Petroleum & Coal
ISSN 1337-7027

Available online at www.vurup.sk/petroleum-coal
Petroleum & Coal 55 (3) 148-155, 2013

PID CONTROLLER TUNING: IMPROVEMENT OF CLASSIC APPROACHES IN CHEMICAL PROCESSES

Behnam Baloochy*

*Department of Modelling and Process Control, Technology Development Research Division, Research Institute of Petroleum Industry, Tehran, Iran, *baloochyb@ripi.ir*

Received February 18, 2013, Accepted July 1, 2013

Abstract

In chemical processes, the classic techniques for tuning a PID controller have become even more popular with the advent of controllers capable of tuning themselves. In this paper, self-regulating controller using the relay oscillation method has been employed to specify the process parameters of ultimate gain and ultimate period. The auto tuning procedure starts by taking input/output measurements from the process. Ziegler-Nichols or modified Ziegler-Nichols tuning methods may be utilized to determine PID controller tuning parameters. The same technique has been improved to specify a first-order plus dead time model for self-regulating processes and an integrating process model for integrating processes. Model based tuning rules, such as Internal Model Control (IMC), may be exploited to establish tuning parameters for various feedback controllers.

Keywords: PID Tuning; Self-regulating Controller; Process Control; Chemical Process; Ziegler-Nichols tuning.

1. Introduction

Tuning a control loop is the adjustment of its control parameters (gain/proportional band, integral gain/reset, derivative gain/rate) to the optimum values for the desired control response. Stability is a basic requirement, but beyond that, different systems have different behavior, different applications have different requirements, and requirements may conflict with one another. Determining the tuning parameters of a PID controller based on ultimate gain and ultimate period is known as Ziegler-Nichols frequency response tuning [1-2]. The technique, developed more than 50 years ago, has been used extensively to tune loops in the process industries. The original Ziegler-Nichols tuning rules were designed to provide a quarter amplitude damped response to a load disturbance. Once considered ideal, the under damped and oscillatory nature of Ziegler-Nichols tuning has been criticized for destabilizing control loops, i.e. increasing variability instead of reducing it [3].

An improvement in performance is achieved by modifying the tuning rules in such a way to get a desired phase or amplitude margin in the loop [4]. Modified Ziegler-Nichols tuning rules are more conservative than the original rules, reducing the overshoot and oscillation following a setpoint change or load disturbance. Once the ultimate gain and ultimate period of a loop are known, published tuning rules can be used to get initial controller settings.

The Ziegler-Nichols frequency response tuning method was originally a closed-loop tuning technique that was performed manually. The gain of a proportional-only controller would be gradually increased until the loop oscillated at a sustained period. This ultimate period, along with the controller gain (ultimate gain), would be used to calculate the P, I, and D settings. However, there are some obvious drawbacks to this technique: getting the loop to cycle continuously is a time-consuming process and there is a risk that the oscillations will grow beyond stability. There is no way to specify the magnitude of oscillations.

The approach became significantly more attractive after introducing relay oscillation auto-tuning, as described in [4] and [5]. An example of the implementation of this approach in a Distributed Control System (DCS) is given in [7] and [8]. The relay oscillation tuning method

identifies the ultimate gain and ultimate period, so that controller settings may be determined from these parameters. Relay oscillation is an on-demand automatic tuning technique. Auto-tuning can be categorized as tuning on-demand or continuous adaptive tuning. On-demand tuning must be initiated by a human. Continuous adaptive tuning is performed automatically following setpoint changes, significant disturbances, or from low level injected excitation signals. Most loops need only tuning on demand upon commissioning or perhaps scheduling of tuning parameters to deal with process nonlinearities.

In recent years significant progress has been made with model based tuning, in particular with Internal Model Control (IMC) and Lambda tuning [9-10]. Both approaches result in a first-order closed loop response to setpoint changes. A tuning parameter relating to the speed of response is used to vary the tradeoff between performance and robustness. Both methods adjust the PID controller reset (or reset and rate) to cancel the process pole(s) and adjust the controller gain to achieve the desired closed loop response. IMC and Lambda tuning have become popular because oscillation and overshoot are avoided and control performance can be specified in an intuitive way through the closed loop time constant.

One of the limitations of model based tuning is process model identification. An equivalent first-order plus deadtime process model with parameters of static gain, apparent deadtime, and apparent time constant is usually identified for self-regulating processes. For integrating processes, model parameters of process gain and deadtime are determined. Model identification is typically made by an open loop step test. Compared to the relay oscillation method open loop methods are not easy to automate. With open loop methods, human intervention is often required to assure an accurate model due to nonlinearities in the process, valve hysteresis, and load disturbances. A different technique is required for self-regulating and integrating processes.

This paper describes the relay oscillation tuning method and an enhancement to it that identifies a self-regulating and an integrating process model in addition to ultimate gain and ultimate period. This enhancement may be achieved without additional manipulation of the process input. One automated procedure lets the user choose among many different tuning rules. For example, modified Ziegler-Nichols rules might be chosen if the process deadtime is small in comparison to time constant and fast setpoint tracking, and disturbance rejection is required. On the other hand, IMC or Lambda tuning rules might be used if the process deadtime is moderate compared to time constant or minimum variance control is desired. For control of integrating processes, like liquid level control, modified Ziegler-Nichols rules result in tight regulation, while Lambda tuning rules make averaging control possible. When there is mild to moderate interaction between loops, choosing the appropriate closed loop time constants with Lambda tuning can minimize the impact of the interactions. Interacting loops may also be tuned by relay oscillation [13-14] but a sequential or iterative procedure is required. The enhanced relay oscillation tuning method also offers the capability to auto-tune the two-degree-of-freedom PID controller [15], the deadtime compensating PID controller (Smith Predictor) and the SISO fuzzy logic controller described in [16-17].

2.Tuning Based on Relay Oscillation Method

During the tuning process the controller is replaced by the relay as shown in Fig. 1. With the loop at steady state, an initial step change is made to the process input. The amplitude of the step change is set to control the amplitude of oscillations in the process output. The ability to control the amplitude of process oscillation is a significant advantage over the closed loop technique to identify ultimate gain and ultimate period.

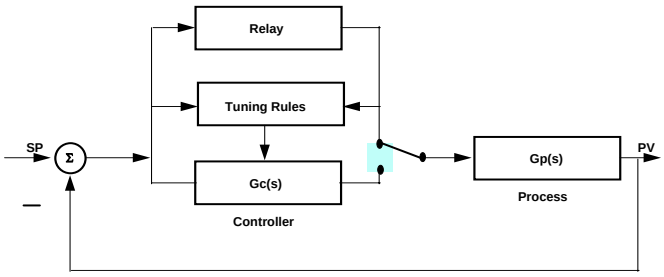

Figure 1. Relay Oscillation Principle

Fig. 2 shows a trend plot of the relay output and process output for a typical tuning sequence. After the initial step change in relay output, the process output begins to change after an interval of deadtime. After the first relay step, the loop is open until the process output moves by a predefined amount. At that point the relay is stepped in the opposite direction as shown. The process output begins to move in the opposite direction after the deadtime interval. The loop is now under two-state control. Each time the process output crosses the setpoint or initial value, the relay is switched; this continues for one or more periods. When active tuning is complete, control of the loop is returned to the controller with its original mode, setpoint, and output restored. Relay action causes the loop to oscillate at its ultimate period, T_u. Referring to Fig. 2 the ultimate gain, K_u , is the ratio of relay amplitude to amplitude of process oscillations or, more accurately, $K_u = \frac{4d}{\pi a}$

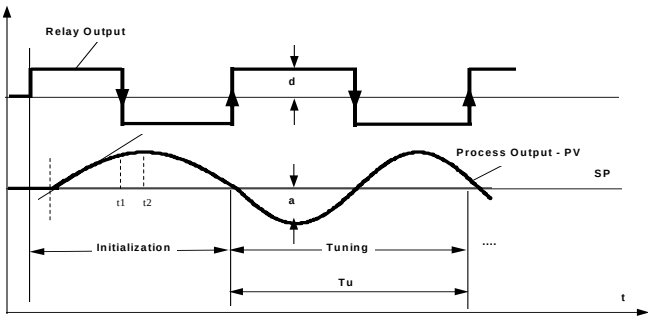

Figure 2. Trend plots of relay output and process output during active tuning

Relay hysteresis is used during initialization as described, but may also be used to prevent relay switching due to noise. Using hysteresis to delay relay switching after the process output crosses the setpoint is effective for noise protection, but it impacts the accuracy of the ultimate period and ultimate gain identification. The preferred method of noise protection is to disable relay switching for a short time following a change in relay output. Disabling relay switching for 50% of the process deadtime has proven to be an effective method of noise protection without adversely affecting the ultimate gain and ultimate period identification.

It is important to identify the process deadtime for noise protection and as one of the process model parameters. Apparent deadtime is identified by calculating a tangent from the point of maximum slope of the process output during initialization. The tangent is extrapolated to intercept the setpoint or initial value line of the process output. The time between the initial relay step and this intercept is the apparent deadtime.

Another method may be used to determine deadtime: the time between the first switching of the relay in the opposite direction, t1 in Fig. 2, and the time at which the process output reaches the maximum, t2, is the deadtime in a first order plus deadtime model. The apparent deadtime may be calculated as the average of the two results.

3. Specification of Model Parameters

One of the advantages of relay oscillation tuning is that it is applicable to both self-regulating and integrating processes. Having determined the apparent deadtime, an integrating process model has already been identified since the maximum slope of the process output during initialization is known. This slope is the gain for an integrating process. Regardless of whether the process is integrating or self-regulating, the auto-tuner will save the integrating gain value along with the apparent deadtime.

Knowing the deadtime, ultimate gain, and ultimate period it is possible to calculate a first-order plus deadtime process model as detailed in [11]. The following equations are derived from the first-order process model transformed into the frequency domain in the exponential form:

$$T_c = \frac{T_u}{2\pi}\tan(\pi - \frac{2\pi T_d}{T_u}) \quad (1)$$

$$K_s = \frac{1}{K_u}\sqrt{1+\frac{4\pi^2 T_c^2}{T_u^2}} \quad (2)$$

Where: T_c = process time constant; T_u = ultimate period; T_d = process apparent deadtime; K_s = process static gain; K_u = ultimate gain.

The process time constant in equation (1) is expressed by a tangent function which gives a good approximation when the deadtime is relatively long in relation to the time constant. For processes with insignificant deadtime a large error results in the time constant computation, even for a small error in deadtime identification. A heuristic approach to replace equation (1) has been developed [11] and implemented in a DCS [8]. It is based on knowing the ratio of ultimate period and deadtime, which is approximately equal to two for deadtime dominant processes and close to four for processes that are lag dominant. Using this approach allows the time constant and static gain to be identified within an accuracy of 10-20 percent without having to modify the relay oscillation tuning procedure.

The accuracy of model identification for a self-regulating process may be improved somewhat by identifying the process static gain through a closed loop step test. Following relay oscillation a small setpoint change may be made with the controller in automatic mode. The static gain is calculated as the ratio of percent change in setpoint and percent change in controller output between steady states. Knowing the static gain, ultimate gain, and ultimate period, the process deadtime and time constant may be calculated from the following equations [11-12]:

$$T_c = \frac{T_u}{2\pi}\sqrt{(K_s K_u)^2 - 1} \quad (3)$$

$$T_d = \frac{T_u}{2\pi}(\pi - \tan^{-1}\frac{2\pi T_c}{T_u}) \quad (4)$$

Table 1 demonstrates the accuracy of using equations (3) and (4). The ultimate gain and ultimate period were determined for several first-order plus deadtime process models using relay oscillation. The time constant and deadtime were calculated using equations (3) and (4) and the actual static gain. The difference between actual and calculated time constant and deadtime averaged less than five percent. If the static gain is identified with a five percent error in accuracy, the calculated time constant would be in error by as much as 10 percent.

Upon completion of active tuning, the auto-tuner has identified the ultimate gain, ultimate period, deadtime, time constant, static gain and integrating gain for the process. The user may input the type of process, self-regulating or integrating, the type of tuning rules to use, and perhaps an additional tuning factor such as the closed loop time constant. The tuning rules are then applied and the initial settings are written to the controller.

Table 1 Comparison of actual and calculated process model from static gain, ultimate gain and period

Ks actual	Tc actual	Td actual	Ku	Tu	Tc calc	Td calc
1.0	100	12	13.67	48.0	104.2	12.5
1.0	25	3	13.64	12.0	26.0	3.1
1.0	50	12	7.51	44.0	52.1	11.0
1.0	12.5	3	7.36	11.2	13.0	3.0
1.0	66.67	12	9.5	46.0	69.2	12.3
1.0	16.67	3	9.77	11.2	17.3	3.0
1.0	200	12	26.02	50.0	207.0	12.8
1.0	50	3	27.06	12.0	51.7	3.1
1.0	15	2	12.32	8.0	15.64	2.1
2.0	15	2	6.15	8.0	15.61	2.1
0.5	15	2	24.68	8.0	15.67	2.1

4. Extension and Modification of Ziegler- Nichols Tuning Method

Modified Ziegler-Nichols tuning rules are based on ultimate gain and ultimate period. An extension of these rules has been developed by Astrom and Hagglund [15] which utilizes an additional process parameter, the static gain, and results in a significant improvement over Ziegler-Nichols rules. These extended tuning rules, or simple tuning rules as they are called, also determine the setpoint weighting factor for the two-degree-of-freedom PID controller. Loops tuned for good load disturbance attenuation are generally underdamped for setpoint change response and can benefit from setpoint weighting. Astrom and Hagglund developed the extended tuning rules by tuning an ISA standard controller for a number of different process models using the dominant pole design and relating the settings as a function of ultimate gain, ultimate period and normalized process gain, κ.

$\kappa = \dfrac{1}{K_u K_s}$ where K_s is process static gain and

K_u is ultimate gain

PI Control:

$$\frac{K}{K_u} = f_k(\kappa) = 0.053\exp(2.9\kappa - 2.6\kappa^2), \quad \text{if } M_s = 1.4$$

$$\frac{K}{K_u} = f_k(\kappa) = 0.13\exp(1.9\kappa - 1.3\kappa^2), \quad \text{if } M_s = 2.0$$

$$\frac{T_i}{T_u} = f_{ti}(\kappa) = 0.90\exp(-4.4\kappa + 2.7\kappa^2)$$

M_s is the design parameter - sensitivity

When normalized gain κ is known, $\dfrac{K}{K_u}$, $\dfrac{T_i}{T_u}$, and $\dfrac{T_d}{T_u}$ are defined as a function of κ.

The design also provides the optimal value for the setpoint weighting factor β

$$\beta = f_\beta(\kappa) = 1.1\exp(-0.0061\kappa + 1.8\kappa^2), \text{ if } M_s = 1.4$$

$$\beta = f_\beta(\kappa) = 0.48\exp(0.4\kappa - 0.17\kappa^2), \text{ if } M_s = 2.0$$

PID Control:

$$\frac{K}{K_u} = f_k(\kappa) = 0.33\exp(-0.31\kappa - 1.0\kappa^2), \quad \text{if } M_s = 1.4$$

$$\frac{K}{K_u} = f_k(\kappa) = 0.72\exp(-1.6\kappa + 1.2\kappa^2), \quad \text{if } M_s = 2.0$$

$$\frac{T_i}{T_u} = f_{ti}(\kappa) = 0.67\exp(-1.4\kappa + 0.008\kappa^2)$$

$$\frac{T_d}{T_u} = f_{td}(\kappa) = 0.16\exp(-1.2\kappa - 0.32\kappa^2)$$

$$\beta = f_\beta(\kappa) = 0.58\exp(-1.3\kappa + 3.5\kappa^2), \text{ if } M_s = 1.4$$

$$\beta = f_\beta(\kappa) = 0.25\exp(0.56\kappa - 0.12\kappa^2), \text{ if } M_s = 2.0$$

5. Internal Model Control Tuning

IMC tuning rules for self-regulating processes use the three model parameters for a first order plus deadtime model and a user-defined filter time constant to determine PID controller settings. The filter time constant should be set to the desired closed loop time constant. A larger value of filter time constant gives more damped tuning. Lambda tuning rules use a parameter, λ, which is also the desired closed loop time constant for a self-regulating process. The closed loop time constant is usually chosen to be longer than the open loop time constant, perhaps as much as three times longer. This is to ensure robustness in the event of inaccuracy in model identification and changing process conditions.

Process analysis tools have been used to assist in choosing the closed loop time constant to achieve minimum variance control [10]. The closed loop time constant may be chosen as a function of the corner period of the power spectrum of the process output in manual control (corner period = $2\pi\lambda$). A proper closed loop time constant will attenuate slow disturbances without amplifying noise in the process measurement. The following tuning rules are given in [9] (IMC) and [10] (Lambda).

IMC Tuning Rules (self-regulating process)

PI Control:

$$K = \frac{T_c}{K_s(\tau_f + T_{dt})} \qquad T_i = T_c$$

where τ_f is filter time constant

PID Control (ISA standard form):

$$K = \frac{T_c + T_{dt}/2}{K_s(\tau_f + T_{dt}/2)}$$

$$T_i = T_c + T_{dt}/2$$

$$T_d = \frac{T_c T_{dt}}{2T_c + T_{dt}}$$

Lambda Tuning Rules (self-regulating process)

PI Control:

$$K = \frac{T_c + T_{dt}/4}{K_s\lambda} \qquad T_i = T_c + T_{dt}/4$$

λ defines the desired closed loop time constant.

PID Control:

There is no direct form of Lambda Tuning for the PID controller using a first order plus deadtime process model.

Model based control of integrating processes enables the objective of averaging control to be achieved. Rather than attempting to maintain tight control by aggressively moving the manipulated variable, averaging control uses the tank, in the case of level control, to absorb disturbances. By allowing level to swing within limits, variability in outlet flow is reduced minimizing disturbances to downstream processes.

The IMC rule given below should be used for integrating processes when deadtime is significant. The IMC and Lambda results are equivalent when there is no deadtime. For integrating processes the tuning parameters, τ_f and λ, are the desired time for the disturbance effect

to be arrested, i.e. the time the process output begins to recover following a disturbance. Increasing the value τ_f and λ relaxes control.

IMC Tuning Rule (integrating process)

PI control:

$$K = \frac{2\tau_f + T_d}{K_i(\tau_f + T_d/2)^2} \qquad T_i = 2\tau_f + T_d$$

$K_i = \frac{\Delta y}{\Delta u \Delta t}$ - integrating process gain

Δy - % change in the process output

Δu - % change in the process input

Δt - time interval over which Δy and Δu are calculated

Lambda Tuning Rule (integrating process)

PI control:

$$K = \frac{4}{K_i T_i} \qquad T_i = 2\lambda$$

Model based tuning using IMC or Lambda rules has some advantages over other tuning rules. Controllers are less sensitive to noise, valve life is prolonged, and process variability may be minimized. However, the pole-zero cancellation approach results in poor load disturbance performance when the process time constant is long. A technique suggested to overcome this problem [9] is to apply the integrating process model in place of the self-regulating model. Keep in mind that the relay oscillation auto-tuning technique is capable of identifying both models for a self-regulating process. Using the IMC rule for an integrating process may improve the performance of a self-regulating process with a long time constant.

A technique has been suggested to obtain both minimum variance control and good load response [10]. Controller settings may be scheduled as a function of deviation between process output and setpoint. When the error is small, settings are derived from IMC or Lambda tuning rules; if the error is larger, tuning rules such as the modified Ziegler-Nichols rules might be applied.

Model based tuning may be extended to the Smith Predictor deadtime compensating PID controller. The first order plus deadtime process model provides the Smith Predictor tuning parameters of process gain, time constant, and deadtime. The PI controller settings may be determined using IMC or Lambda rules applying the gain and time constant from the process model, with little or no deadtime.

The enhanced relay oscillation tuning method may also be used to auto-tune the SISO fuzzy logic controller described in [16-17]. The process parameters needed to tune the scaling factors are the ultimate period, ultimate gain, apparent deadtime, and apparent time constant. The change-in-error scaling factor is set as a function of the ratio of deadtime to time constant. The change-in-output scaling factor is calculated from ultimate gain and change-in-error scaling factor. The error scaling factor is calculated from ultimate period, scan rate, and change-in-error scaling factor.

6. Conclusion

Relay oscillation auto-tuning may be extended to identify a first-order plus deadtime or integrating process model in addition to ultimate gain and ultimate period. This capability offers a wider choice of tuning rules and the potential for improved loop performance.

The ability to specify relative performance and robustness is an attractive feature of model based tuning rules. The relay oscillation method is easier to automate than an alternative open loop tuning method and may be less disruptive to the process.

References

[1] Ziegler, J. G. and Nichols, N. B.: “Optimum Settings for Automatic Controllers”, *Transactions of the ASME,* Vol. 64, Nov. 1942, pp. 759 - 768.

[2] Ziegler, J. G. and Nichols, N. B.: “Optimum Settings for Automatic Controllers”, *Transactions of the ASME,* Vol. 115, June 1993, pp. 220 - 222.

[3] Bialkowski W.L. and Haggman, Brian:“Quarter Amplitiude Damping Method Is No Longer The Industry Standard”, *American Papermaker,* March 1992

[4] Astrom, K. J. and Hagglund, T.: “Automatic tuning of PID controllers”, ISA 1988, Research Triangle Park, NC, USA.

[5] Astrom, K. J., and Hagglund, T.: “A frequency domain method for automatic tuning of simple feedback loops”, IEEE 23rd Conference on Decision and Control, Las Vegas, Dec. 1984.

[6] Astrom, K. J., and T. Hagglund, “Automatic tuning of simple regulators with specifications on phase and amplitude margins”, *Automatica,* 20, 1984, pp. 645-651.

[7] McMillan, Gregory K., Wojsznis, Willy K. and Meyer, Ken: “Easy Tuner for DCS”, ISA 1993, Chicago.

[8] Fisher-Rosemount Systems Inc., “Installing and Using the Type ACS401 Intelligent Tuner”, November 1992.

[9] Chien, I-Lung and Fruehauf, P. S.: “Consider IMC Tuning to Improve Controller Performance”, *Chemical Engineering Progress,* October 1990.

[10] Bialkowski, W. L.: “Control Engineering Course Material”, *Entech Seminar* 1991, Toronto, Canada.

[11] Wojsznis, W. K. and Blevins, T. L.: “ System and Method for Automatically Tuning a Process Controller”, US patent pending No. 08/070 090.

[12] Hang, Chang C., Lee, Tong H. and Ho, Weng K.: “Adaptive Control”, ISA 1993.

[13] Shen, Shih-Haur and Yu, Cheng-Ching: “Use of Relay-Feedback Test for Automatic Tuning of Multivariable Systems”, *AIChE Journal*, April 1994

[14] Vasnani, Vinod U.: “Towards Relay Feedback Auto-Tuning of Multi-Loop Systems”, Ph.D. Thesis, National University of Singapore, Singapore, 1994

[15] Astrom, K.J. and Hagglund, T.: “PID Controllers: Theory, Design, and Tuning”, 2nd Edition, ISA 1995

[16] Qin, S. Joe: “Auto-Tuned Fuzzy Logic Control”, American Control Conference, Baltimore, 1994

[17] Fisher-Rosemount Systems, Inc., “Installing and Using Type ACS201 Intelligent Fuzzy Logic Control”, October 1994.