Disponible en www.sciencedirect.com

ScienceDirect

Revista Iberoamericana de Automática e Informática industrial 9 (2012) 182–193

www.elsevier.es/RIAI

Estudio comparativo de algoritmos de auto-ajuste de controladores PID. Resultados del Benchmark 2010-2011 del Grupo de Ingeniería de Control de CEA.

J. A. Romero-Pérez[a,*], O. Arrieta[b,c], F. Padula[d], G. Reynoso-Meza[e], S. Garcia-Nieto[e], P. Balaguer[a]

[a] *Departamento de Ingeniería de Sistemas Industriales y Diseño. Universidad Jaume I*
Campus del Riu Sec, E-12080 Castelló de la Plana, España
[b] *Departament de Telecomunicació i d'Enginyeria de Sistemes. Universitat Autònoma de Barcelona*
08193 Bellaterra, Barcelona, España
[c] *Departamento de Automática. Escuela de Ingeniería Eléctrica. Universidad de Costa Rica*
11501-2060 San José, Costa Rica
[d] *Dipartimento di Ingegneria dell'Informazione. Università degli Studi di Brescia*
Via Branze 38, 25213 Brescia, Italia
[e] *Instituto de Automática e Informática Industrial. Universidad Politécnica de Valencia*
Edificio 8G - Acceso D - Planta 3. Camino de Vera, s/n, 46022 Valencia, España

Resumen

En este artículo se comparan tres métodos de auto-ajuste de controladores PID que consideran diferentes tipos de experimentos para obtener información de la dinámica del proceso y distintos métodos de cálculo de los parámetros del controlador para minimizar el efecto de las perturbaciones. Los métodos fueron presentados en el concurso anual organizado por el grupo de Ingeniería de Control de CEA-IFAC del curso 2010-2011. Para la comparación se aplica la metodología de evaluación de algoritmos de auto-ajuste que tiene en cuenta tanto la fase de experimento como las prestaciones que se consiguen en la fase de control.

Palabras Clave: PID, auto-ajuste, rechazo de perturbaciones.

1. Introducción

Las técnicas de auto-ajuste de los controladores PI-PID son una característica muy demandada en la industria. Con la aplicación de los métodos de auto-ajuste se busca minimizar el tiempo y el grado de experiencia necesarios para ajustar sus parámetros para controlar un proceso determinado. Como consecuencia de lo anterior, muchos productos comerciales incluyen rutinas de auto-ajuste. Sin embargo, en muchas ocasiones, tales rutinas de auto-ajuste no dan buenos resultados o los garantizan bajo condiciones particulares (Greg Baker, 2009). Usualmente, dichas condiciones están en función del índice de controlabilidad del proceso o dependen de un conocimiento previo del mismo (por ejemplo, si la planta tiene o no un integrador). Por ello, nuevas técnicas de auto-ajuste, que sean capaces de hacer frente a una gran variedad de plantas y procesos son siempre foco de nuevas investigaciones.

En general el funcionamiento de un controlador con auto-ajuste tiene dos fases. La primera es la fase de experimento, durante la cual se obtiene información del comportamiento dinámico del sistema. Esta fase finaliza con el cálculo de los parámetros del controlador, haciendo uso de la información experimental obtenida. La segunda fase es la de control, en la que el controlador pasa a controlar el sistema para cumplir las especificaciones deseadas.

Desde un punto de vista práctico, un método de auto-ajuste debe cumplir dos requisitos: 1) Lograr un ajuste adecuado del controlador y de esta forma garantizar el comportamiento deseado en la fase de control, por ejemplo minimizar el efecto de las perturbaciones. Además debe cumplir las especificaciones de robustez. 2) Conseguir el requisito anterior provocando la menor excitación posible en el sistema durante la etapa de experimento. Por ejemplo, mediante un experimento corto en el cual la salida del sistema no alcance valores excesivamente alejados del valor nominal de funcionamiento.

El cumplimiento de estos dos requisitos simultáneamente supone un compromiso entre uno y otro: mientras mayores limitaciones se tengan en la fase de experimento para aplicar cambios en la entrada del sistema, menor será la información de la

* Autor en correspondencia
Correos electrónicos: romeroj@uji.es (J. A. Romero-Pérez), Orlando.Arrieta@uab.cat (O. Arrieta), fabrizio.padula@ing.unibs.it (F. Padula), gilreyme@posgrado.upv.es (G. Reynoso-Meza), sergarro@isa.upv.es (S. Garcia-Nieto), pbalaguer@uji.es (P. Balaguer)

doi:10.1016/j.riai.2012.02.009

dinámica a partir de la cual realizar el ajuste del controlador que garantice cumplir los requerimientos del sistema en bucle cerrado. Un método de auto-ajuste será mejor en la medida que cumpla en mayor grado los dos requisitos anteriores.

En este artículo se comparan tres métodos de auto-ajuste de controladores PID que consideran diferentes tipos de experimentos para obtener información de la dinámica del proceso y distintos métodos de cálculo de los parámetros del controlador. Los métodos fueron presentados en el concurso anual organizado por el grupo de Ingeniería de Control del Comité Español de Automática (CEA-IFAC), del curso 2010-2011. Para la comparación se aplica la metodología de evaluación de algoritmos de auto-ajuste propuesta en Romero and Sanchis (2011), la cual tiene en cuenta tanto la fase de experimento como las prestaciones que se consiguen en el rechazo de perturbaciones.

La estructura del artículo es la siguiente. Inicialmente, en la Sección 2, se hace un planteamiento formal del problema de auto-ajuste que se aborda en el artículo. En las Secciones 3, 4 y 5 se presentan los tres métodos de auto-ajuste que serán comparados en este trabajo. Los métodos se han denominado *auto-ajuste evolutivo*, *auto-ajuste iterativo* y *auto-ajuste directo* atendiendo a la técnica usada para el cálculos de los parámetros de los controladores. En la sección 6 se describe de forma breve las principales características de la metodología usada para la evaluación de los algoritmos, la cual puede ser consultada de manera más extensa en Romero and Sanchis (2011). La Sección 7 se dedica al estudio comparativo de los tres métodos de auto-ajuste a partir de los resultados obtenidos de la aplicación de la metodología. Finalmente, las conclusiones se presentan en la sección 8.

2. Planteamiento del problema

El sistema de control SISO que se considera en este artículo se muestra en la Fig. 1.

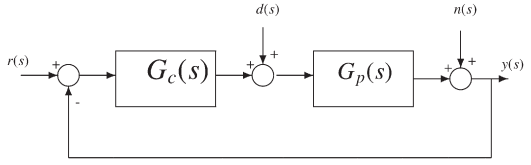

Figura 1: Sistema de Control SISO.

La función de transferencia del controlador PID es:

$$G_c(s) = K_c\left(1 + \frac{T_d s}{1 + \frac{T_d s}{N}} + \frac{1}{T_i s}\right) \tag{1}$$

El objetivo es ajustar el controlador PID de forma automática para conseguir minimizar el efectos de la perturbación d bajo unas condiciones de robustez adecuadas del sistema controlado, teniendo en cuenta que el modelo del proceso $G_p(s)$ y el ruido de medida n son desconocidos.

3. Auto-ajuste evolutivo

El algoritmo de auto-ajuste evolutivo se basa en la solución del problema de optimización no-convexa desarrollado en Åström et al. (1998); Panagopoulos et al. (2002), que considera la maximización de la ganancia integral del controlador sujeto a restricciones que garantizan la robustez del lazo de control. Dicho problema de optimización ha sido resuelto utilizando algoritmos evolutivos con anterioridad (Reynoso-Meza et al., 2011b,c), sin embargo, no ha sido implementado aún el ámbito de auto-ajuste. El método de auto-ajuste propuesto aquí integra el uso de la optimización evolutiva, de ahí el nombre de *auto-ajuste evolutivo*.

Dada una planta $G_p(s)$, se desea ajustar un controlador que maximize la ganancia integral k_c/T_i, dado que se busca optimizar para el rechazo de perturbaciones. Luego entonces, se plantea el problema de optimización:

$$\min_{\boldsymbol{x}\in\Re^4} J(\boldsymbol{x}) = -\frac{K_c}{T_i} \tag{2}$$

considerando como planta un proceso de primer orden con retardo de tiempo (PORT):

$$G_p(s) = \frac{K}{Ts+1}\mathrm{e}^{-Ls} \tag{3}$$

Es bien conocido que maximizar el desempeño del controlador sin ninguna otra consideración, conduce al cálculo de controladores con poco margen de robustez. En tal caso, al implementar el controlador en la planta real, se pueden llegar a tener comportamientos inesperados. Por tanto, deben considerarse algunas restricciones en el anterior problema de optimización, para garantizar un mínimo de robustez en el lazo de control. A continuación, se proponen 3 tipos de restricciones para lograr un desempeño aceptable del controlador en el proceso real.

3.1. Restricciones

El valor máximo de la magnitud de la función de sensibilidad $M_s = \max|S(\mathrm{j}\omega)| = \max\left|\frac{1}{1+G_p(\mathrm{j}\omega)G_c(\mathrm{j}\omega)}\right|$ es un indicador común para valorar la robustez del lazo de control (Åström and Hägglund, 2006). Valores aceptados por la comunidad de control se encuentran en el intervalo $M_s \in [1{,}2; 2{,}0]$ Åström et al. (1998); Panagopoulos et al. (2002). Luego entonces, el problema de optimización planteará un valor mínimo aceptable para el controlador M_s^{target}. De tal manera, solo serán consideradas las soluciones tales que:

$$M_s \leq M_s^{target} \tag{4}$$

El cálculo de M_s es el que requiere de mayor coste computacional. Por tanto, debe buscarse una aproximación práctica de M_s sobre un rango (o conjunto de valores) de frecuencias determinado para poder asegurar su cómputo en el período de muestreo dado.

Dado que se parte del ajuste de un controlador PID con filtro de la derivada para una planta de primer orden con retardo, la función de sensibilidad del sistema viene dada por:

$$\begin{aligned} S(\mathrm{j}\omega) &= \frac{1}{1+G_c(\mathrm{j}\omega)G_p(\mathrm{j}\omega)} \\ &\approx \frac{(a\omega^4+b\omega^2)+(c\omega^3+d\omega)\mathrm{j}}{(e\omega^4+f\omega^2+g\omega^0)+(h\omega^3+i\omega)\mathrm{j}} \end{aligned} \quad (5)$$

donde, considerando una aproximación de Padé de primer orden $e^{-Ls} \approx \frac{1-\frac{L}{2}s}{1+\frac{L}{2}s}$ para el retardo, se tiene que:

$$\begin{aligned} a &= \frac{T_i \cdot T_d \cdot L}{2N} \\ b &= -T_i\left(T+\frac{T_d}{N}+\frac{L}{2}\right) \\ c &= -\frac{T_i \cdot T_d \cdot T}{N} - T_i \cdot L\left(T+\frac{T_d}{N}\right) \\ d &= T_i \end{aligned}$$

$$\begin{aligned} e &= \frac{T_i \cdot T_d \cdot T \cdot L}{2N} \\ f &= -T_i\left(T+\frac{T_d}{N}+\frac{L}{2}\right) \\ &\quad -k_c \cdot k_p \cdot T_i \cdot T_d\left(\frac{1}{N}+1\right) \\ &\quad +\frac{k_c \cdot k_p \cdot L}{2}\left(T_i+\frac{T_d}{N}\right) \\ g &= k_c \cdot k_p \\ h &= -T_i\left(T+\frac{T_d}{N}+\frac{L}{2}\right) \\ &\quad +\frac{k_c \cdot k_p \cdot L \cdot T_i \cdot T_d}{2}\left(\frac{1}{N}+1\right) \\ i &= T_i + k_c \cdot k_p\left(T_i+\frac{T_d}{N}\right) \\ &\quad -\frac{k_c \cdot k_p \cdot L}{2} \end{aligned} \quad (6)$$

A partir de lo anterior, es sencillo calcular la magnitud para un determinado valor de $\omega \in [\underline{\omega}, \overline{\omega}]$.

La segunda restricción trata de garantizar márgenes razonables en la acción de control que ejercerá el controlador sobre el proceso. Para ello, se buscará acotar la máxima acción de control para la planta identificada. Se propone emplear la siguiente restricción:

$$K_c + \text{máx}(1, L) \cdot K_c/T_i + K_c \cdot T_d \leq K_u \quad (7)$$

donde K_u es la ganancia última estimada del proceso a controlar.

Finalmente, la tercera restricción es empleada para considerar el ruido en la medición y su efecto en la acción de control. Un indicador plausible es:

$$M_u = \text{máx}\left|\frac{G_c(\mathrm{j}\omega)}{1+G_c(\mathrm{j}\omega)G_p(\mathrm{j}\omega)}\right| \quad (8)$$

Para evitar el cálculo de M_u, se emplea la aproximación (Åström and Hägglund, 2006) :

$$M_u \approx k_c \cdot N \quad (9)$$

Para mantener dicho valor acotado, se emplea la restricción $M_u \approx K_c \cdot N \leq K_u$. Esta restricción es importante para que el mecanismo de optimización pueda considerar la sensibilidad al ruido en la medida, y pueda lograr un ajuste más preciso del filtro de la derivada.

3.2. Planteamiento

Los algoritmos evolutivos representan una aproximación confiable cuando el ajuste de controladores se plantea como un problema de optimización (Fleming and Purshouse, 2002). Dada su naturaleza de optimizadores globales, son capaces de hacer frente a problemas de optimización no convexos, no lineales y altamente restrictivos (Herreros et al., 2002; Tavakoli et al., 2007; Kim et al., 2008; Nobakhti and Wang, 2008; Iruthayarajan and Baskar, 2009; Xue et al., 2010; Iruthayarajan and Baskar, 2010; Zhao et al., 2011). Por ello, se proponen como herramienta de optimización en la propuesta de mecanismo de auto-ajuste mediante la solución del problema de optimización con restricciones siguiente:

$$\min_{\boldsymbol{x}\in\Re^4} \mathfrak{J}(\boldsymbol{x}) = \begin{cases} J(\boldsymbol{x}) \ \textbf{\textit{if}} \ \sum_{k=1}^{3} \phi_k(\boldsymbol{x}) = 0 \\ \sum_{k=1}^{3} \phi_k(\boldsymbol{x}) \ \textbf{\textit{otherwise}} \end{cases} \quad (10)$$

donde:

$$\phi_1(\boldsymbol{x}) = \frac{\text{máx}\{0, M_s - M_s^{target}\}}{M_s^{target}} \quad (11)$$

$$\phi_2(\boldsymbol{x}) = \frac{\text{máx}\{0, k_c + \text{máx}(1, L) \cdot \frac{k_c}{T_i} + k_c \cdot T_d - K_u\}}{K_u} \quad (12)$$

$$\phi_3(\boldsymbol{x}) = \frac{\text{máx}\{0, k_c \cdot N - K_u\}}{K_u} \quad (13)$$

A continuación, se describe el algoritmo de auto-ajuste de controladores PID que incorpora el anterior problema de optimización. El mismo se compone de cinco etapas: estimación de ruido, experimentación, identificación, inicialización y optimización.

1. Durante un periodo de tiempo determinado (en estado estable), se medirá la variación en la señal de medida para estimar la amplitud del ruido. Ello es necesario y fundamental para poder elegir adecuadamente la amplitud de la prueba de relé, así como el intervalo de histéresis.
2. Se empleará una prueba de relé con offset (Hang et al., 2002; Wang et al., 1997) para obtener información del modelo. Este tipo de prueba es una ampliación a la conocida prueba de relé, que permite estimar un modelo razonable de la planta a controlar.

3. Con los datos obtenidos en el proceso de experimentación mediante la prueba de relé con offset, se procederá a identificar un modelo de primer orden con retardo (Hang et al., 2002; Wang et al., 1997).
4. El proceso de optimización evolutiva parte de una población de vectores de decisión. Para ello, se inicializan de forma aleatoria un conjunto de vectores dentro de un espacio de búsqueda acotado. Para mejorar la convergencia del algoritmo de auto-ajuste, se incluyen en la población una serie de soluciones factibles para resolver el problema. Es decir, se incluyen controladores que son ajustados por medio de reglas heurísticas y/o teóricas a partir del modelo de primer orden con retardo identificado
5. Durante el proceso de optimización se llevan a cabo dos procesos principales: la generación del *offspring* (conjunto de soluciones candidatas) y la selección de las mejores soluciones encontradas.

Como técnica evolutiva para la optimización, se utilizará el algoritmo de Evolución Diferencial (en lo sucesivo DE) (Storn and Price, 1997; Storn, 2008). Este algoritmo cuenta con dos operadores principales para generar su *offspring* (nueva población de individuos): mutación y cruce. Existen varias versiones fundamentales del algoritmo, de acuerdo a la naturaleza de los operadores anteriores. En este trabajo se empleará la versión conocida como DE/*current-to-rand*/2 (Price, 1999), que utiliza una estrategia de recombinación lineal no invariante rotacional como operador de cruce.

Mutación Para cada vector del espacio de decisión $\boldsymbol{x}^i|_k$ (vector padre), se construye un vector mutante $\boldsymbol{v}^i|_k$ en cada generación k de acuerdo a la ecuación (14):

$$\boldsymbol{v}^i|_k = \boldsymbol{x}^{r_1}|_k + F(\boldsymbol{x}^{r_2}|_k - \boldsymbol{x}^{r_3}|_k) \tag{14}$$

donde $r_1 \neq r_2 \neq r_3 \neq i$ y F es conocido como el factor de escalado.

Cruce (Recombinación lineal) Para cada vector del espacio de decisión $\boldsymbol{x}^i|_k$ y su vector mutante $\boldsymbol{v}^i|_k$, se construye un vector de prueba (vector hijo) $\boldsymbol{u}^i|_k = [u_1^i|_k, u_2^i|_k, \ldots, u_n^i|_k]$:

$$\boldsymbol{u}^i|_k = \boldsymbol{x}^i|_k + F_i(\boldsymbol{v}^i|_k - \boldsymbol{x}^i|_k) \tag{15}$$

donde F_i es el factor de escalado para la recombinación lineal.

Como mecanismo de selección, se empleará una comparación por parejas entre cada vector padre y su vector hijo. De tal forma, se elegirá el que tenga el mejor *fitness* para conformar la población en la siguiente generación:

$$\boldsymbol{x}^i|_{k+1} = \begin{cases} \boldsymbol{u}^i|_k & if \quad f(\boldsymbol{u}^i|_k) \leq f(\boldsymbol{x}^i|_k) \\ \boldsymbol{x}^i|_k & otherwise \end{cases} \tag{16}$$

Dadas las condiciones del *benchmark*, se debe garantizar que el anterior esquema de evolución pueda ejecutarse dentro de un periodo de muestreo. Como lo anterior no es posible, cada generación del proceso de evolución es dividida en varias llamadas del algoritmo de auto-ajuste. Así mismo, para mejorar la convergencia del algoritmo, inmediatamente que un vector hijo es creado, se le compara con su vector padre.

Finalmente, en cada ejecución del algoritmo durante la fase de optimización, el mejor controlador de la población es seleccionado. El mismo es utilizado para calcular la acción de control que se envía al sistema.

El algoritmo resultante tiene los parámetros que se muestran a continuación:

- Factor de escalado $F = 0{,}9$. Con este valor se promueve la exploración en el espacio de variables de decisión (Storn, 2008).
- Factor de escalado en la recombinación lineal $F_i = 0{,}95$. Acorde con Price (1999), un buen valor inicial para F_i lo representa $F_i = 0{,}5 \cdot (1 + F)$.
- Población del algoritmo evolutivo: $|P| = 10{\cdot}4 = 40$ (Storn and Price, 1997). Un valor heurístico recomendado para la población del algoritmo DE es diez veces el número de variables de decisión. Para garantizar la ejecución del algoritmo en cada tiempo de muestreo, 5 miembros de la población son evaluados en cada llamada del algoritmo.
- Cotas máximas: $[\overline{K_c}, \overline{T_i}, \overline{T_d}, \overline{N}] = [K_u, 2 \cdot T, 2 \cdot L, 20]$.
- Cotas mínimas: $[\underline{K_c}, \underline{T_i}, \underline{T_d}, \underline{N}] = [0, T/10, 0, 3]$.
- Rango de frecuencias (con abuso de notación) $[\omega = 1/100 : 1/25 : 8]$.
- Histéresis = $max\{0{,}01, 2 \cdot AmplitudRuido\}$
- $M_s^{target} = 1{,}8$.

4. Auto-ajuste iterativo

El método de sintonía propuesto, basado en Sanchis et al. (2010), busca la maximización de la ganancia integral del controlador ($K_i = K_c/T_i$) sujeta a las siguientes restricciones de margen de fase (ϕ_m), margen de ganancia (γ_m) e igualdad de los zeros del controlador (z_i, z_d): $\phi_m \geq \phi_{m,r}$, $\gamma_m \geq \gamma_{m,r}$, $z_d = z_i$, donde $\phi_{m,r}$ y $\gamma_{m,r}$ son los márgenes de fase y ganancia requeridos. La maximización se lleva a cabo definiendo el siguiente parámetro de sintonía, que es la relación entre la frecuencia de cruce de ganancia y el cero del controlador:

$$a = \frac{\omega_{cg}}{z_c} \quad ; \quad z_c = z_i, z_d \tag{17}$$

donde ω_{cg} es la frecuencia donde la magnitud de la respuesta en frecuencia en lazo abierto es 1, esto es, donde se mide el margen de fase.

Para un valor dado de a, la fase del controlador a la frecuencia de cruce última ω_{cg}, depende solo de a, como se puede ver en la siguiente ecuación.

$$\arg(G_c(j\omega_{cg})) = 2 \arctan(a) - \arctan\left(\frac{a}{N}\right) - \frac{\pi}{2} \tag{18}$$

Así pues, para un valor de a dado, el cálculo del controlador se lleva a cabo en dos pasos:

1. La frecuencia de cruce última (ω_{cg}) se obtiene como la frecuencia donde la fase del sistema cumple con las ecuaciones del margen de fase, es decir, donde la fase del sistema es

$$\arg(G_p(j\omega_{cg})) = -\pi + \phi_{m,r} - \arg(G_c(j\omega_{cg})) \qquad (19)$$

Entonces el valor del cero ($z_i = z_d$) se calcula como

$$z_i = z_d = \frac{\omega_{cg}}{a}$$

2. El valor de K_i se calcula entonces de la condición de magnitud unitaria en la frecuencia de cruce última.

$$|G_c(j\omega_{cg})G_p(j\omega_{cg})| = 1 \quad \rightarrow \quad K_i \qquad (20)$$

Resultando las ecuaciones:

$$K_i(a) = \frac{\omega_{cg}\sqrt{1+\frac{a^2}{N^2}}}{|G_p(j\omega_{cg})|(1+a^2)} \qquad (21)$$

3. Una vez definidos los parámetros del controlador, el margen de ganancia, γ_m, puede calcularse.

En Romero et al. (2011) se estudió la forma de las funciones $K_i(a)$ y $\gamma_m(a)$ obtenidas del diseño PID para diferentes modelos, demostrándose que en todos los casos $K_i(a)$ es una función suave con un único máximo en $a_{K_{max}}$. De la misma forma, $\gamma_m(a)$ tiene también un máximo, el cual se localiza para valores de a más pequeños que $a_{K_{max}}$. Como resultado, la función $\gamma_m(a)$ se incrementa conforme se reduce a desde $a_{K_{max}}$. Es esta propiedad la que permite desarrollar un algoritmo de sintonía que maximiza la ganancia integral del controlador (K_i) sujeta a restricciones de margen de fase.

El algoritmo de sintonía requiere la respuesta en frecuencia del proceso, que se aproxima mediante una línea recta que pasa por dos puntos identificados mediante experimentos de realimentación con relé, (Tan et al., 2005). Usando la aproximación por rectas de la respuesta de fase y de magnitud del sistema, es posible encontrar una expresión analítica del valor de $a = a_{K_{max}}$ que maximiza $K_i(a)$. Dicha expresión sólo depende de las pendientes de las rectas que aproximan la respuesta de frecuencia del sistema.

El valor $a_{K_{max}}$ es el punto de partida para el ajuste óptimo del PID. El primer paso es diseñar el PID para dicho valor de $a = a_{K_{max}}$, aplicando las ecuaciones (19)-(21). Si el margen de ganancia γ_m estimado cumple con la restricción $\gamma_{m_i} > \gamma_{m,r}$, entonces el algoritmo finaliza pues se ha encontrado el controlador óptimo. En caso contrario el algoritmo sigue buscando el valor a que cumple con el margen de ganancia establecido. Teniendo en cuenta que $\gamma_m(a)$ se incrementa conforme a se decrementa respecto $a_{K_{max}}$, la búsqueda se realiza decrementando a una cantidad Δa en cada iteración hasta que se alcanza un valor de a que cumple con el margen de ganancia.

Así pues, el algoritmo de sintonía automática propuesto se puede resumir en los siguientes pasos:

1. Realiza un experimento relé sin retardo, con amplitud μ y histéresis h. El resultado es la frecuencia ω_1 y la amplitud de oscilaciones A_1. El primer punto de la respuesta en frecuencia viene dado por la frecuencia ω_1, magnitud $M_1 = \frac{\pi A_1}{4\mu}$, y fase $\phi_1 = -\pi - \arg\left(\frac{4\mu}{\pi A_1}\left(\sqrt{1-(\frac{h}{A_1})^2} - j\frac{h}{A_1}\right)\right)$.
2. Realiza un segundo experimento relé con retardo $L_2 = \frac{(-\phi_1+\phi)\phi_1}{\phi\omega_1}$, donde ϕ es la fase del punto deseado. El resultado es la frecuencia ω_2, y la amplitud de oscilaciones A_2. El segundo punto de la respuesta en frecuencia viene dado por la frecuencia ω_2, magnitud $M_2 = \frac{\pi A_2}{4\mu}$ y fase $\phi_2 = L_2\omega_2 - \pi - \arg\left(\frac{4\mu}{\pi A_2}\left(\sqrt{1-(\frac{h}{A_2})^2} - j\frac{h}{A_2}\right)\right)$.
3. Calcula el PID óptimo, tal y como se ha descrito anteriormente, empleando la aproximación lineal, $\phi_{m,r}$ y $\gamma_{m,r}$.
4. Calcula la frecuencia a la que se mide el margen de ganancia ω_{gt1}. Si dicha frecuencia no está demasiado alejada del rango de frecuencias $[\omega_2, \omega_1]$, se da por buena la aproximación por una línea recta usada y el ajuste del PID, finalizando la sintonía. En caso contrario se estima un tercer punto de la respuesta de frecuencia. Para ello se realiza un nuevo experimento de relé con retardo $L_3 = \frac{(\phi_2-\phi_1)\log(\frac{\omega_1}{\omega_{gt1}})}{\omega_{gt1}\log(\frac{\omega_1}{\omega_2})}$. El resultado es la frecuencia ω_3, y amplitud A_3. El tercer punto de la respuesta en frecuencia viene dado por la frecuencia ω_3, magnitud $M_3 = \frac{\pi A_3}{4\mu}$ y fase $\phi_3 = L_3\omega_3 - \pi - \arg\left(\frac{4\mu}{\pi A_3}\left(\sqrt{1-(\frac{h}{A_3})^2} - j\frac{h}{A_3}\right)\right)$.
5. Diseña el controlador PID empleando la aproximación con ambas lineas rectas, con especificaciones $\phi_{m,r}$ y $\gamma_{m,r}$, y finaliza el proceso de sintonizado.

Como se puede ver en el paso 4 del algoritmo anterior, la identificación de un tercer punto de la respuesta de frecuencia se realiza sólo si es necesario, o sea, si el modelo inicial formado por una recta no cubra un rango de frecuencia suficiente para asegurar un buen ajuste del PID. En ese caso se mejora el modelo y se diseña nuevamente el controlador, de ahí el nombre de *auto-ajuste iterativo* dado al método en este artículo.

5. Auto-ajuste directo

Este método se basa en aproximar la respuesta del sistema por un modelo paramétrico que incluye un retardo de tiempo. A partir del modelo identificado se realiza el ajuste del controlador mediante una ecuaciones directas (de ahí el nombre de *auto-ajuste directo*) que relaciona los parámetros del modelo obtenido en la fase de experimento con los parámetros del PID para conseguir un ajuste que trata de minimizar el efecto de las perturbaciones.

Para los procesos con una respuesta en lazo abierto asintóticamente estable, se obtiene un modelo PORT como el indicado en la ecuación (3), mientras que para el caso del sistema con integrador el modelo a obtener es de la forma:

$$G_p(s) = \frac{K}{s}e^{-Ls} \qquad (22)$$

En esta fase del experimento, lo primero que se hace es identificar si el sistema tiene o no integrador. Para esto se calcula la pendiente media de la respuesta filtrada del sistema en diferentes intervalos. Si esta varía el sistema no sería catalogado como integrante y se procedería con la identificación de un modelo tipo PORT. Para estos casos, se emplea el método *123c*, propuesto por Alfaro (2006), que utiliza las medidas de dos puntos de la curva de reacción del proceso para calcular los parámetros de un modelo de la forma (3), haciendo:

$$\begin{aligned} K &= \frac{\Delta y}{\Delta u} \\ T &= 0{,}9102(t_{75} - t_{25}) \\ L &= 1{,}2620t_{25} - 0{,}2620t_{75} \end{aligned} \tag{23}$$

en donde t_{25} y t_{75} son los tiempos requeridos para que la respuesta del sistema alcance el 25 y 75 por ciento de su valor final.

En caso de que el cálculo inicial de la pendiente se mantenga constante, esta característica indicaría un comportamiento integrante en cuyo caso se propone identificar un modelo del tipo (22), de la siguiente manera:

$$\begin{aligned} K &= \frac{2}{\Delta u}\frac{y_{85} - y_{75}}{t_{85} - t_{75}} \\ L &= 2t_{10} \end{aligned} \tag{24}$$

donde y_{75} y y_{85} representan las medidas de la respuesta en el 75 y 85 por ciento y t_{10}, t_{75} y t_{85}, los tiempos requeridos en alcanzar los porcentajes de respuesta especificados.

Una vez que se tiene un modelo identificado, se procede al cálculo de los parámetros del controlador PID. En este caso, para los sistemas cuyo modelo es PORT, se utilizó el método propuesto por Padula and Visioli (2011), que optimiza el índice de rendimiento *IAE* en regulación, sujeto a un valor de robustez de $M_s = 2$.

Las ecuaciones de sintonía se basan en la información de (3), y son las siguientes:

$$\begin{aligned} K_c' &= \frac{1}{K}\left(0{,}2002\left(\frac{L}{L+T}\right)^{-1{,}414} + 0{,}06139\right) \\ T_i' &= T\left(0{,}446\left(\frac{L}{T}\right)^{0{,}9541} + 0{,}1804\right) \\ T_d' &= T\left(0{,}6777\left(\frac{L}{T}\right)^{0{,}4968} - 0{,}1499\right) \end{aligned} \tag{25}$$

En el caso del sistema con integrador, la sintonía empleada es la presentada en Padula and Visioli (2010), que también minimiza el *IAE* en regulación, sin embargo en este caso, dado las dinámicas del sistema, se escogió una robustez de $M_s = 1{,}4$. Los parámetros se calculan en base al modelo (22) como:

$$\begin{aligned} K_c' &= \frac{0{,}4058}{KL} \\ T_i' &= 3{,}5035L \\ T_d' &= 0{,}5267L \end{aligned} \tag{26}$$

Las dos sintonías anteriores fueron formuladas para una estructura de PID distinta a la que posee el controlador de la ecuación (1), por lo que simplemente se hace una conversión de parámetros (25) y (26), de la siguiente manera:

$$\begin{aligned} K_c &= K_c'\frac{(T_i' + T_d')}{T_i'} \\ T_i &= T_i' + T_d' \\ T_d &= \frac{T_d'T_i'}{T_i' + T_d'} \end{aligned} \tag{27}$$

En todos los casos se fija el valor del filtro derivativo como $N = 10$.

Con el objetivo de obtener una señal más limpia, se implementa un filtro discreto no-causal, mediante la función *filtfilt* de Matlab. Específicamente la función que describe el filtro utilizado es la siguiente:

$$F_y(z) = \frac{1 - 0{,}95}{1 - 0{,}95z} \tag{28}$$

en donde $|F_y(z) = 1|$.

En resumen, el algoritmo empírico consta de los siguientes pasos:

1. Realizar una prueba escalón para obtener información del sistema.
2. Determinar si el proceso a controlar se ajusta a un modelo PORT o a un sistema con integrador y retardo.
3. Calcular los parámetros del controlador PID acorde a las ecuaciones (25) o (26), según sea el caso y obtener luego las ecuaciones equivalentes (27).

6. Metodología de evaluación

Para realizar el estudio comparativo entre los tres algoritmos descritos en las secciones anteriores se aplicó la metodología de evaluación de métodos de auto-ajuste propuesta en Romero and Sanchis (2011). La metodología de evaluación tiene en cuenta el comportamiento de un algoritmo en la fase de experimento, así como al desempeño del sistema controlado en la fase de control. A continuación se relacionan las características más importantes de la metodología de evaluación usada para la comparación:

- El índice de desempeño que se mide en la fase de control (I_c) tiene en cuenta la atenuación de las perturbaciones y la robustez del bucle de control.
- El índice de desempeño que se mide en la fase de experimento (I_e) tiene en cuenta tanto la duración del experimento como la variación de la salida durante el mismo.
- El cálculo de los índices I_e e I_c se hace de forma que los resultados obtenidos para los distintos modelos sean comparables entre sí. Para ello se realiza una normalización de los índices teniendo en cuenta la dinámica concreta de cada modelo.

- La normalización del índice I_c se hace respecto de resultados de comportamiento óptimo de controladores PID publicados en Panagopoulos et al. (2002), de manera que los valores se pueden interpretar como relativos a esos resultados. Para ello se calcula $IAE_p^* = \frac{IAE_p}{IAE_2}$, donde IAE_p es el IAE que se obtiene con el PID a evaluar, e IAE_2 es el valor de IAE que se obtiene con un PID óptimo, que minimiza el IAE considerando una $M_s = 2$, cuyos valores han sido reportados en Panagopoulos et al. (2002) para cada uno de los modelos (29)-(33).

  El índice IAE_p^* indica cuantas veces es mayor el IAE conseguido con los algoritmos propuestos respecto del IAE que se obtiene con un ajuste óptimo de PID, que minimiza el IAE con la restricción $M_s = 2$, suponiendo conocido el modelo exacto, según Panagopoulos et al. (2002).

- La normalización del índice I_e se hace respecto de los resultados obtenidos mediante el método de realimentación con relé para cada modelo, por lo que los valores del índice pueden ser interpretados como relativos a esos resultados. Para ello se calcula $IAE_e^* = \frac{IAE_e}{IAE_R}$ y $T_e^* = \frac{T_e}{T_R}$, donde IAE_e y T_e son el IAE y el tiempo del experimento e IAE_R y T_R son el IAE y el periodo de una oscilación de la salida del sistema cuando se realiza un experimento de realimentación con un relé de amplitud unitaria.

- Para hacer una evaluación más realista se tiene en cuenta un ruido de medida en la salida del sistema. En la práctica, el ruido de medida limita los valores mínimos de la entrada ya que la relación entre la salida del sistema y el ruido de medida debe permitir extraer la información de la dinámica necesaria para el ajuste del controlador.

El comportamiento es evaluado para un conjunto de modelos que representan los tipos de dinámicas más comunes en la industria de procesos (G_p). En concreto se trata de los siguientes modelos:

$$G_1(s) = \frac{1}{s(s+1)^3} \tag{29}$$

$$G_2(s) = \frac{e^{-5s}}{(s+1)^3} \tag{30}$$

$$G_3(s) = \frac{1}{(s+1)(1+0{,}2s)(1+0{,}2^2 s)(1+0{,}2^3 s)} \tag{31}$$

$$G_{4,5,6,7}(s) = \frac{1}{(s+1)^\alpha} \quad \alpha = 4,5,6,7 \tag{32}$$

$$G_8(s) = \frac{-2(s-2)}{(s+1)^3} \tag{33}$$

A partir de los resultados obtenidos de los índices I_e e I_c para cada modelo, se calcula los índices totales para las fases de experimento y control mediante las ecuaciones (34) y (35) respectivamente, donde el sub-índice j corresponde al número del modelo en las ecuaciones (29)-(33).

$$I_{e\_total} = \sqrt[8]{\prod_{j=1}^{8} I_{e_j}} \tag{34}$$

$$I_{c\_total} = \sqrt[8]{\prod_{j=1}^{8} I_{c_j}} \tag{35}$$

Además, se calculan otros valores que permiten una mejor comparación de los algoritmos como son: la media geométrica de los tiempos de los experimentos ($T^*_{e_{total}}$), una medida de la amplitud de la salida durante los experimentos calculada como el cociente entre la media geométrica del IAE y de los tiempos de experimentos ($IAE^*_{e_{total}}/T^*_{e_{total}}$), la media geométrica del IAE de las salidas ante perturbaciones de un escalón unitario ($IAE^*_{p_{total}}$) y la media geométrica de los valores de M_s conseguido en cada diseño ($M_{s_{total}}$). Esto valores se calculan mediante las ecuaciones (36).

$$T^*_{e_{total}} = \sqrt[8]{\prod_{j=1}^{8} T^*_{e_j}}, \quad IAE^*_{e_{total}} = \sqrt[8]{\prod_{j=1}^{8} IAE^*_{e_j}} \tag{36a}$$

$$IAE^*_{p_{total}} = \sqrt[8]{\prod_{j=1}^{8} IAE^*_{p_j}}, \quad M_{s_{total}} = \sqrt[8]{\prod_{j=1}^{8} M_{s_j}} \tag{36b}$$

Finalmente, se combinan los índices de la fase de control y la de experimento para calcular el índice global, que tendrá la expresión:

$$I_{total} = I_{c\_total}\sqrt[4]{I_{e\_total}} \tag{37}$$

La raíz cuarta en el índice de experimento es para favorecer en la evaluación a aquellos métodos de auto-ajuste que consigan una mejor respuesta del sistema controlado.

7. Estudio comparativo

En esta sección se presentan los resultados de la comparación entre los tres métodos de auto-ajuste descrito en las secciones anteriores.

7.1. Características generales de los algoritmos

Las características fundamentales de los tres algoritmos se han resumido en la tabla 1. Como se puede ver, los métodos difieren en varios aspectos. En cuanto al tipo de experimentos para extraer información de la dinámica del sistema, se usa tanto la entrada escalón para obtener un modelo paramétrico que tenga en cuenta el retardo de tiempo, como experimentos de realimentación con relé del cual se usan dos alternativas, una que identifica la frecuencia y ganancia últimas del proceso y a a partir de ellas calcular un modelo paramétrico, y otra que identifica varios puntos de la respuesta de frecuencia (RF), conformando de esta forma un modelo no paramétrico.

Respecto al método usado para el cálculo de los parámetros de los controladores también existen diferencias importantes. En los métodos evolutivo e iterativo se plantea el cálculo de

los parámetros de los controladores mediante una optimización en-línea que trata de maximizar la ganancia integral del controlador sujeto a restricciones para garantizar la robustez del diseño. En el método iterativo se propone un algoritmo de optimización unidimencional a partir de la re-parametrización del controlador, reduciendo de esta forma el coste computacional de la implementación. En el caso del algoritmo evolutivo, la optimización se resuelve mediante Algoritmos Genéticos con restricciones en el máximo del módulo de la función de sensibilidad y las variaciones en la acción de control. Dada la naturaleza estocástica del método de optimización utilizado en este algoritmo, los resultados pueden estar sujeto a ciertas variaciones. Para su estudio se ha llevado a cabo un análisis estadístico de los resultados que brinda el algoritmo, y se ha comprobado que dichas variaciones son, en general, poco significativas (Reynoso-Meza et al., 2011a). En el algoritmo directo el cálculo los parámetros del PID se realiza mediante unas ecuaciones que relacionan los parámetros de un modelo PORT con los parámetros de un controlador que minimiza el *IAE* de la perturbación garantizando un comportamiento robusto (M_s = 2). Dichas ecuaciones fueron obtenidas en Padula and Visioli (2010, 2011).

Tabla 1: Resumen de las característica de los métodos de auto-ajuste

Método	**Experim.**	**Modelo**	**Ajuste**
Evolutivo	Relé	PORT	Optimización heurística
Directo	Escalón	PORT	Reglas directas
Iterativo	Relé/iterativo	Puntos de la RF	Optimización simplificada

Además de las características mostradas en la Tabla 1, los métodos que se se comparan en este artículo difieren en otros aspectos. Uno de ellos es el cálculo del coeficiente de filtrado (N) del controlador PID. El algoritmo evolutivo plantea el cálculo de ese parámetro como parte del procesos de optimización, mientras que los algoritmos directo e iterativo consideran un valor de N establecido a priori, concretamente $N = 10$. Así mismo, el algoritmo evolutivo incluye una restricción en las variaciones de la acción de control, y por el contrario, los algoritmos directo e iterativo no incluyen dicha restricción.

7.2. Resultados de las simulaciones

En las Figs. 2-6 se muestran las respuestas temporales de los tres método para los modelos G_1, G_2, G_3, G_5 y G_8 respectivamente, donde se distinguen de forma clara las fases de experimento y control.

Se puede observar que en todos los casos, excepto para el modelo G_2, los experimentos de menor duración corresponden al método de auto-ajuste directo que se basa en la respuesta escalón para el cálculo del modelo. Para los método evolutivo e iterativo la duración de los experimentos es mayor, sobre todo en caso del método iterativo donde se identifican varios puntos

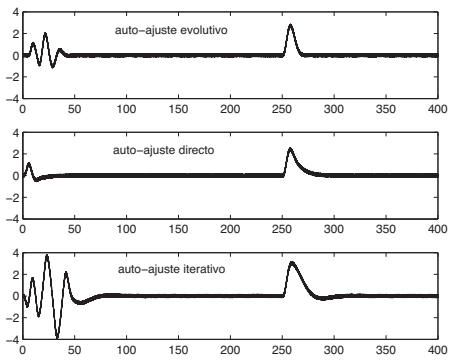

Figura 2: Experimento y respuesta a una entrada escalón unitario en la perturbación conseguida para el modelo G_1.

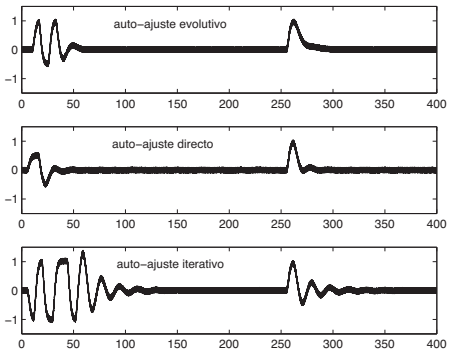

Figura 3: Experimento y respuesta a una entrada escalón unitario en la perturbación conseguida para el modelo G_2.

de la respuesta de frecuencia mediante la realización de varios experimento de relé de forma consecutiva.

Con respecto a la amplitud de la salida del sistema durante la fase de experimento se aprecia que en general tienen el mismo orden de magnitud y dependiendo del modelo es mayor en uno u otro caso. En cuanto a la fase de control se aprecia que en la mayoría de los casos se obtienen respuestas poco oscilatorias y sólo en el método iterativo aplicado el modelo G_2 aparecen una respuesta algo más oscilatoria que en el resto de casos estudiado.

En las Figs. 7, 8 y 9 se muestran los resultados de los índices conseguidos por los tres algoritmos para cada uno de los modelos considerados en el estudio. Las gráficas de la primera columna de cada una de estas figuras corresponden a índices de la fase de experimento, mientras que en las gráficas de la segunda columna se muestran índices de la fase de control. De la fase de experimento se muestra el tiempo de experimento normalizado (T_e^*), la medida de la amplitud de la salida dada por el cociente (IAE_e^*/T_e^*) y el índice total de la fase de experimento (I_e). De la fase de control se muestra la M_s, el *IAE* normalizado de la salida ante una perturbación escalón unitario (IAE_p^*) y el índice total de la fase de control (I_c).

Se puede apreciar que los menores tiempos de experimento

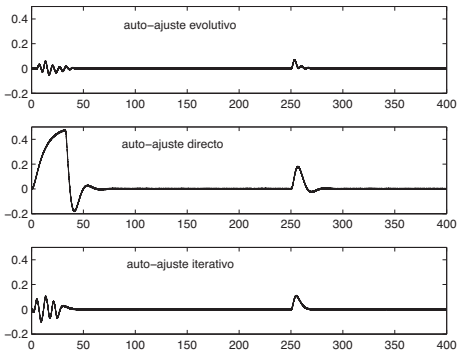

Figura 4: Experimento y respuesta a una entrada escalón unitario en la perturbación conseguida para el modelo G_3.

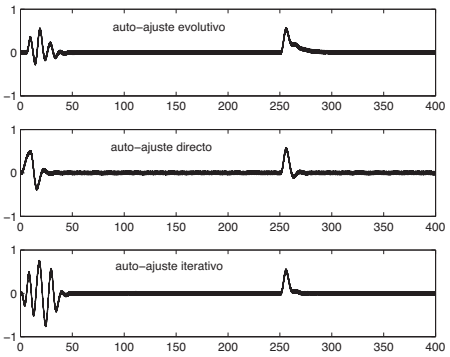

Figura 5: Experimento y respuesta a una entrada escalón unitario en la perturbación conseguida para el modelo G_5.

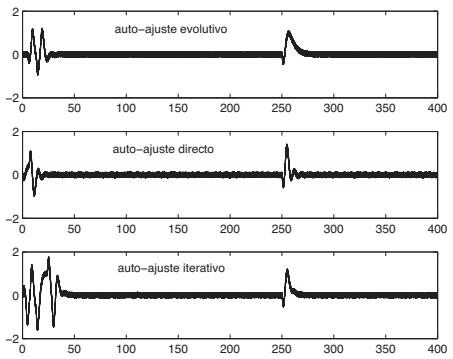

Figura 6: Experimento y respuesta a una entrada escalón unitario en la perturbación conseguida para modelo G_8.

corresponden en general al algoritmo empírico con experimentos de respuesta a una entrada escalón, excepto para el modelo G_3 que tiene una dinámica más lenta que el resto de modelos y en consecuencia su respuesta al escalón tarda en alcanzar el estado estable, como se muestra en la figura 4. Entre los algoritmos evolutivo e iterativo, el primero tiene en general experimentos más cortos ya que sólo identifica el punto de frecuencia y ganancia últimas mediante un experimento de realimentación con relé, mientras que el iterativo identifica varios puntos de la respuesta de frecuencia. Un comportamiento singular se encuentra en el caso del modelo G_3, donde el tiempo de experimento es mayor para el algoritmo evolutivo. Lo anterior es debido a que en el algoritmo evolutivo el tiempo que dura el experimento incluye el tiempo necesario para finalizar la optimización, que para ese modelo en concreto resulta más costosa.

La amplitud de la salida del sistema durante los experimentos es menor en el caso del algoritmo evolutivo, para el cual se obtienen los valores más pequeños del cociente IAE_e^*/T_e^*. Se puede apreciar como para los experimentos de relé, tanto para el algoritmo evolutivo como para el iterativo, las amplitudes de las salida para los diferentes procesos son similares, a pesar de que las ganancias de los mismos son muy dispares. Por el contrario, para los experimentos de respuesta escalón del algoritmo directo, se aprecia una gran dispersión entre los valores máximo que alcanza la salida. Los valores de IAE_e^*/T_e^* en este caso son mayores que los conseguidos en los algoritmos evolutivo e iterativo.

En cuanto al comportamiento total durante la fase de experimento, el algoritmo evolutivo tiene menores valores de I_e que el algoritmo iterativo, debido fundamentalmente a que los valores de amplitud de la salida del primero son inferiores a los alcanzados con el segundo algoritmo. En cuanto al algoritmo directo, los valores de I_e son en general menores que en los otros dos algoritmos, excepto para el modelo G_3 para el que se obtienen valores relativamente muy altos de tiempo de experimento y amplitud de la salida, como ha sido comentado anteriormente.

Respecto a la fase de control, en lo relativo a la robustez, se puede decir que en la mayoría de los casos se consiguen valores razonables de M_s (en un intervalo entre 1.5 y 2.5), excepto en algunos modelos concretos para cada algoritmo. Con el algoritmo directo se obtiene un diseño menos robusto para el modelo G_8, o sea para el sistema de fase no mínima. En el algoritmo iterativo el peor comportamiento de este indicador es para el modelo G_2 que tiene un retardo de tiempo. En el algoritmos evolutivo la menor robustez en el diseño se tiene para los modelos G_3, G_4. Se debe destacar que en los algoritmos de auto-ajuste directo y evolutivo, el valor de M_s es un una especificación de diseño que se ha fijado en $M_s = 2$; la discrepancia entre el valor de diseño y el conseguido en el ajuste se pueden deber a errores de modelado al aproximar la dinámica del sistema mediante un modelo PORT. En el algoritmo iterativo, por el contrario, se usan como especificaciones de diseño el margen de ganancia y de fase y el cálculo de M_s se realiza con fines comparativos entre los tres algoritmos.

El índice IAE_p^* indica cuantas veces es mayor el IAE conseguido con los algoritmos propuestos respecto del IAE que se

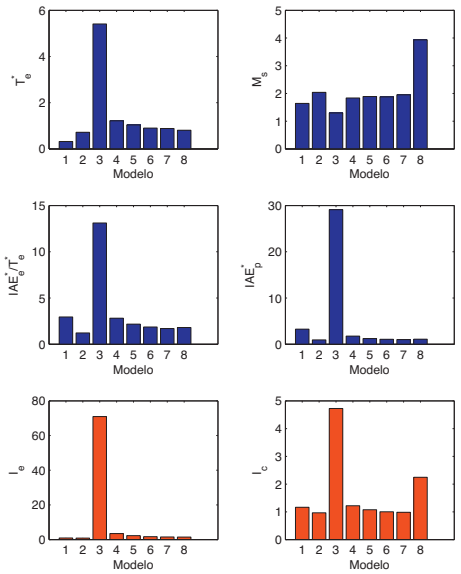

Figura 7: Resultados de los índices obtenidos para con el algoritmo de auto-ajuste directo

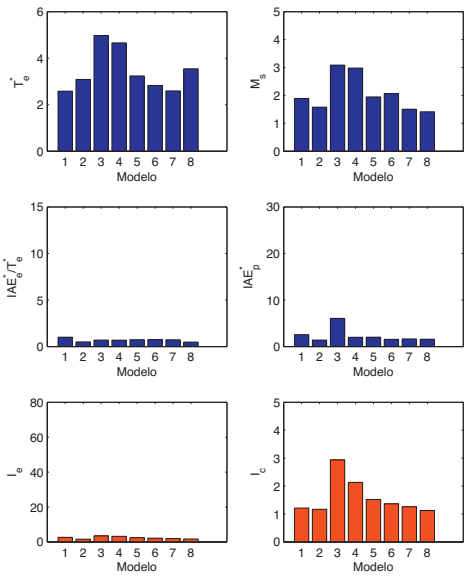

Figura 8: Resultados de los índices obtenidos para con el algoritmo de auto-ajuste evolutivo

obtiene con un ajuste óptimo de PID, que minimiza el IAE con la restricción $M_s = 2$, suponiendo conocido el modelo exacto, según Panagopoulos et al. (2002). Se puede ver que para los tres algoritmos el peor resultado se obtienen con el modelo G_3, sobre todo en el caso del algoritmo directo, en el que el $IAE_p^* \approx 30$. Otro modelo para el cual se obtienen valores de IAE_p^* relativamente altos es el G_1, aunque no de forma tan significativa como para el G_3.

En cuanto al índice total de la fase de control I_c que mide el compromiso entre la robustez, dado por M_s, y el IAE_p^*, se ve que los peores resultados para el algoritmo de auto-ajuste directo se obtienen para los modelos G_3 y G_8, en el caso del algoritmo evolutivo los modelos G_3 y G_4 y para el algoritmo iterativo los modelos G_2 y G_3.

7.3. Comparación de los índices globales

En la Tabla 2 se resumen los valores de los índices globales para las fases de experimento ($I_{e_{total}}$), control ($I_{c_{total}}$) e índice total (I_{total}) obtenidos para cada método según las ecuaciones (34), (35) y (37). En la última columna se muestran los resultados del ajuste por Ziegler and Nichols (1942) a partir los datos de la dinámica del sistema obtenidos mediante un experimento con relé según Åström and Hägglund (1984). Estos datos serán usados como referencia para comparar los tres métodos de auto-ajuste descritos en las secciones anteriores con un método extensamente conocido y aplicado en el ajuste de controladores PID.

Como se pude ver en la Tabla 2, el método iterativo es el que mejor prestaciones consigue en la fase de control, pero al mismo tiempo es el método que mayor coste en la fase de experimentos requiere. El método directo es de los tres el que requiere experimentos menos costosos, sin embargo su comportamiento

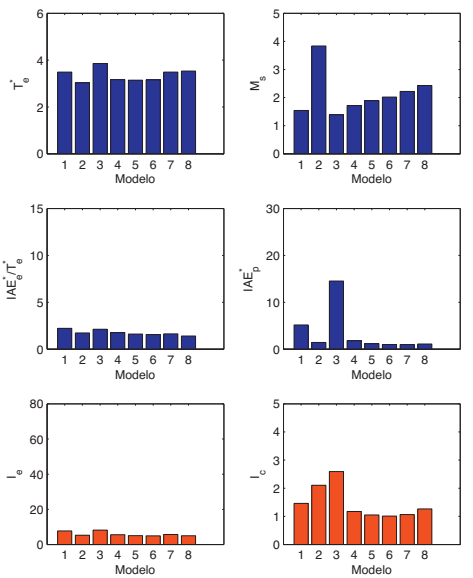

Figura 9: Resultados de los índices obtenidos para con el algoritmo de auto-ajuste iterativo

en la fase de experimento con el modelos G_2 provoca que el $I_{e_{total}}$ sea mayor que para el método evolutivo. A pesar de ello, el método directo presenta el mejor índice global desempeño, o sea, el mejor balance entre coste de experimentos y reducción del efecto de las perturbaciones, según la ecuación (37).

En cuanto a la comparación con el ajuste obtenido por Ziegler and Nichols (1942), se observa que con este último se obtiene una $M_{s_{total}}$ menor, o sea controladores más robustos, y peor compensación de la perturbación, siendo la $IAE^*_{p_{total}}$ mayor que en los otros casos. En cuanto a la fase de experimento, los resultados presentados son para un experimento de relé con amplitud unitaria, y son del orden de magnitud de los valores para los otros métodos de auto-ajuste.

Tabla 2: Resumen de los resultados de los índices obtenidos para cada método de auto-ajuste. En la última columna se muestran los resultados de Ziegler and Nichols (1942) a partir los datos obtenidos mediante un experimento con relé según Åström and Hägglund (1984).

Índices	*Evolutivo*	*Directo*	*Iterativo*	*Z-N*
Fase de Experimento				
$\mathbf{T^*_{e_{total}}}$	3.30	1.00	3.30	1.7
$\mathbf{IAE^*_{e_{total}}/T^*_{e_{total}}}$	0.68	2.50	1.70	2.1
$\mathbf{I_{e_{total}}}$	2.28	2.53	5.81	3.5
Fase de Control				
$\mathbf{IAE^*_{p_{total}}}$	2.10	2.00	2.00	2.3
$\mathbf{M_{s_{total}}}$	2.00	2.00	2.00	1.8
$\mathbf{I_{c_{total}}}$	1.50	1.41	1.38	1.5
Índices totales				
$\mathbf{I_{total}}$	1.72	1.64	1.85	1.85

8. Conclusiones

En este artículo se comparan tres métodos de auto-ajustes de controladores PID, los cuales buscan minimizar el efectos de las perturbaciones en la salida del sistema, siguiendo diferentes estrategias para la obtención de información de la dinámica del sistema y para el cálculo de los parámetros de los controladores. Para la comparación se ha utilizado una metodología que tiene en cuenta el comportamiento del algoritmo al aplicarse a un conjunto de modelos representativos de las dinámicas encontradas en aplicaciones industriales.

El estudio comparativo demuestran que, a pesar de las grandes diferencias entre los tres algoritmos aquí analizados, los ajustes conseguidos con ellos no son significativamente distintos y en todos los casos se obtienen resultados de control aceptables. Se ha comprobado que el algoritmo que más esfuerzo realiza en la fase de experimento para la extracción de la información de la dinámica del proceso consigue un mejor ajuste del controlador, y por tanto, mejores prestaciones en la fase de control. No obstante a eso, algoritmos de auto-ajuste con experimentos simples pueden dar buenos resultados en cuanto a la minimización del efecto de las perturbaciones en la salida, mejorando de forma significativa las prestaciones conseguidas con métodos muy establecidos como el de Ziegler y Nichols.

Los métodos estudiados en este artículo fueron presentados en el concurso anual organizado por el grupo de Ingeniería de Control de CEA-IFAC del curso 2010-2011. La evaluación de los algoritmos considera tanto la fase de experimento como las prestaciones de bucle cerrado conseguidas en la fase de control, resultando en un índice total de evaluación que representa un compromiso entre ambas fases. Teniendo en cuenta esto, la clasificación final del concurso, según los resultados de I_{total} mostrados en la Tabla 2, es la siguiente: primer lugar y ganador del concurso: algoritmo directo; segundo lugar: algoritmo evolutivo; tercer lugar: algoritmo iterativo.

English Summary

Comparative Study of Auto-tuning Algorithms for PID Controllers. Results of the 2010-2011 Match of the Control Engineering Group of CEA.

Abstract

In this paper three PID auto-tuning algorithms which are mainly focused on the disturbance rejection problem are compared. The algorithms differ in both the experiments carried out to obtain information of the process dynamic and the methods for calculating the controller parameters. The algorithms were presented in the 2010-2011 Match of the Control Engineering Group of CEA. The comparison is based on an evaluation methodology that takes into account the experimental phase as well as the closed loop performance during the control phase.

Keywords: PID, auto-tuning, disturbance rejection.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Gobierno Español mediante los proyectos con referencia DPI2008-02133 y DPI2010-15230, la Fundación Caixa Castellón-Bancaixa y la Universitat Jaume I a través de proyecto con referencia P1-1A2010-16, la Universidad Politécnica de Valencia mediante el programa de becas FPI-2010/19. Además, es muy apreciado el apoyo de la Universidad de Costa Rica, el MICIT, y el CONICIT del Gobierno de la República de Costa Rica.

Referencias

Alfaro, V. M., 2006. Low-order models identification from process reaction curve. Ciencia y Tecnología (Costa Rica) 24 (2), 197–216, (in Spanish).

Åström, K., Hägglund, T., 2006. Advanced PID Control. ISA - The Instrumentation, Systems, and Automation Society.

Åström, K., Panagopoulos, H., Hägglund, T., 1998. Design of PI controllers based on non-convex optimization. Automatica 34, 585–601.

Åström, K. J., Hägglund, T., 1984. Automatic tuning of simple regulators with specifications on phase and amplitude margins. Automatica 20, 645–651.

Fleming, P., Purshouse, R., 2002. Evolutionary algorithms in control systems engineering: a survey. Control Engineering Practice 10 (11), 1223 – 1241.

Greg Baker, W., January 2009. Is automated PID tuning dependable? Avilable on-line at `http://www.controleng.com/index.php?id=483&cHash=081010&tx_ttnews[tt_news]=12682`.

Hang, C., Åström, K., Wang, Q., 2002. Relay feedback auto-tuning of process controllers-a tutorial review. Journal of Process Control 12 (1), 143 – 162.

Herreros, A., Baeyens, E., Perán, J. R., 2002. Design of pid-type controllers using multiobjective genetic algorithms. ISA Transactions 41 (4), 457 – 472.

Iruthayarajan, M. W., Baskar, S., 2009. Evolutionary algorithms based design of multivariable pid controller. Expert Systems with applications 36 (5), 9159 – 9167.

Iruthayarajan, M. W., Baskar, S., 2010. Covariance matrix adaptation evolution strategy based design of centralized pid controller. Expert Systems with Applications 37 (8), 5775 – 5781.

Kim, T.-H., Maruta, I., Sugie, T., 2008. Robust PID controller tuning based on the constrained particle swarm optimization. Automatica 44 (4), 1104 – 1110.

Nobakhti, A., Wang, H., 2008. A simple self-adaptive differential evolution algorithm with application on the alstom gasifier. Applied soft computing 8, 350 – 370.

Padula, F., Visioli, A., 2010. Tuning of fractional PID controllers for integral processes. In: Proceedings of FDA'10. The 4th IFAC Workshop Fractional Differentiation and its Applications. Badajoz, Spain.

Padula, F., Visioli, A., 2011. Tuning rules for optimal PID and fractional-order PID controllers. Journal of Process Control 21 (1), 69–81.

Panagopoulos, H., Åström, K., Hägglund, T., 2002. Design of PID controllers based on constrained optimisation. IEE Proceedings Control Theory & Applications 149 (1), 32–40.

Price, K. V., 1999. An introduction to differential evolution. McGraw-Hill Ltd., UK, Maidenhead, UK, England, pp. 79 – 108.

Reynoso-Meza, G., Blasco, X., Sanchis, J., García-Nieto, S., September 2011a. Auto-ajuste evolutivo de controladores PID. In: de Automática, C. E. (Ed.), Memorias de las XXXII Jornadas de Automática.

Reynoso-Meza, G., Sanchis, J., Blasco, X., Herrero, J. M., 2011b. Handling control engineering preferences: How to get the most of pi controllers. In: Proceedings of 16th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA2011), September 5-9, Toulouse-France.

Reynoso-Meza, G., Sanchis, J., Blasco, X., Martínez, M., April 2011c. An empirical study on parameter selection for multiobjective optimization algorithms using differential evolution. In: Proceedings of IEEE Symposium on Differential Evolution (SDE2011).

Romero, J., Sanchis, R., 2011. Benchmark para la evaluación de algoritmos de auto-ajuste de controladores PID. Revista Iberoamericana de Automática de Informática Industrial 8, 112–117.

Romero, J.-A., Sanchis, R., Balaguer, P., 2011. PI and PID auto-tuning procedure based on simplified single parameter optimization. Journal of Process Control 21 (6), 840–851.

Sanchis, R., Romero, J., Balaguer, P., 2010. Tuning of PID controllers based on simplified single parameter optimization. International Journal of Control 83 (9), 1785–1798.

Storn, R., 2008. Differential Evolution Research - Trends and Open Questions. Springer-Verlag, pp. 1 – 31.

Storn, R., Price, K., 1997. Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11, 341 – 359.

Tan, K., Lee, T., Ferdous, R., 2005. Springer, Ch. Automatic PID Controller Tunning-The Nonparametric Approach, pp. 147–182.

Tavakoli, S., Griffin, I., Fleming, P. J., September 2007. Multi-objective optimization approach to the PI tuning problem. In: Proceedings of the IEEE congress on evolutionary computation (CEC2007). pp. 3165 – 3171.

Wang, Q.-G., Hang, C.-C., Zou, B., 1997. Low-order modeling from relay feedback. Industrial & Engineering Chemistry Research 36 (2), 375–381.

Xue, Y., Li, D., Gao, F., 2010. Multi-objective optimization and selection for the PI control of ALSTOM gasifier problem. Control Engineering Practice 18 (1), 67 – 76.

Zhao, S.-Z., Iruthayarajan, M. W., Baskar, S., Suganthan, P., 2011. Multi-objective robust pid controller tuning using two lbests multi-objective particle swarm optimization. Information Sciences 181 (16), 3323–3335.

Ziegler, J., Nichols, N., 1942. Optimum Settings for Automatic Controllers. ASME Transactions, 759–768.