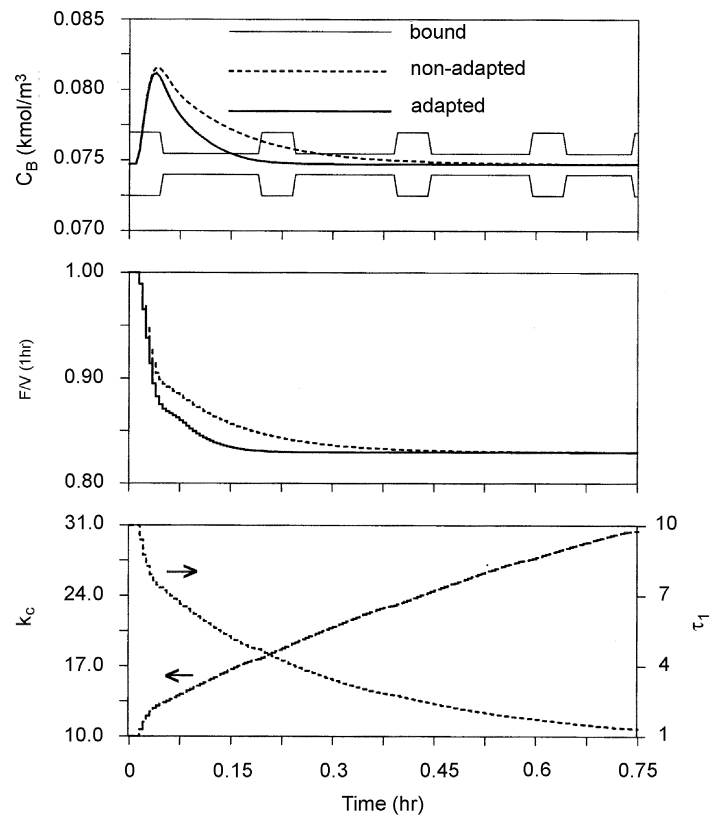

Fig. 9. CSTR feedback response to disturbance step change using k_c=10 and τ_I=10.
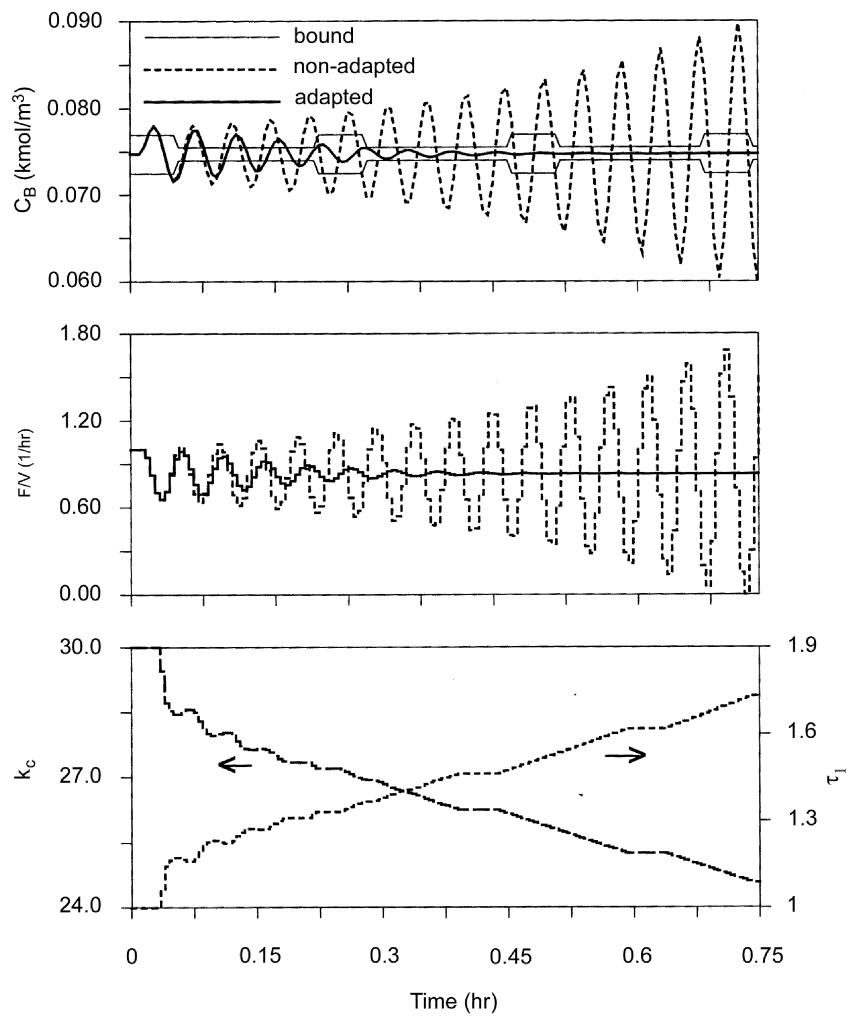
Fig. 10. CSTR feedback response to disturbance step change using k_c=30 and τ_I=1.

Forced Evaporator Example

A forced circulation evaporator is shown in Fig. 11. The process is originally proposed by Newell and Lee [20] and is modeled as follows:

$$M \frac{dC_2}{dt} = F_1 C_1 - F_2 C_2 \tag{12}$$

$$W \frac{dP_2}{dt} = F_4 - F_5 \tag{13}$$

where C_1 and C_2 are the input and product compositions, respectively and P_2 is the operating pressure (kPa). F_1 and F_2 are the feed and product flow rates (kg/min), respectively. F_4 and F_5 are the vapor and condensate flow rates, respectively (kg/min). M is the liquid holdup in the evaporator (20 kg) and W is a constant (4 kg/kPa). The liquid level in the separator is considered well controlled by manipulating the product flow rate. Therefore, the flow rates are given as follows:

$$F_2 = F_1 - F_4 \tag{14}$$

$$F_4 = [0.16(F_1 + F_3)(-0.3126C_2 - 0.5616P_2 + 0.1538P_{100} + 41.57) - F_1 C_p (0.3126C_2 + 0.5616P_2 + 48.43 - T_1)] / \lambda_{s1} \tag{15}$$

$$F_5 = \frac{2UA(0.507P_2 + 55 - T_{200})C_p F_{200}}{\lambda_{s2}(UA + 2C_p F_{200})} \tag{16}$$

where P_{100} is the steam pressure (kPa). F_{200} (kg/min) and T_{200} (^{o}C) are the flow and temperature of the cooling water, respectively. UA is the product of heat transfer coefficient and the transfer area (6.84 kW/K) and C_p is the heat capacity of the cooling water (0.07 kW/kg min). F_3 is the recycle flow rate (20 kg/min). λ_{s1} is the latent of steam at saturation condition (36.6 kW/Kg min) and λ_{s2} is the latent heat of evaporation of water (38.5 kW/kg min). F_1 is fixed at 10 kg/min and its temperature T_1 is fixed at 40 ^{o}C. The cooling water enters the cooler at 25 ^{o}C. The initial steady state value for the outputs is $C_2 = 13.5747\%$ and $P_2 = 28.0865$ kPa which corresponds to $C_1 = 5\%$, $P_{100} = 200$ kPa and $F_{200} = 200$ kg/min. The control objective is to maintain the outputs within desired values using P_{100} and F_{200} as manipulated variables. This control problem is selected because of its strong cross-loop interaction. The diagonal element of the relative gain array is around 0.5. In this paper, P_{100} is used to regulate P_2 and F_{200} to regulate C_2. Both manipulated variables are constrained between 0 and 400.

The initial values for the PI settings for each control loop is determined by reaction curve (RC) method [20]. In this case, the PI settings for the first loop, i.e. $F_{200} \rightarrow C_2$, are found to be $k_c = 88.8$ and $\tau_I = 31$ min, and those for the second loop, i.e. $P_{100} \rightarrow P_2$, to be $k_c = 8$ and $\tau_I = 24$ min. Alternatively, the proposed tuning method can be used to find better values for the PI control parameters. In order to utilize the proposed tuning method the performance specifications should be designed as explained in the next paragraph.
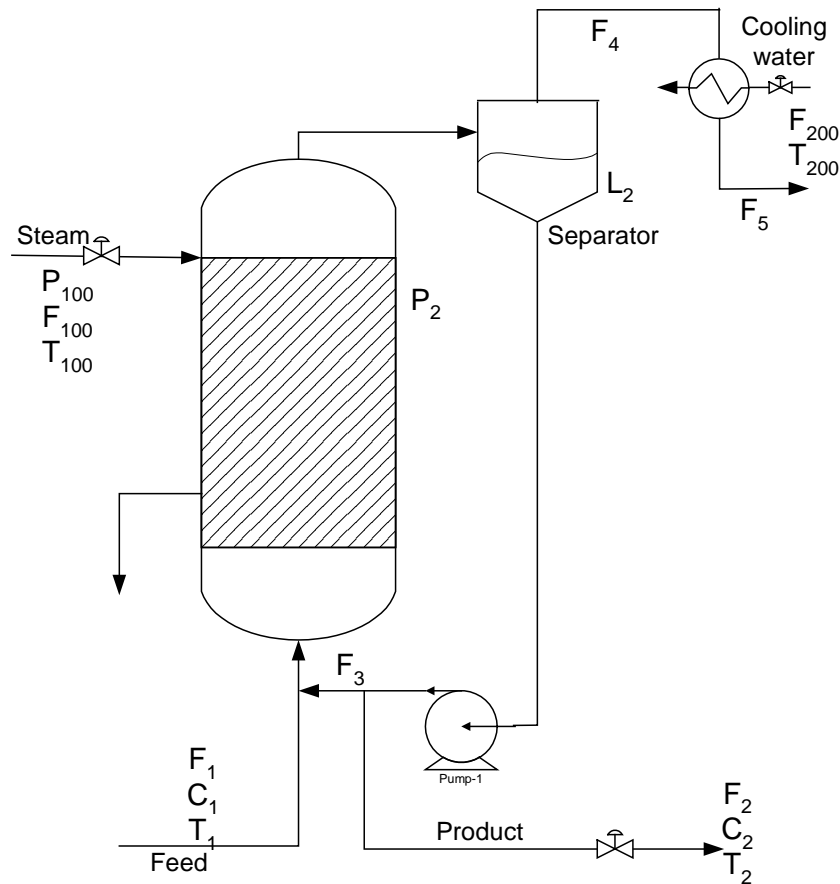
Fig. 11. Evaporator process.

The nominal performance envelope for set point change is designed such that it limits the overshoot to 8% for the first 30 samples, brings the response to within ±5% of the final steady state for the following 40 samples and finally to within ±1.1% for the last 40 samples for the first output. For the second output the envelope limits the overshoot to 5% for the first 30 samples, brings the response to within ±2% of the final steady state for the following 40 samples and finally to within ±1.1% for the last 40 samples. The nominal performance envelope for disturbance rejection is designed such that it limits the overshoot to 5% for the first 30 samples, brings the response to within ±2.0% of the final steady state for the following 40 samples and finally to within ±0.5% for the last 40 samples for both outputs. A sampling time of 1 min is used in all simulation. In all simulations that follow, the profile of the manipulated variables will be

excluded to save space. It should be noted also that the modeling equations (Eqs. 12-16) are used twice during simulation. They will be used to simulate the plant upon which the control law is implemented. In addition, they will be used by the tuning algorithm to predict the plant future outputs. In the second case, imperfect model will be considered. Specifically, the overall heat transfer coefficient, U for the model is considered 20% larger than that for the plant. In addition, the constant M for the model is considered 10% less than that for the plant. This procedure is applied to all the simulations that follow and is expected to create model-plant mismatch, which makes the control task more critical. Note that in all the following simulation, $\tau_I = 1/k_I$ is shown in the figure and P_w is fixed at 10 for fair comparison.

Figure 12 demonstrates the feedback response to set point change of [1%, 0 kPa] as deviation from the initial steady state. The dashed line in Fig. 12 shows the feedback response using fixed PI settings found via reaction curve method. The resulting response is acceptable except of sluggishness observed in the pressure response. The light solid lines in Fig. 12 represent the performance specification. The tuning method is triggered at the beginning due to set point change and then re-triggered at the 120th sampling instant due to threshold violation. Fig. 12 illustrates the process response obtained from using the tuning method with the RC settings being used as the initial guess for the PI parameters. It is clear that the adapted concentration response is slightly affected because it is originally within its performance specification. The adapted pressure response delivered faster recovery to the set point. The k_{c2} profile shows initial step increase and similarly the τ_{I2} profile demonstrates initial step decrease. These variations are the source for the improved performance of the second output (pressure). The profile of the PI settings for the first loop shows no adaptation until the second triggering period. The performance envelope in the second triggering zone is tighter and moreover the concentration response suffers from slow drift. This in turn made the model to detect continuous violation of the lower bound. This explains the motive for the continuous adaptation of the PI settings for the first loop in the last 75 sampling instants.

Figure 13 demonstrates the feedback response to three consecutive set point changes of [+2%, +2 kPa], [+1%, +1 kPa] and [+2%, +2 kPa] as deviation from the initial steady state. Evidently, the tuning method managed to speed up the pressure response. The concentration response seems to be acceptable and requires no further improvement through adaptation. Despite this fact, the PI settings for the first loop were continuously adapted especially for the second half of the simulation. Nevertheless, the adapted concentration response reacts marginally to these variations because C_2 has slow dynamics. However, due to cross-loop interaction the changes in the first loop parameters are reflected on the second output performance as manifested by the minor oscillation occurred in the third set point change zone.

Figure 14 depicts the closed-loop response to step disturbances while the process is operating at the initial steady state. Specifically, T_{200} for the plant is taken 5 degrees less than that for the model. Similarly, T_1 for the plant is taken 5 degrees less than that for
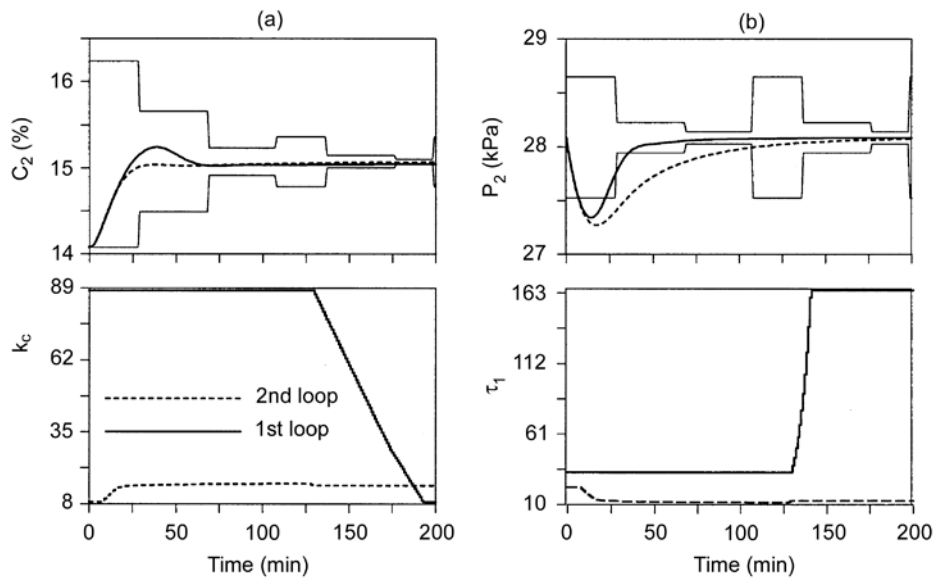
Fig. 12. Evaporator response to set point change, (a-b) light solid: bound; dashed: non-adapted PI; solid: adapted PI.
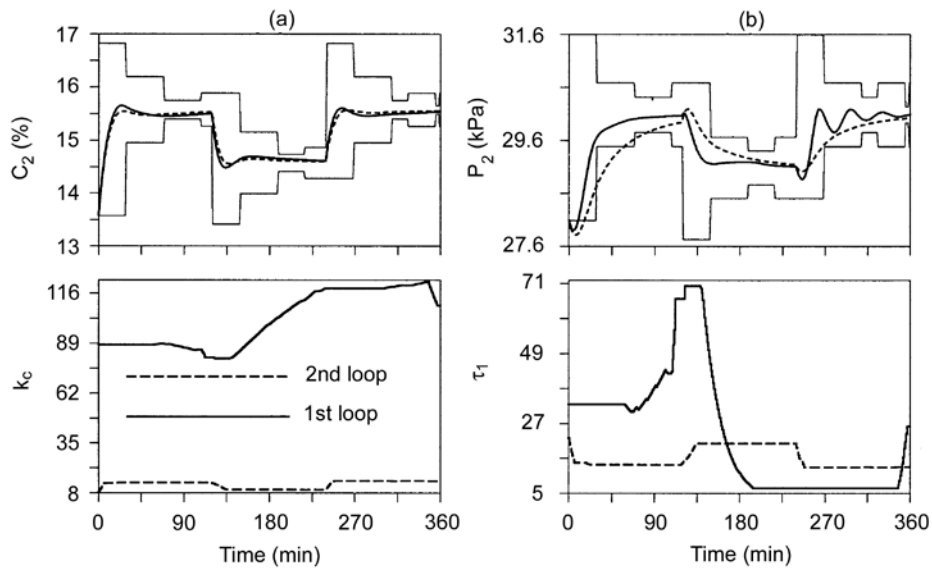

Fig. 13. Evaporator response to consecutive set point changes, (a-b) light solid: bound; dashed: non-adapted PI; solid: adapted PI.

the model. These step changes are introduced after 5 minutes from the start of the simulation. The adapted PI response is shown in Fig. 14 by the solid line. In this simulation, the RC settings are used as the initial guess. As one can see, the main effort of the tuning algorithm was dedicated to speed up the response of both outputs. As a result shorter settling time was observed. This was achieved at the cost of notable oscillation in the pressure response. Fortunately, the oscillation diminished eventually because k_{cI} and τ_{II} were modified in the correct direction.
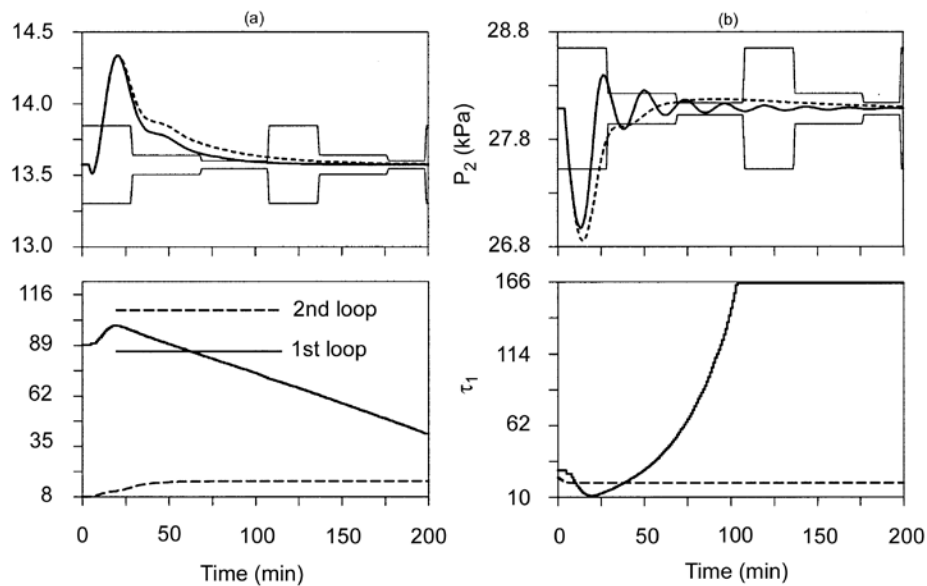

Fig. 14. Evaporator response to disturbance step change, (a-b) light solid: bound; dashed: non-adapted PI; solid: adapted PI.

Figure 15 illustrates the feedback reaction to set point change of [1%, 1 kPa] followed by a step disturbance introduced at time = 120 minutes from the start of the simulation. The disturbance employed here is a step change of -5°C in T_{200} and $+5$ $^{\circ}$C in T_1. Note that the input upsets are introduced into the plant but not into the model. This implementation procedure is considered here to simulate real practice and to create model-plant mismatch. This mismatch is additional to that produced by the parametric uncertainty mentioned earlier in the example. This situation is expected to make the control problem more difficult. It is clear that the adapted response outperforms the non-adapted one in the sense of faster dynamics. Initially, the parameters of the second loop were modified properly, which resulted in faster response for the pressure. Later on, i.e. in the disturbance rejection zone, the

parameters of the first loop were altered to reduce the overshoot and speed up the concentration response. Interestingly, this alteration also brought in its positive effect on the second output, which made the adaptation of the second-loop parameters unnecessary. It is noted that the parameters of the first loop started changing even before triggering for disturbance rejection. The reason behind this phenomenon is discussed earlier in the CSTR example.
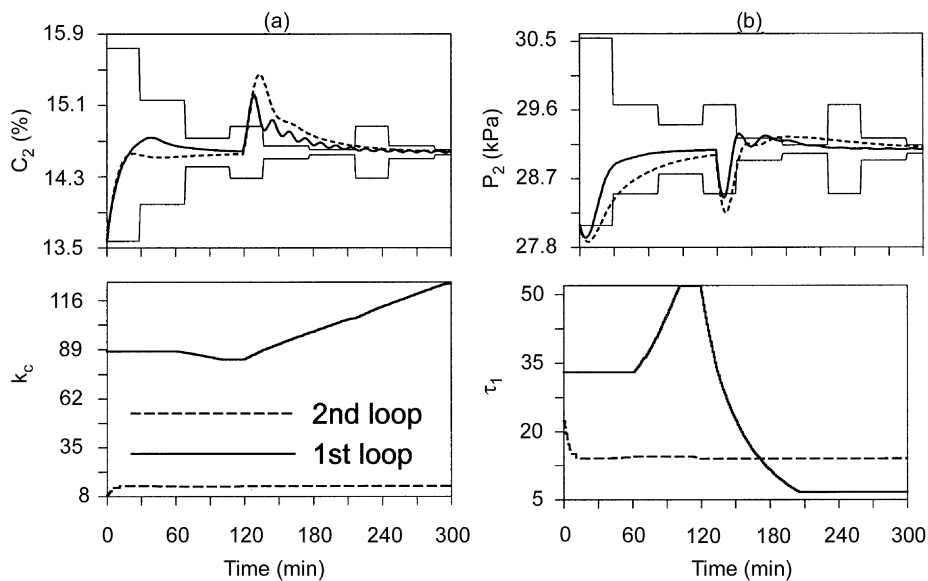

Fig. 15. Evaporator response to set point change followed by step disturbance, (a-b) light solid: bound; dashed: non-adapted PI; solid: adapted PI.

Conclusion

This paper presents an automatic tuning method for the online adjustment of the PI settings. The method is based on the general tuning guidelines and on real-time performance specification. The method requires a model representation of the plant. However, any form of modeling equations is sufficient to implement such a tuning strategy. One attractive feature of the tuning method is that it is computationally tractable. The PI parameters are tuned by means of simple algebraic manipulations where other tuning methods require iteration, root finding, or optimization. Another feature of this method is that it relies on intuitive guidelines instead of complex theoretical analysis, which a practitioner finds it difficult to understand. The proposed method is implemented on two process models in the

presence of modeling errors. The results of the simulations revealed the ability of the proposed tuning method to adjust the PI settings online such that much enhanced control performance can be obtained. The investigation also revealed that the proposed method is successful even if different initial guesses for the PI settings are used. The method has a single adjustable variable, which is the prediction horizon. The selection of this variable is straightforward.

References

[1]   Ali, Emad. "On-line Tuning Strategy for PI Control Algorithms." *Journal of King Saud University*, 11, *Eng. Sci*., No. 1 (1999), 49-70.

[2]   Ali, Emad. "Control of Non-linear Chemical Processes Using Adaptive PI Algorithms." *Ind. Eng. Chem. Res*., 39 (2000), 1980-1992.

[3]   Ali, Emad. "On-line Tuning Strategy for PI Control Algorithms Based on Simple Linear Models." *J. Chemical Engineering of Japan,* 35, No. 4 (2002), 324-333.

[4]   Ali, Emad. "Online Tuning Strategy for Multi-loop SISO PI Control Algorithms in Multivariable Interactive Systems." *Journal of King Saud University*, 14*, Eng. Sci.,* No. 2 (2002), 183-198.

[5]   Unar, M.A., Smith, D.J. and Shah, S.A. *Design and Tuning of Fixed Structure PID Controllers: A Survey*. Technical Report CSC-96016, Scotland: University of Glasgow, 1996.

[6]   Astrom, K.J., Hagglund, T., Hang, C.C. and Ho, W.K. "Automatic Tuning and Adaptation for PID Controllers: A Survey." *Control Eng. Practice*, 1, No. 4 (1999), 699-714.

[7]   Zadeh, L.A. "Fuzzy Sets." *Information and Control*, 8 (1965), 338-353.

[8]   Bandyopadhyay, R. and Patranabis, D. "A Fuzzy Logic Based PI Autotuner." *ISA Transaction*, 37 (1989), 227-235.

[9]   Xu, J., Lui, C. and Hang, C. "Tuning of Fuzzy PI Controllers Based on Gain Phase Margin Specification and ITAE Index." *ISA Transaction*, 35 (1996), 79-91.

[10]   Rajani, K.M. and Nikhil, R. P. "A Robust Self-tuning Scheme for PI- and PD-type Fuzzy Controllers." *IEEE Transaction on Fuzzy Systems*, 7 (1999), 2-16.

[11]   Miler, R., Kramer, W., Doblhoff, O. and Bayer, K. "Strategies for Optimal Dissolved Oxygen (CO) Control." *6th International Conference on Computer Application in Biotechnology*, Garmisch Partenkirchen, May 14-17, (1997), 315-318.

[12]   Ling, C. and Edgar, T. "Real-time Control of a Water-gas Shift Reactor by a Model-based Fuzzy Gain Scheduling Technique." *J. Process Control*, 7, No. 4 (1997), 239-253.

[13]   McMillan, G., Wojsznis, W. and Borders, G. "Flexible Gain Scheduler." *ISA Transaction*, 33 (1994), 35-41.

[14]   Chan, K. C. "Development of a Feedback Controller Tuner Using Virtual Fuzzy Sets." *Computers in Industry*, 28 (1996), 219-232.

[15]   Ramkumar, K. B. and Chidambaram, M. "Fuzzy Self-tuning PI Controller for Bioreactor." *Bioprocess Engineering*, 12 (1995), 263-267.

[16]   He, S., Tan, S., Xu, F. and Wang, P. "Fuzzy Self-tuning of PID Controllers." *Fuzzy Sets and Systems*, 56 (1993), 37-46.

[17]   Pfeiffer, B.M. and Isermann, R. "Self-tuning of Classical Controllers with Fuzzy Logic." *Math. Comp. in Simu*., 37 (1994), 101-110.

[18]   Yen, J. and Langari, R. *Fuzzy Control*, New Jersey: Prentice Hall, 1999.

[19]   Wu, W. "Adaptive Non-linear Control of Non-minimum-phase Processes", *Chem. Eng. Sci*., 54

(2000), 3815-3829.

[20]   Newell, R. B. and Lee, P.L. *Applied Process Control: A Case Study*. Sydney: Prentice-Hall, 1989.

تنغيمّ آلي متواصل لنظم التحكم النسبي التكاملي مبني على مواصفات زمنية

عماد الدين مصطفى

قسم الهندسة الكيميائية

كلية الهندسة ، جامعة الملك سعود

ص ب ٨٠٠، الرياض ١١٤٢١، المملكة العربية السعودية

ملخص البحث. يهدف هذا البحث الى تطوير خطة تنغيم متواصلة لمعاملات نظم التحكم النسبي التكاملي لتحقيق كفائة مرغوبة مبنية على مواصفات في النطاق الزمني. تستخدم الخطة المقترحة نموذج رياضي للعملية التصنيعية و ذلك لتقدير التغيرات الزمنية المستقبلية و تحديد كمية مخالفة المواصفات المرغوبة. تستخدم الكمية القصوى لمخالفة المواصفات مع التوجيهات العامة للتنغيم في تشكيل نظم و مجموعات المشوشة لإيجاد قيم معاملات نظم التحكم النسبي التكاملي. تطبق هذة الخطة بطريقة متواصله لتوفير تنغيم آلي مستمر لمعاملات نظام التحكم على نطاق واسع من ظروف التشغيل. كما تتميز الخطة بالبساطة و قلة الحسابات الرياضية حيث لا تتطلب الخطة نماذج خاصة للنظم الدينامية بالإضافة الى اعتماد الخطة على عدد بسيط من المجموعات و القوانين المشوشة. أوضحت نتائج التطبيق بالمحاكاة على نمادج رياضية لعمليات كيميائية نجاح الخطة في توفير كفائة في الأداء مقارنة بتلك التي تقدمها نظم التحكم النسبي التكاملي التقليدية.