Desarrollo de una herramienta en Matlab para Sintonización de Controladores PID, utilizando algoritmos genéticos basado en técnicas de optimización multiobjetivo

Development of a tool for tuning in matlab PID controllers, using genetic algorithms based on optimization techniques multiobjective

Brayan René Acevedo Jaimes, ingeniero electrónico de la Universidad Francisco de Paula Santander. Investigador adscrito al Grupo de Investigación en Automatización y Control (GIAC). Miembro estudiante ISA (International Society Automation) sección Colombia
payo_aj29@hotmail.com

Juan Camilo Fonseca Galindo, ingeniero electrónico de la Universidad Francisco de Paula Santander. Investigador adscrito al Grupo de Investigación en Automatización y Control (GIAC). Miembro estudiante ISA (International Society Automation) sección Colombia
juankmilofg@gmail.com

July Andrea Gómez Camperos, ingeniera mecatrónica de la Universidad de Pamplona. Candidata a magíster en controles industriales. Dinamizadora de Tecnoparque Colombia Nodo Ocaña. Investigadora adscrita al Grupo de Investigación en Automatización y Control (GIAC).
julyandreagomez@misena.edu.co

Servicio Nacional de Aprendizaje - SENA
Centro de Formación CIES
Tecnoparque Colombia Nodo Ocaña
Regional Norte de Santander

Fecha de recepción: 21 de marzo de 2014
Fecha de aceptación: 16 de mayo de 2014

RESUMEN

Este documento muestra la versatilidad y eficiencia que presenta el desarrollo de una herramienta en Matlab para sintonización de controladores proporcional, integral y derivativo (PID) utilizando algoritmos genéticos (AG) basado en técnicas de optimización multiobjetivo (MOP) fundamentado en frentes de Pareto, calculando de manera óptima las constantes de ganancia proporcional, ganancia integral y ganancia derivativa (KP, KI, KD) para minimización del error, atenuación del sobrepico máximo y reducción del tiempo de establecimiento en una planta determinada. Se compara el desempeño que tiene la implementación de algoritmos genéticos en dar soluciones a múltiples objetivos en controladores PID; con la sintonización de controladores PID existente en Sisotool de Matlab se simularon diferentes sistemas de control de lazo cerrado conformados por una función de transferencia, su controlador y lazo de realimentación. En estos sistemas se analiza el comportamiento que presentan los controladores al aplicarle un *step* a la entrada de la planta.

Con la fabricación de esta herramienta se pretende optimizar la forma de sintonización de controladores PID que en la actualidad se está utilizando; hoy día no existe ninguna herramienta que ayude de forma estructural al proceso de sintonización sin utilizar un nivel complejo de programación y un amplio conocimiento en control; el uso e integración de una serie de técnicas que permitan obtener una herramienta versátil y eficiente; utilizable en la tarea de sintonización de controladores PID y que con la cual se pueda simular y calibrar a través de métodos de inteligencia computacional evolutiva, sin necesidad de tener profundos conocimiento de programación.

Palabras clave: algoritmos genéticos, optimización multiobjetivo, controlador PID, Matlab, sintonización.

ABSTRACT

This document shows the versatility and efficiency by the development of a tool in Matlab for tuning proportional,

integral and derivative controller (PID) using genetic algorithms (GA), based techniques for multiobjective optimization (MOP) based on Pareto fronts, calculating optimally constant proportional gain, integral gain and derivative gain (KP, KI, KD) for error minimization, mitigation of the maximum overshoot and settling time reduction in a given plant. Performance that is implementing genetic algorithms provide solutions to multiple targets in PID controllers, the tuning of existing PID controllers Sisotool Matlab compares different control systems closed loop formed by a transfer function is simulated, the controller and feedback loop. In these systems the behavior that drivers have to pass on a stair at the entrance to the ground is analyzed.

With the realization of this tool is to optimize the shape of tuning PID controllers currently in use, today there is no tool to help structurally the tuning process without using a complex programming level and extensive knowledge in control, the use and integration of a number of techniques that allow a more versatile and efficient tool, usable in the task of tuning PID controllers which can simulate and calibrate by methods of evolutionary computational intelligence without have profound knowledge of programming.

Keywords: Genetic Algorithms, Multiobjective Optimization, PID, MatLab, Controller Tuning.

INTRODUCCIÓN

Los controladores PID en la actualidad son una de las soluciones más prácticas, confiables y sólidas para el control de procesos industriales. A pesar de que la mayoría de controladores PID funcionan moderadamente bien aplicando reglas de sintonía sencillas, existen procesos que necesitan de un ajuste más preciso para garantizar su óptimo desempeño, de modo que una correcta sintonización de los parámetros de ganancia del controlador conlleva un buen funcionamiento de este.

Por consiguiente, en este documento se implementa una herramienta que da solución a un problema de optimización multiobjetivo de controladores PID en función de tres acciones básicas de control: acción proporcional, integral y derivativa, representando el modelo de la planta por una función de transferencia, su controlador y lazo de realimentación, teniendo como funciones objetivo el error, sobrepico máximo y tiempo de establecimiento. Para optimizar los controladores PID y obtener las constantes óptimas de ganancia (KP, KI y KD) del controlador se utilizaron algoritmos genéticos, que son algoritmos basados en las leyes de selección natural. Además, se maneja un problema de optimización multiobjetivo usando frentes de Pareto para encontrar los individuos más calificados que satisfagan las funciones objetivo propuestas.

1. Marco conceptual

Uno de los mayores retos de los seres humanos es poder controlar las diferentes tareas que se realizan en la industria: desde prender y apagar un dispositivo con la ayuda de un botón, hasta controlar aeronaves no tripuladas. Los controladores han generado muchos beneficios para la apropiada operación de procesos o sistemas, por lo cual son de gran interés, tanto en sus funcionalidades como en el mejoramiento continuo del desempeño de procesos industriales (M. Claudia, 2006). Por esta razón, el control resulta ser una atractiva área de trabajo para aplicar los algoritmos genéticos y proponer soluciones factibles.

1.1 Controladores PID

El diseño de controladores se realiza en función del conocimiento del proceso, es decir, a partir del modelo del proceso, del esquema de control y de las restricciones que se le imponen al proceso. A diferencia de ello, la sintonización de los controladores se realiza sin que se disponga de dicha información. Todo controlador PID está gobernado por los valores de ciertas constantes que ponderan las ganancias en el controlador, estas se denominan ganancia proporcional, ganancia

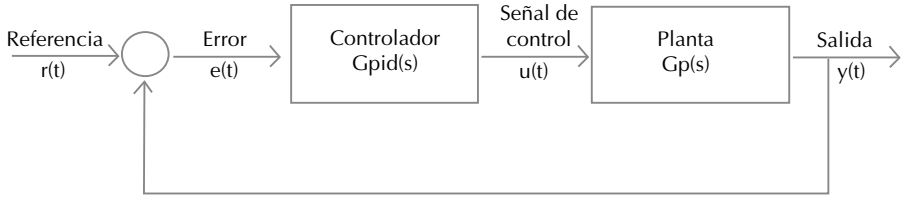
integral y ganancia derivativa (I. A. Ruge y M. A. Alvis, 2009).

El primer término, ganancia proporcional (KP), tendrá el efecto de reducir el tiempo de elevación. Una ganancia integral KI tendrá el efecto de eliminar el error en estado estacionario, pero puede empeorar la respuesta transitoria. Una ganancia derivativa KD tiene el efecto de incrementar la estabilidad al sistema, reducir el sobrepico y mejorar la respuesta transitoria.

En la ecuación 1 se aprecia la expresión matemática que modela un controlador PID en el dominio de Laplace (K. Ogata, 1993). En la Figura 1 se observa el diagrama de bloques de un control PID a una planta. Como entrada al sistema se tiene la referencia r (t) y como salida del sistema se tiene y (t); el controlador en este caso se denomina Gpid(s) y el sistema o planta que se desea controlar se encuentra como Gp(s) (J. Camilo y M. Alejandra, 2013).

$$Gpid(s) = (U(s))/(E(s)) = Kp + Ki/s + Kds \quad (1)$$

Figura 1. Diagrama de bloques para un control PID

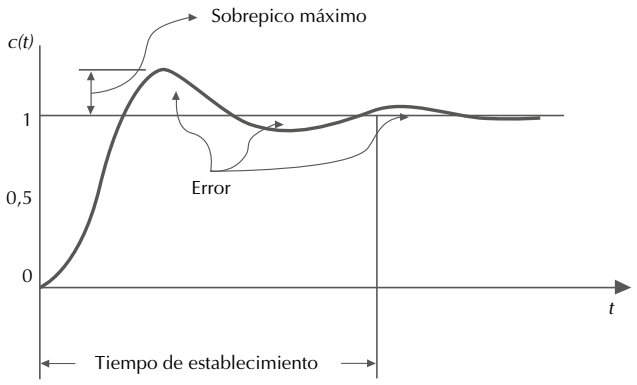


Fuente: J. Camilo y M. Alejandra, 2013.

Los métodos de sintonización en su mayoría están fundamentados en estudios experimentales de la respuesta a una entrada escalón unitario (*step*), razón por la cual los parámetros del controlador obtenidos por esta metodología arrojan respuestas medianamente estables; dichos parámetros se utilizan como un punto de partida para hacer una sintonización mucho más fina y así obtener una respuesta deseada.

Se establecen condiciones de diseño para obtener los valores de ganancias. Las condiciones de diseño más comunes son: error de estado estacionario, sobrepico máximo y tiempo de establecimiento, que es el tiempo necesario para que la respuesta alcance y permanezca dentro de un porcentaje (generalmente del 2 o 5) del error alrededor del valor final. En la Figura 2 se aprecian estas condiciones de diseño (K. Ogata, 1993), (DiStefano, 1992).

Figura 2. Condiciones de diseño en controlador PID



Fuente: autor.

1.2 Algoritmos genéticos

Los algoritmos genéticos fueron introducidos por John H. Holland (John Holland, 1992) a principios de los sesenta y son, por mucho, la técnica evolutiva más popular en la actualidad.

Los algoritmos genéticos resultan ser un tipo de algoritmo matemático altamente paralelo que transforma un conjunto de objetos matemáticos individuales con respecto al tiempo usando operaciones modeladas de acuerdo con el principio darwiniano de reproducción y supervivencia del más apto, y tras haberse presentado de forma natural una serie de operaciones genéticas entre las que destaca la recombinación sexual (John Koza, 1992; G. Toscano, 2001).

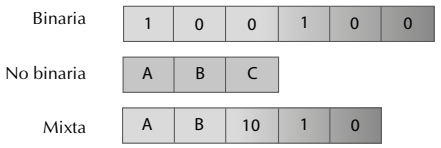
La idea fundamental de los algoritmos genéticos consiste en encontrar una solución aceptable a un problema por medio del mejoramiento de un conjunto de individuos cuya función de evaluación corresponde a una solución del problema. Esta optimización se realiza mediante procesos selectivos y de intercambio de información genética (John Holland, 1992).

Para poder aplicar el algoritmo genético se requieren los siguientes componentes básicos:

- Una representación o codificación de las soluciones potenciales del problema. Existen tres tipos de codificación: binaria, no binaria y mixta, como se observa en la Figura 3. Es necesario mencionar que la codificación binaria es una de las más comúnmente utilizadas; cuando no es la adecuada, se debe a que la codificación del problema se define mejor con otro tipo

de código, en algunos casos la codificación por números (Maneiro, 2002). El tipo de codificación por implementar depende de la naturaleza del problema que se va a solucionar.

Figura 3. Tipos de codificación



Fuente: Muñoz (2005).

- Una forma de crear una población inicial de posibles soluciones, normalmente un proceso aleatorio.

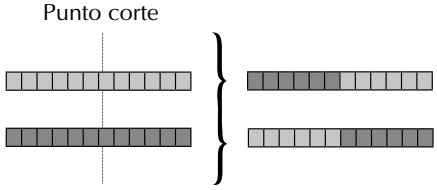
- Una función de evaluación (*fitness*) que juegue el papel del ambiente, clasificando las soluciones en términos de su aptitud; esta función puede determinar qué tan buenos o malos serán los resultados y la convergencia o no del método.

- Un mecanismo de selección que permita elegir a los individuos de acuerdo con su aptitud. En los algoritmos genéticos se puede llevar el proceso de selección de diversas maneras, ya sea determinística o probabilísticamente. Las técnicas de selección pueden ser proporcional (John Holland, 1992), mediante torneo (A. Wetzel, 1983) o de estado uniforme (Darrel Whitley, 1989).

- Operadores genéticos como cruza, mutación y elitismo, que alteren la composición de los hijos que se producirán para las siguientes generaciones (Gunter Rudolph, 1994).

Existen tres tipos principales de cruza. De un punto: se selecciona un punto de manera aleatoria dentro del cromosoma de cada padre y a partir de este se intercambian los materiales genéticos para dar origen a nuevos individuos, como se aprecia en la Figura 4 (John Holland, 1992); de dos puntos: igual a la anterior, excepto que se generan dos puntos de cruza por cada padre (A. de Jong, 1975); uniforme: cruza de n puntos (Lawrence Davis, 1991).

Figura 4. Cruce



Fuente: Muñoz (2005).

El operador de mutación consiste en alterar las características genéticas de un individuo, con el objeto de aumentar la probabilidad de exploración del espacio de búsqueda y disminuir el riesgo de estancamiento del algoritmo en óptimos locales.

1.2.1 Optimización multiobjetivo

El problema de encontrar un vector de variables de decisión es que satisfagan las restricciones y optimice una función vectorial cuyos elementos representen las funciones objetivo. Estas funciones forman una descripción matemática de criterios de desempeño que están usualmente en conflicto entre sí. Por lo tanto, el término optimizar significa encontrar aquella solución que daría un valor aceptable al diseñador en todas las funciones objetivo (A. Osyczka, 1984).

La optimización multiobjetivo maneja los siguientes conceptos:

- **Variables de decisión**, que resultan ser un conjunto de n parámetros cuyos valores dan una solución que puede o no ser válida a un problema de optimización (G. Toscano, 2001).
- **Restricciones**, generalmente en todos los problemas de ingeniería; estas delimitan el problema y validan las soluciones. Por lo tanto, se puede decir que las restricciones dibujan el contorno de la región donde se encuentra el conjunto factible del problema. Las restricciones son funciones de las variables de decisión y pueden ser tanto de igualdad como de desigualdad (G. Toscano, 2001).
- **Funciones objetivo**, las que forman el criterio de evaluación para saber qué tan buena es una solución; al igual que las restricciones, son funciones de las variables de decisión. En la optimización multiobjetivo existen dos o más funciones objetivos (f1 (x), f2 (x),….., fk (x)) en cada problema (G. Toscano, 2001).
- **Problema de optimización objetivo** en estos problemas el objetivo de la optimización es encontrar un vector de decisión o individuo que minimice los resultados evaluados por las funciones objetivos y además cumpla con las restricciones de igualdad o desigualdad impuestas; a esto se le conoce como un conjunto factible en la población (G. Toscano, 2001).
- **Dominancia de Pareto**, para dos vectores de decisión (x*), (y*) ϵ X (G. Toscano, 2001).

$\overline{x^*} < \overline{y^*}$ — $\overline{x^*}$ *domina a* $\overline{y^*}$ — Si $f_i(\overline{x^*}) < f_i(\overline{y^*})$ *para toda* $i = 1, \ldots, k$

$\overline{x^*} \leq \overline{y^*}$ — $\overline{x^*}$ *domina debilmente a* $\overline{y^*}$ — Si $f_i(\overline{x^*}) \leq f_i(\overline{y^*})$ *para toda* $i = 1, \ldots, k$

$\overline{x^*} \sim \overline{y^*}$ — $\overline{x^*}$ *es indiferente a* $\overline{y^*}$ — Si $f_i(\overline{x^*}) \sim f_i(\overline{y^*})$ *para toda* $i = 1, \ldots, k$

Es importante notar que aunque la dominancia se da en el espacio de las variables de decisión, la comparación se da en el resultado de la evaluación de las funciones objetivo, es decir, en el espacio de las funciones objetivo.

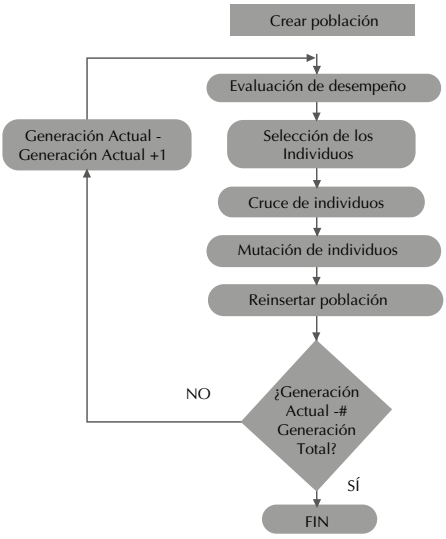
- **Óptimo de Pareto**. Podemos decir que un vector de decisión (x*) que es miembro del conjunto factible es óptimo de Pareto si y solo si no existe otro vector de decisión (y*) que pertenezca al conjunto factible y que lo domine (G. Toscano, 2001).

1.2.2 Estructura general de los AG

Se comienza con una población inicial de soluciones aleatorias (población). Cada individuo en la población es llamado cromosoma, el cual representa una solución al problema. Los cromosomas evolucionan a través de iteraciones sucesivas, llamadas también generación; durante cada generación los nuevos cromosomas son evaluados, usando la medida de aptitud. Luego, mediante los operadores de cruce y mutación, se seleccionan los cromosomas con mayor probabilidad de supervivencia, a fin de realizar la exploración y la explotación de las funciones objetivo. Por último, se reinsertan a la población actual; después de terminar el número de generaciones se selecciona al individuo con mayor desempeño, el cual representa la solución óptima (I. A. Ruge y M. A. Alvis, 2009).

En la Figura 5 se observa el diagrama básico de un algoritmo genético.

Figura 5. Diagrama de flujo de algoritmo genético simple



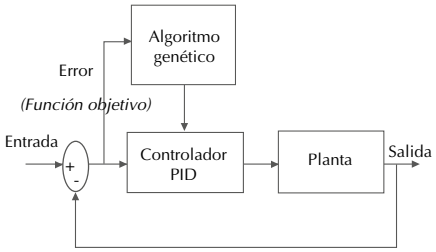
Fuente: autor.

2.3 Aplicación de algoritmos genéticos en controladores PID

En su mayoría, los algoritmos evolutivos han sido empleados para ajustar los parámetros de controladores PID, con el propósito de mejorar el rendimiento de controladores y así obtener un diseño multidisciplinar que involucre tanto al diseño de control como al diseño mecánico y al electrónico. A continuación se muestra, en la

Figura 6, el diagrama de bloques que se tiene para aplicar los algoritmos genéticos al controlador PID.

Figura 6. Algoritmos genéticos aplicados a controladores PID



Fuente: I. A. Ruge y M. A. Alvis, 2009.

2. Metodología e implementación

El algoritmo que se aplicó fue desarrollado e implementado en Matlab 2013 utilizado en un computador portátil Lenovo Z460 con procesador Intel® corre™ i5 M460 a 2,53 GHz con 8 GB de memoria RAM, Windows 8, 64 bits. Se desarrolló una herramienta en Matlab en donde se tomaron como funciones objetivos el error, el sobrepico máximo y como restricción el tiempo de establecimiento, de manera que el problema de optimización multiobjetivo resuelto se define así:

$$f1(Kp, Ki, Kd) = \int_0^T [e(t)]^2 = Error$$

$$f2(Kp, Ki, Kd) = Sobrepico\ Máximo$$

$$R(Kp, Ki, Kd) = Tiempo\ de\ establecimiento$$

Luego del proceso de optimización, al evaluar cada función objetivo en cada iteración la herramienta selecciona y reordena los valores óptimos de las constantes KP, KI y KD para todas las soluciones obtenidas en cada aumento de generación, de modo que al terminar el número máximo de generaciones tendremos los mejores resultados de las constantes a lo largo de todas las generaciones evaluadas.

Los algoritmos 1 y 2 muestran el pseudocódigo del proceso de optimización multiobjetivo y de la obtención de los valores provenientes de las funciones objetivo, respectivamente.

Algoritmo 1. Pseudocódigo general

1. Número de individuos en la población (tamaño de la población).
2. Número máximo de iteraciones (número de generaciones).
3. Establecer dimensiones del espacio de búsqueda.
4. Inicializar función de transferencia (numerador y denominador).
5. Definir función para obtener el valor de las funciones objetivo (algoritmo 2).
6. Aplicar algoritmo de optimización usando algoritmos genéticos y optimización multiobjetivo con base en frentes de Pareto (*script* en Matlab).

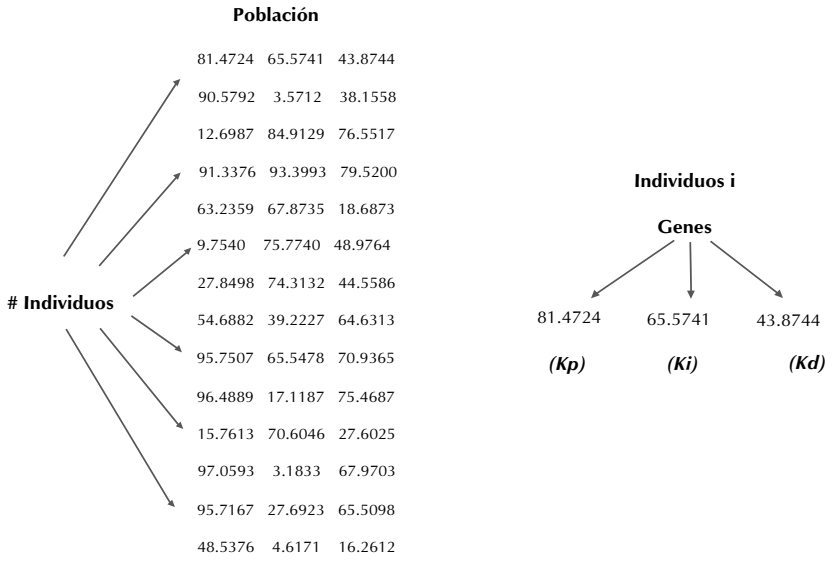
7. Obtener los mejores resultados.

Algoritmo 2. Pseudocódigo para obtener funciones objetivo

1. Obtener los parámetros de la función de transferencia.
2. Obtener las ganancias para evaluar (KP, KI y KD).
3. Definir la función de transferencia resultante de multiplicar el controlador PID (ecuación 1) y la función de transferencia de la planta.
4. Hacer el lazo de retroalimentación (*feedback*).
5. Obtener la respuesta al paso (*step*) de la función de transferencia resultante del *feedback*.
6. Obtener los valores de error, sobrepico máximo y tiempo de establecimiento con la respuesta al paso unitario.
7. Ordenar del óptimo al menos bueno los valores obtenidos por la evaluación.

La población inicial opera con una codificación de tipo numérico real, dando como ventaja la omisión de procesos de codificación y decodificación, que son tenidos en cuenta para pasar a los individuos de la población al espacio de evaluación definido por la función objetivo, trabajando así con los coeficientes de las constantes en forma directa. En la Figura 7 se muestra el formato de la población inicial.

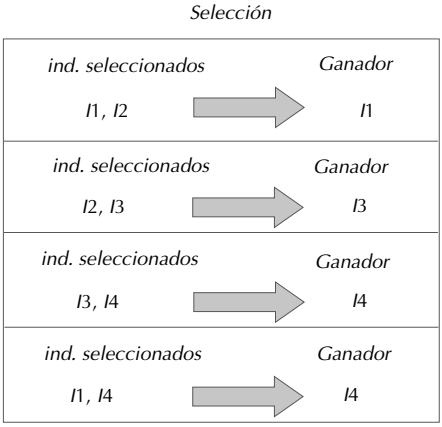
Figura 7. Población inicial



Fuente: autor.

En este algoritmo se usó selección por torneo, utilizando dominancia como método de comparación; la idea básica de este tipo de selección consiste en elegir con base en comparaciones directas de los individuos de la población. Se escoge un número de individuos para someterlos a competición –generalmente compiten de a dos individuos–, después comparamos con base en su aptitud, de modo que el ganador es el individuo más apto de la competencia; así se garantiza que el mejor individuo será seleccionado un mayor número de veces P que el resto de individuos. En la Figura 8 Se observa el proceso de selección utilizado en la herramienta.

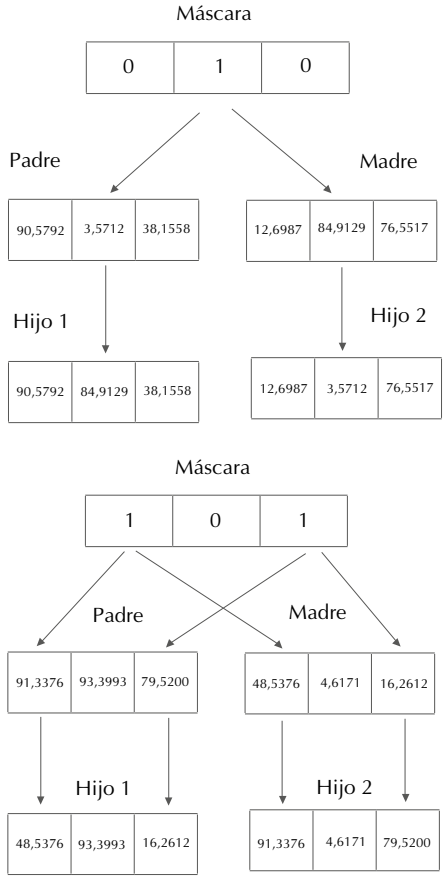
Figura 8. Selección



Fuente: autor.

El operador de cruce trabaja sobre dos individuos, denominados individuos progenitores o padres. Este operador produce generalmente otros dos individuos, que serán nuevos puntos en el espacio de búsqueda del algoritmo. El operador de cruce implementado en el algoritmo es un cruce de tipo uniforme, y se basa en el uso de una máscara llamada de cruce; la obtención de esta es de forma aleatoria, dando la misma probabilidad a que en cada una de las posiciones de la máscara haya un 1 o un 0. En la Figura 9 se observan los dos tipos de cruce –de un punto (izquierda) y en dos puntos (derecha)–.

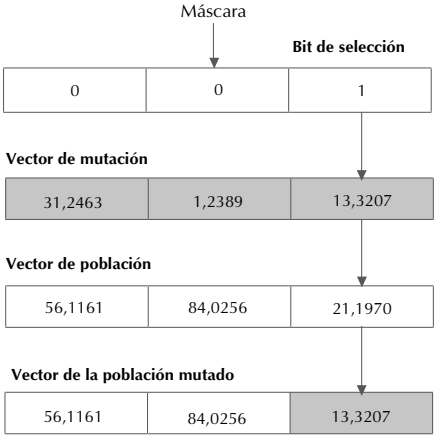
Figura 9. Cruce uniforme



Fuente: autor.

Además, se usó una mutación de tipo uniforme, la cual nos garantiza un porcentaje de mutación constante a lo largo de todo el proceso evolutivo; en la mutación se habla de un progenitor y un descendiente, además de una máscara aleatoria de tipo binario, encargada de seleccionar cuál será el bit del gen correspondiente por mutar. De este modo, el operador de mutación ayuda a renovar la población, y lo hace de forma arbitraria, pues el resultado de la descendencia nada tiene que ver con su progenitor. En la Figura 10 se muestra el proceso de mutación implementado.

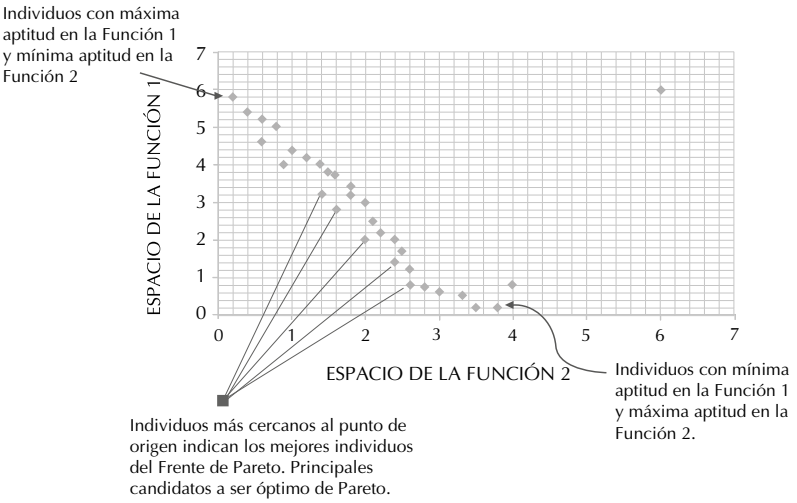
Figura 10. Mutación



Fuente: autor.

Después de haber hecho mutación, se busca obtener un conjunto de individuos aptos para la solución del problema; de este modo la población se somete a un proceso de optimización multiobjetivo basado en frentes de Pareto. El proceso de optimización multiobjetivo se enfoca en la búsqueda de individuos que satisfagan las restricciones impuestas y al mismo tiempo sobresalgan en la evaluación de las funciones objetivo. Estos criterios de desempeño por lo general están en conflicto entre sí. En la Figura 11 se muestra una gráfica de cómo se encuentran distribuidos los individuos en el frente de Pareto.

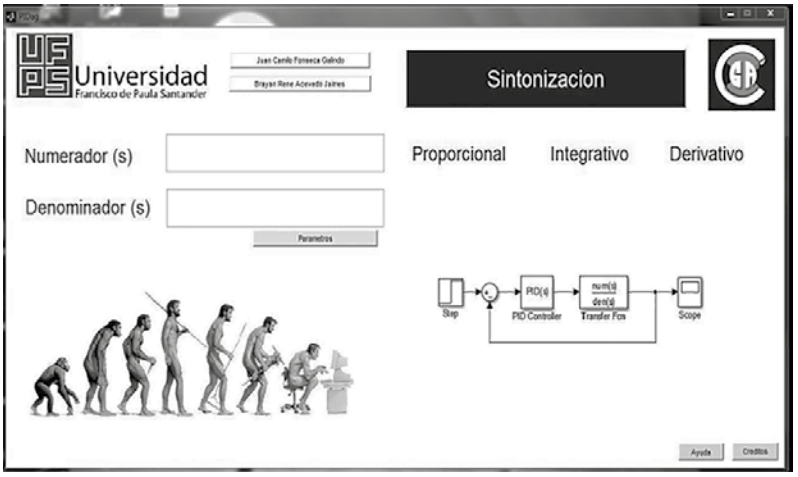
Figura 11. Frente de Pareto



Fuente: autor.

La herramienta cuenta con una interfaz GUIDE (editor de interfaces de usuario GUI) para interactuar con el usuario, cambiar los parámetros de operación del algoritmo y mostrar los resultados obtenidos por la herramienta desarrollada. La interfaz tiene los siguientes elementos, que se observan en la Figura 12:

Figura 12. PIDag



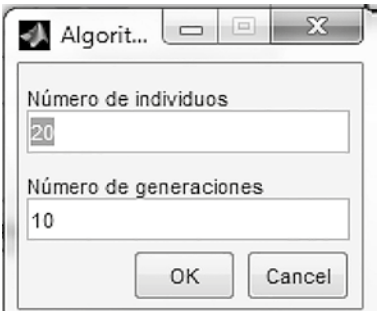
Fuente: autor.

La herramienta posee botones. Uno es utilizado para iniciar la sintonización de la planta; dos despliegan información de los diseñadores de la herramienta de sintonización; otro es usado para ingresar al menú de parámetros de la herramienta en donde se modifican parámetros de operación del algoritmo. Existe un botón de ayuda para orientar al usuario sobre cómo emplear correctamente la herramienta.

Existen dos cuadros de texto editables en donde se ingresan los coeficientes del modelo de la planta en función de Laplace, el numerador y denominador, respectivamente. En el botón parámetros se despliegan tres submenús, que se explican a continuación:

En el de algoritmo genético se despliega una ventana donde se pueden modificar el número de individuos que evaluará el algoritmo durante el proceso y el número de generaciones en donde se especifica el número de generaciones que el algoritmo se ejecutará. Este submenú se aprecia en la Figura 13.

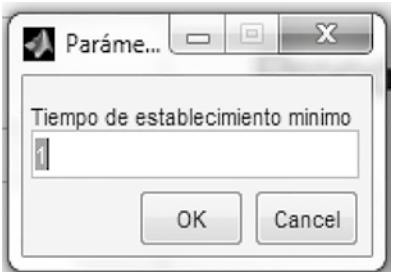
Figura 13. Algoritmo genético



Fuente: autor.

En el submenú parámetros de salida se despliega una ventana donde se define el tiempo de establecimiento mínimo con el cual el algoritmo va a restringir los individuos de la población; es el tiempo mínimo en el cual se desea que la planta se estabilice. Este submenú se aprecia en la Figura 14.

Figura 14. Parámetros de salida



Fuente: autor.

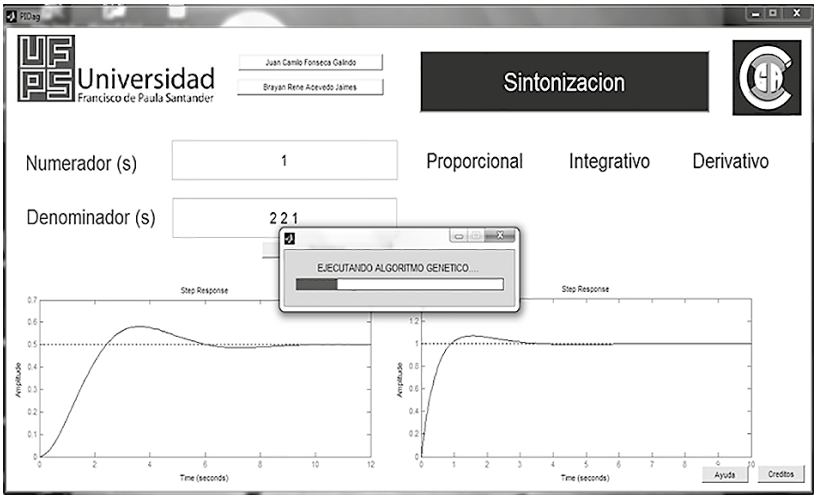
Figura 15. Límites de constantes

Fuente: autor.

En el momento de ejecutar la herramienta se deben ajustar los parámetros de búsqueda del AG, entre los cuales están los límites superiores e inferiores de cada una de las constantes. Estos límites garantizan que las constantes del PID obtenido no estén por fuera del rango establecido. Los rangos de operación pueden ser modificados a conveniencia, con el fin de garantizar una mejor exploración del espacio de búsqueda o delimitar los espacios de trabajos en donde se requiera que las constantes (Kp, Ki, Kd) puedan operar.

Al ejecutarse la sintonización se aprecia una barra que indica el avance del algoritmo genético. Al terminarse la ejecución de la simulación se muestran las respectivas constantes, la gráfica de la respuesta de la planta en lazo abierto y la planta sintonizada con su respectivo controlador, como se observa en la siguiente figura.

Figura 16. PIDag en ejecución



Fuente: autor.

3. Simulación y resultados

Para todas las pruebas desarrolladas se utilizó una población de 30 individuos y 20 iteraciones, que corresponden en cada caso a 600 evaluaciones de cada una de las funciones objetivo. Para el algoritmo de optimización, la población inicial generada es de tipo aleatorio y se operó con un espacio de búsqueda de $0 \leq KP \leq 50$, $0 \leq KI \leq 20$ y $0 \leq KD \leq 20$.

Se realizaron pruebas de la herramienta con plantas extraídas del libro de K. Ogata *Ingeniería de*

control moderna. En cada una de ellas se obtuvieron las gráficas de la planta en lazo cerrado sin controlador, y gráficas de la planta en lazo cerrado con el controlador obtenido de la herramienta PIDag, y se compararon resultados con la herramienta Sisotool de Matlab.

3.1 Función de prueba 1

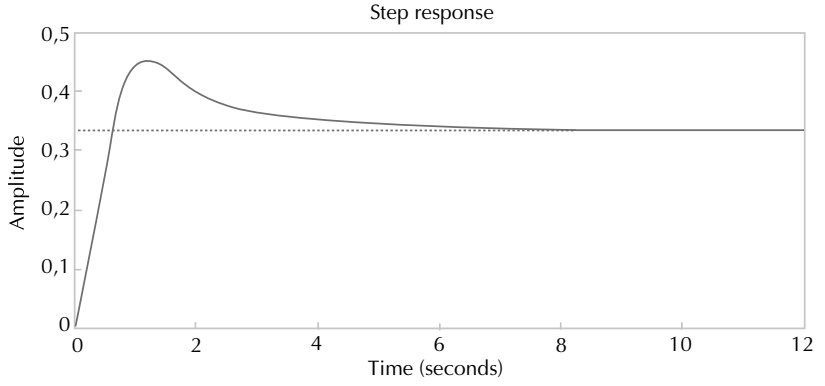
$$\frac{5s + 2}{s^3 + 5s^2 + 8s + 4}$$

Proporcional	Integrativo	Derivativo
37,6692	18,9251	13,0447

Tabla 1. Resultados para la función de prueba

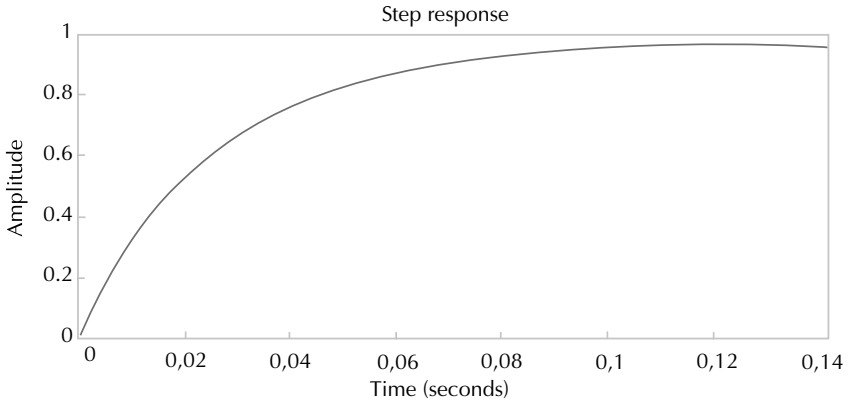
	Sobrepico máximo	Tiempo establecimiento	Error
Planta	0,4506	5,7288	0,67
Algoritmo G	0,9995	1,14	2,33e-7
Sisotool	1,22	2,5	0,05

Figura 17. Respuesta de la planta 1 sin controlador ante un escalón unitario



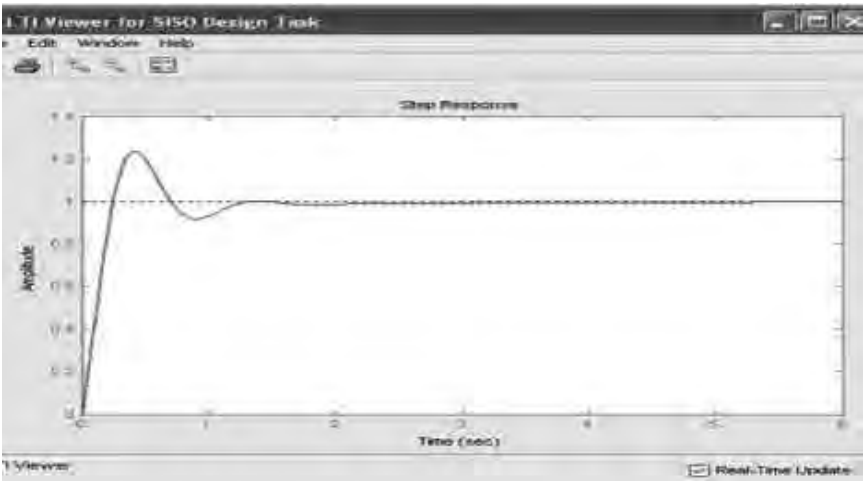
Fuente: autor.

Figura 18. Respuesta de la planta 1 con controlador sintonizado por PIDag ante un escalón unitario



Fuente: autor.

Figura 19. Respuesta de la planta 1 con controlador sintonizado por Sisotool ante un escalón unitario



Fuente: autor.

3.2 Función de prueba 2

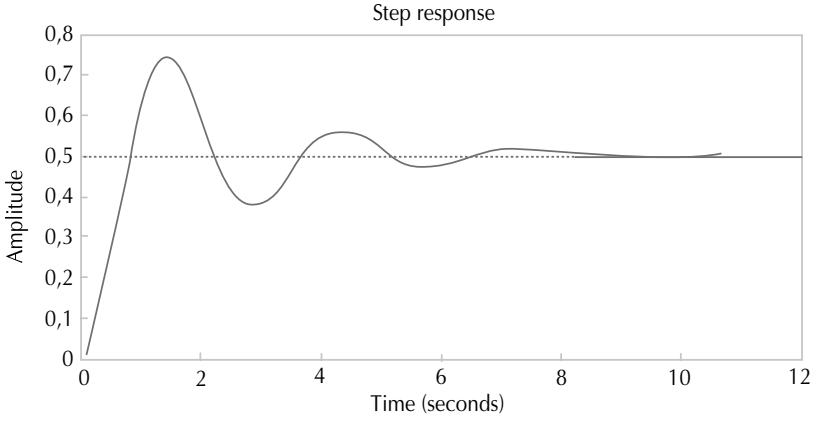
$$\frac{10}{s^2 + 2s + 10}$$

Proporcional	Integrativo	Derivativo
8,4047	17,1467	12,0918

Tabla 2. Resultados para la función de prueba

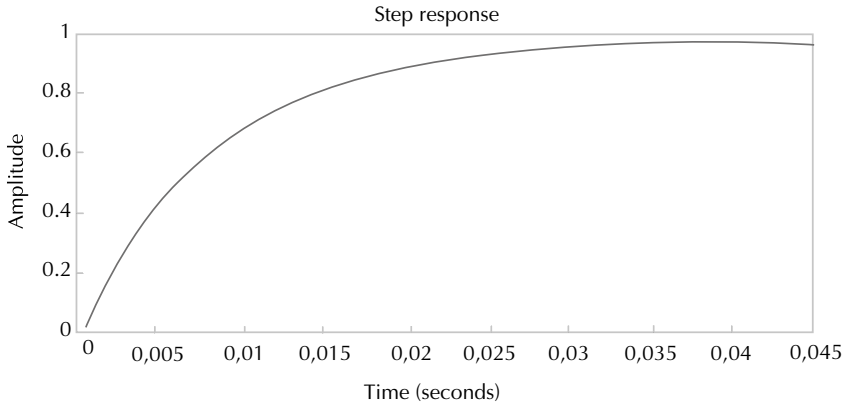
	Sobrepico máximo	Tiempo establecimiento	Error
Planta	0,7421	3,7795	0,5
Algoritmo G	1,0137	1,9484	3,013e-7
Sisotool	1,22	2,5	0,05

Figura 20. Respuesta de la planta 2 sin controlador ante un escalón unitario



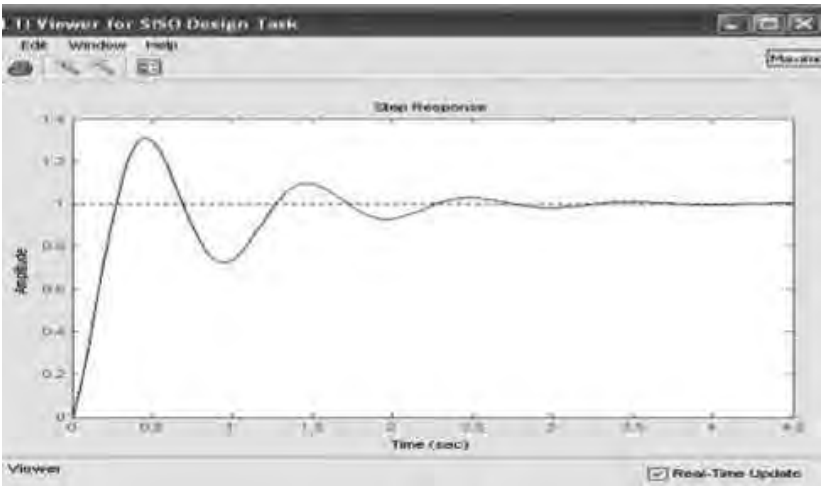
Fuente: autor.

Figura 21. Respuesta de la planta 2 con controlador sintonizado por PIDag ante un escalón unitario



Fuente: autor.

Figura 22. Respuesta de la planta 2 con controlador sintonizado por Sisotool ante un escalón unitario



Fuente: autor.

4. Conclusiones

Se analizó la sintonización de los controladores correspondientes a cada una de las plantas simuladas tomando como criterios de evaluación el error, el sobrepico máximo y tiempo de establecimiento obtenidos en la señal de salida como resultados de la aplicación de un escalón unitario (*step*) a la entrada de cada una de las plantas, cuando se utiliza la herramienta PIDag, con el fin de minimizar la magnitud del error, reducir el sobrepico máximo y obtener tiempos de establecimientos más cortos. Las tablas comparativas y las gráficas de las señales tomadas a la salida de la planta mostraron que la herramienta PIDag, en comparación con el Sisotool de Matlab, obtiene mejores resultados en los tres criterios de evaluación tomados para la sintonización de controladores PID.

Se demostró que la aplicación de un método que se sustenta en principios de selección natural para la optimización de los valores de ganancia de un controlador PID permite dar a conocer una nueva forma de solución no convencional a problemas de control, además de encontrar una variedad de valores óptimos para estos parámetros, dando como opción al programador de elegir el que más se acomode a sus requerimientos de diseño.

Se evidenció que la eficacia de la implementación de algoritmos genéticos aplicados a la sintonización de controladores PID radica en la capacidad de exploración activa que tenga el algoritmo, la manera en que se eligen los individuos de la población, tener una cuidadosa selección de los individuos por recombinar y mutar, además de una buena interpretación de las funciones objetivos.

Al desarrollar una herramienta de sintonización de controladores PID usando algoritmos genéticos en donde se toma más de un criterio de evaluación es imperativo y necesario implementar un proceso de optimización multiobjetivo fundamentado en frentes de Pareto, en donde se demuestra que la asignación de restricciones y funciones objetivo permite garantizar una mejor exploración del espacio de búsqueda y escoger individuos óptimos, que satisfagan las restricciones impuestas y al mismo tiempo sobresalgan en la evaluación de las funciones objetivo en un espacio de búsqueda global sin que el algoritmo converja a un óptimo local.

REFERENCIAS

Castro P. J. Camilo and Guzmán P., M. Alejandra (2013). Optimización multiobjetivo de un controlador PID. *Aplicando Algoritmos Bioinspirados, CIIMA*, 2 (20), 191-197.

De Jong. A. K. (1975). *An Analysis of the Behavior of a Class of Genetic Adaptative Systems*. PhD thesis. Michigan: University of Michigan.

Deb, K. *et al.,* (2002). A fast an elitist multiobjective genetic algorithm: NSGA *II. IEEE Transaction on Evolutionary Computation*, vol. 6(2).

DiStefano, Stubberud and Williams (1992). *Retroalimentación y sistemas de control*. Segunda Edición (Trad. R. Gómez Cruz) Bogotá: Ed. McGraw-Hill.

Gunter, Rudolph (1994).Convergence analysis of canonical genetic algorithms. January IEEE Transactions on Neural Networks 5(1), 96-101.

Holland, John H. (1992). *Adaptation in natural artificial system*. Second edition. Cambridge, Massachusetts: The MIT Press.

Koza, John R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, Massachusetts: The MIT Press.

Lawrence, Davis. (1991). *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold.

Maneiro Malavé, Ninoska (febrero 2002). *Algoritmos genéticos aplicados al problema cuadrático de asignación de facilidades.* Departamento de Investigación Operativa, Escuela de Ingeniería Industrial, Universidad de Carabobo, Valencia, Venezuela.

Muñoz M., A. F. (2003). *Tecnologías de control inteligente: Redes neuronales, algoritmos genéticos y de clonación artificial*. ISSN 1692-72577, vol. 1, No. 1, 4-9.

Muñoz M., A. F. Wseas, M. (July 2005) Advance control of applied artificial cloning paper Wseas, transactions on systems. *Advance Artificial Cloning Applied to Industrial Process Control* vol. 4 Issue 7, 930-938, Atenas, Grecia.

Ogata, K. (1993). *Ingeniería de control moderna*. Segunda edición. (Trad. B. A. Fabian-Kernel). México: Ed. Prentice-Hall.

Osyczka, A. (1984). *Multicreterion Optimization in Engineering with Fortran programs*. Ellis Horwood Limited.

Pinto F., M. Claudia (mayo 2006). *Sintonización de controladores PID utilizando algoritmos evolutivos*. Universidad de Pamplona, Facultad de Ingenierías y Arquitectura, Pamplona, Colombia.

Ruge, I. A. and Alvis, M. A. (2009). Aplicación de los algoritmos genéticos para el diseño de un controlador PID adaptativo. *Tecnura*, vol. 13(23), 82-90.

Syswerda, Gilbert (1989). Uniform Crossover in Genetic Algorithms. In J. David Schaffer (ed.) *Proceedings of the Third International Conference on Genetic Algorithms*. San Mateo, California: Morgan Kaufmann, 2-9.

Toscano, P. G. (septiembre 2001). *Optimización multiobjetivo usando un microalgoritmo genético*. Universidad Veracruzana-Lania.

Van Rensburg, P. J., Shaw, L. S. y Van Wyk J. D. (April 1998). *Adaptive PID-control using a Genetic Algorithm. 1998 Second International Conference on Knowledge-Based Intelligent Electronic Systems*. Australia: Editors, L. C. Jain and R. K. Jain Adelaide, 21-23.

Wetzel. A. (1983). *Evaluation of Effectiveness of genetic algorithms in combinational optimization*. Pittsburgh (unpublished): University of Pittsburgh.

Whitley. Darrel (1989). The Genitor Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best. *In Proceedings of the Third International Conference on Genetic Algorithms*. San Mateo, California: Morgan Kaufmann Publishers, 116-121.