Algoritmo PSO para identificación de parámetros en un motor DC

System identification of a DC motor using PSO algorithm

César Duarte[1], Jabid Quiroga[2]*

[1]Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones Universidad Industrial de Santander, Cra. 27 Calle 9. Ciudad Universitaria, Bucaramanga, Santander, Colombia.

[2]Escuela de Ingeniería Mecánica. Universidad Industrial de Santander, Cra. 27 Calle 9. Ciudad Universitaria, Bucaramanga, Santander, Colombia.

(Recibido el 13 de junio de 2009. Aceptado el 6 de abril de 2010)

Resumen

En este artículo se presenta la aplicación del algoritmo *Particle Swarm Optimization* (PSO) como estrategia de búsqueda para determinar los parámetros de un sistema tipo caja gris. Este proceso de identificación se ilustra utilizando la respuesta a un escalón de un motor DC en lazo abierto. Para agilizar la identificación del sistema a través del uso de PSO, se limita el espacio de búsqueda usando la información que puede extraerse de la respuesta en el dominio del tiempo del sistema a identificar. Los resultados del proceso de identificación de los parámetros del motor se obtienen utilizando la herramienta *PSOt* disponible para entorno Matlab® desarrollada por Brian Birge.

----- *Palabras clave*: Identificación de sistemas, particle swarm optimization, optimización inteligente

Abstract

In this paper an application of *Particle Swarm Optimization* (PSO) algorithm is presented as strategy for parameters determination of a grey box system. This system identification process is implemented based on the open loop step response for a DC motor. The vector space is bounded via step time response to speed up the parameter identification. The PSO algorithm is implemented and validated using the PSOt Matlab® toolbox developed by Brian Birge.

----- *Keywords*: System Identification, particle swarm optimization, intelligent optimization

* Autor de correspondencia: telefax 57 + 7 + 634 63 76, correo electrónico: jabib@uis.edu.co (J. Quiroga)

Introducción

Los modelos matemáticos de los sistemas cumplen un rol esencial en aplicaciones como simulación, predicción, control y diagnóstico en campos tan variados como la ingeniería, la economía, la medicina, y la fisiología, entre otros. Es teóricamente posible obtener modelos matemáticos de un sistema usando los principios de la física, la química, la biología, etc. Sin embargo, en muchos sistemas la composición de éstos ocasiona que la derivación del modelo sea una tarea compleja. En estas circunstancias el modelo matemático del sistema puede ser construido a partir de datos experimentales. El proceso de desarrollo de modelos a partir de datos experimentales es llamado identificación de sistemas. Este proceso consiste en el desarrollo y análisis de métodos para obtener modelos de cajas grises y cajas negras de sistemas.

Actualmente, muchas investigaciones en al área de identificación de sistemas se han orientado hacia el uso de algoritmos de procesamiento paralelo como son los Algoritmos Genéticos (AG) y *Particle Swarm Optimization* [1-3]. El uso de estas técnicas se prefiere en procesos de modelamiento complejos en donde la función objetivo del proceso de identificación del sistema puede quedar atrapada en mínimos locales, lo cual dificulta el proceso de convergencia hacia la solución óptima. Por otro lado, las anteriores técnicas aseguran convergencia y la obtención al menos del mejor mínimo local [2]. En este artículo se propone el uso de PSO en la identificación de los parámetros de un modelo de caja gris para un motor DC a partir de la respuesta al escalón. Aunque el comportamiento del sistema es de naturaleza lineal, la función objetivo que se utiliza en el proceso de búsqueda de la solución óptima para determinar los parámetros es de tipo no-lineal y será la función a optimizar a través del algoritmo de PSO.

Marco teórico

Particle swarm optimization y algoritmos de optimización inteligentes

El algoritmo PSO pertenece a las técnicas denominadas optimización inteligente y se clasifica como un algoritmo estocástico de optimización basado en población. A esta clasificación igualmente pertenecen los Algoritmos Genéticos (AG). Los anteriores algoritmos se consideran adecuados comparados con los métodos clásicos de optimización cuando el problema de optimización es complejo, estocástico o no lineal con múltiples mínimos locales.

Las ventajas atribuidas a las técnicas inteligentes de optimización son su paralelismo intrínseco, su capacidad para resolver problemas complejos, de gran tamaño, y con un mínimo conocimiento del sistema que se está identificando. Estas ventajas se discuten a continuación [4].

PSO es intrínsicamente paralelo. La mayoría de algoritmos clásicos operan secuencialmente y pueden explorar el espacio de solución solamente en una dirección a la vez. Esta diferencia entre algoritmos clásicos y algoritmos inteligentes se ilustra en la figura 1. En esta figura se puede observar que en el evento en que la solución determinada no es la óptima, el algoritmo clásico la desecha e inicia la búsqueda en otra dirección. Por otra parte, el algoritmo de optimización inteligente puede explorar el espacio de la solución en múltiples direcciones simultáneamente. Si un camino no conduce al óptimo, el algoritmo fácilmente elimina tal camino y continúa buscando otros mejores. Esta forma de operación proporciona una mayor probabilidad de encontrar la solución óptima. Debido al paralelismo, los algoritmos inteligentes de optimización son adecuados para problemas con gran número de variables, donde el espacio de la solución es de gran tamaño para realizar una búsqueda exhaustiva en un tiempo razonable.

Otra fortaleza de los algoritmos de optimización inteligente es su buen desempeño en problemas cuyo espacio de solución presenta múltiples mínimos locales. Muchos algoritmos clásicos de búsqueda como *backpropagation,* usado para entrenar redes neuronales, pueden quedar atrapados en mínimos locales. Los algoritmos de optimización inteligente han demostrado ser efectivos evitando mínimos locales e identificando

el mínimo global en un espacio de búsqueda complejo. Debe notarse que en ocasiones, no hay manera de determinar si una solución es un mínimo global o solamente un muy buen mínimo local. Sin embargo, aún cuando el algoritmo de optimización inteligente no puede encontrar el mínimo global, usualmente encuentra un buen mínimo local.

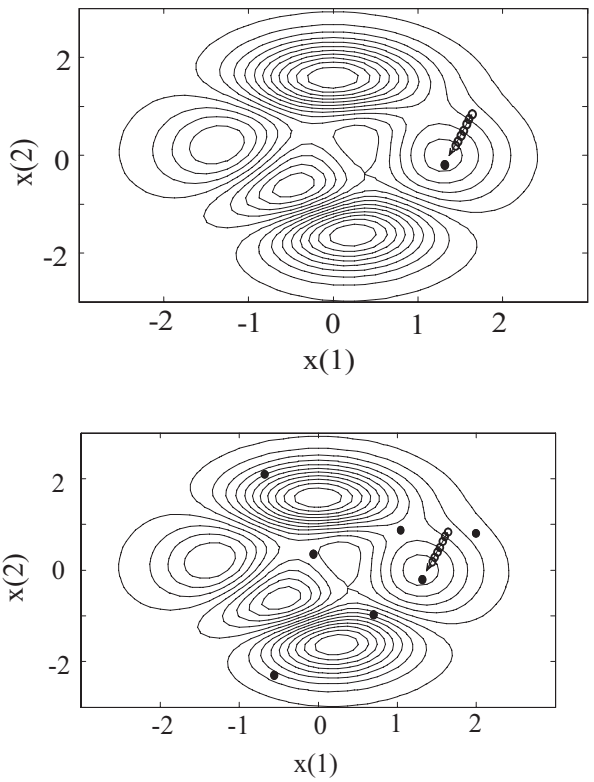

Figura 1 Algoritmos secuenciales vs. Algoritmos inteligentes

Finalmente, los algoritmos de optimización inteligentes no requieren de un conocimiento detallado de la estructura y comportamiento del sistema como otros algoritmos de optimización la requieren. En lugar de usar información específica del sistema para guiar la búsqueda, este algoritmo hace cambios aleatorios en las soluciones candidatas y usa una función de ajuste multivariable para determinar si las soluciones son óptimas o no. Como un nuevo desarrollo de los algoritmos de optimización inteligentes, *Particle Swarm Optimization* es simple en concepto y altamente eficiente desde el punto computacional.

Particle swarm optimization

PSO es un algoritmo estocástico de búsqueda basado en población, propuesto por *Kennedy* y *Eberhart* en 1995 [5] como un modelo de las actividades sociales de insectos, pájaros y peces. Este algoritmo pretende representar el proceso natural de comunicación grupal para compartir conocimiento individual cuando grupos de animales se desplazan, migran o cazan. Si un miembro detecta un camino deseable para desplazarse, el resto de la colonia lo sigue inmediatamente. En PSO, este comportamiento animal es imitado por partículas con ciertas posiciones y velocidades en un espacio de búsqueda, donde la población es llamada *swarm*, y cada miembro del *swarm* es llamado partícula. La población inicial se determina aleatoriamente y cada partícula se desplaza a través del espacio de búsqueda y recuerda la mejor posición que ha encontrado. Cada partícula comunica las buenas posiciones a las demás y dinámicamente ajustan su propia posición y su velocidad con base en las buenas posiciones. La velocidad se ajusta con el comportamiento histórico de las partículas. De esta forma, las partículas tienden a dirigirse hacia un mejor espacio de búsqueda en el proceso de minimización de la función objetivo [5]. Este procedimiento de búsqueda se describe por (1).

$$v_i^{k+1} = w.v_i^k + c_1.rand_1. \left(pbest_i - x_i^k\right) + c_2.rand_2.\left(gbest - x_i^k\right) \qquad (1)$$
$$x_i^{k+1} = x_i^k + v_i^{k+1}$$

En (1) c_1 y c_2 son constantes positivas, definidas como coeficientes de aceleración; w es el factor inercial; $rand_1$ y $rand_2$ son dos números aleatorios (con distribución de probabilidad uniforme) en el rango [0,1]; x_i representa la i^{esima} partícula y $pbest_i$ la mejor posición previa de x_i; *gbest* es la posición de la mejor partícula de toda la población; y v_i es la razón de cambio de la posición (velocidad) de la partícula x_i. Los cambios de velocidad en (1) se

componen de tres partes: *momentum*, *cognitiva* y *social*. De esta forma se obtiene una velocidad que tiende a acercar la partícula a *pbest* y *gbest*.

De acuerdo con (1), el factor inercial w y los dos coeficientes de aceleración c_1 y c_2 afectan el desempeño del algoritmo PSO. Desde el planteamiento de este algoritmo en 1995, una gran cantidad de trabajos se han desarrollado para mejorar la versión original modificando estos tres factores. La variación de estos factores puede balancear la búsqueda global y local durante el proceso de optimización. Las constantes de aceleración sirven para dos propósitos en el algoritmo. Primero, ellas controlan la influencia relativa hacia *gbest* y $pbest_i$ respectivamente. Segundo, los dos coeficientes de aceleración cercanos a cero producirán una búsqueda fina en una región, mientras coeficientes cercanos a uno permitirán a la partícula la posibilidad de sobrepasar al *gbest* y al $pbest_i$, resultando en una búsqueda amplia. Una discusión detallada acerca de la influencia de estos parámetros puede encontrarse en [6].

El procedimiento de implementación del algoritmo de PSO se ilustra en el diagrama de flujo mostrado en la Figura 2. El proceso de optimización puede ser divido en 6 pasos como se describe a continuación.

(i) *Inicialización.* Durante este paso se determinan los límites de la posición y velocidad de las partículas; la población inicial (la cual se calcula aleatoriamente) y el valor $pbest_i$ correspondiente; los parámetros de (1) y la condición de parada del algoritmo.

(ii) *Evaluación de la población inicial*. En este paso, el costo para todas las partículas en la población inicial es evaluado de acuerdo con la función objetivo; y se selecciona la mejor partícula global, *gbest*.

(iii) *Actualización de posición y velocidad*. La posición y velocidad se actualizan de acuerdo con (1); si la posición y la velocidad de las partículas están por fuera de los correspondientes límites, éstos se ajustan a los valores establecidos.

(iv) *Evaluación de la población actualizada*. Similar a (ii), la posición actualizada de las partículas es evaluada de acuerdo al valor de la función objetivo; las partículas *gbest* y la *pbest* serán actualizadas si es necesario.

(v) *Verificación del cumplimiento de la condición de parada*. Esta condición puede corresponder con el número de iteraciones o con el mínimo valor de la función objetivo. Si la condición de parada no ha sido cumplida, el proceso de actualización (iii) será repetido; de lo contrario, el proceso de optimización finaliza.

(vi) *Resultados de salida*. La mejor solución obtenida durante el proceso de optimización, *gbest*, es la salida en este paso.

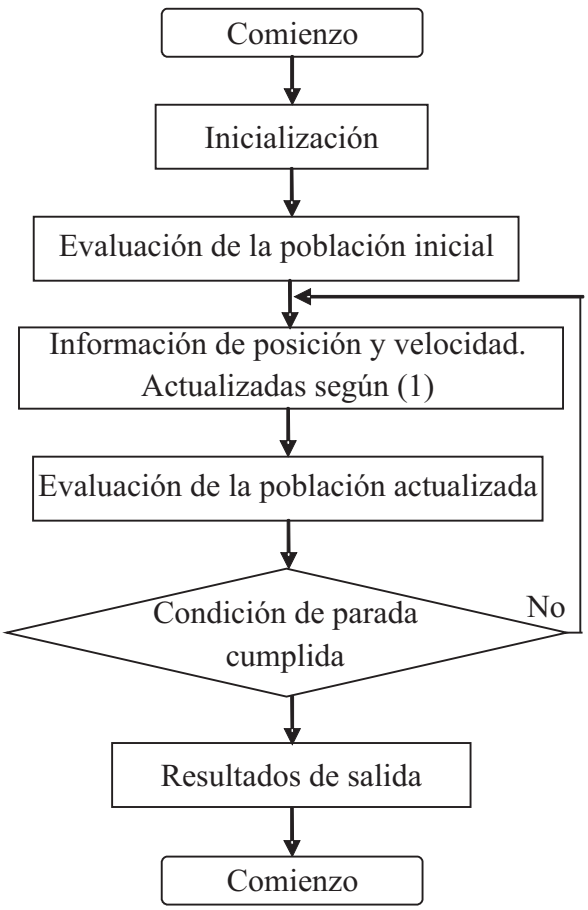

Figura 2 Diagrama de flujo del algoritmo PSO

A. Identificación de parámetros de un motor DC usando PSO

En identificación de sistemas un sistema con una estructura de modelo conocido pero con valores de parámetros desconocidos puede tratarse como una caja gris y resolverse como un problema de optimización. La idea básica es comparar la salida del sistema con la salida del modelo. La discrepancia entre las salidas del sistema y del modelo corresponde a la función objetivo a minimizar. Esta función se define con la medida de cuán bien la salida del modelo se acerca a la medida real de la salida del sistema.

Ziegler y N.B. Nichols [7] reconocieron que la respuesta a la función escalón de un gran número de sistemas de control de procesos exhibe una curva peculiar denominada "***process reaction curve***" como se muestra en la figura 3. Esta curva puede ser generada a partir de datos experimentales o de la simulación dinámica de la planta. La forma en *S* de la curva es común para muchos sistemas de orden superior entre ellos el utilizado en este artículo. Para sistemas que presentan este comportamiento la función de transferencia de la planta puede aproximarse usando (2).

$$\frac{Y(s)}{U(s)} = \frac{Ke^{-t_d s}}{\tau s + 1} \qquad (2)$$

Donde K es la ganancia en estado estacionario, τ es la constante de tiempo y t_d es el tiempo de retardo. En (2) K, τ y t_d son números reales y corresponden a los tres parámetros a ser identificados en este modelo. La Ec. (2) corresponde a un sistema de primer orden más un tiempo de retardo de t_d segundos. Las parámetros que componen la función de transferencia indicada en la Eq.(2) pueden ser determinados aproximadamente a partir de la información que provee la respuesta al escalón del sistema. La respuesta al escalón en dominio tiempo de (2) se expresa en (3)

$$y(t) = K\left(1 - e^{-\frac{(t-t_d)}{\tau}}\right)u(t-t_d) \qquad (3)$$

En la figura 3 se puede observar la respuesta al escalón de un sistema como el descrito en (2). En esta figura se nota que la tangente dibujada a partir del punto de inflexión de la curva, corresponde en magnitud a la relación K/τ, la intersección de la línea tangente con el eje de tiempo determina el tiempo de retraso t_d y t_s corresponde al tiempo en el cual la salida comienza a cambiar. Nótese que t_1 puede ser cualquier tiempo después del cual la salida alcanza el estado estacionario. Estos datos permiten asignar valores a priori de los parámetros que se van a identificar. En el caso de la señal de entrada, figura 4, u_0 es el valor antes de que el escalón sea generado, mientras u_1 es el valor de entrada después del escalón, y t_o indica el tiempo cuando la entrada comienza a cambiar.

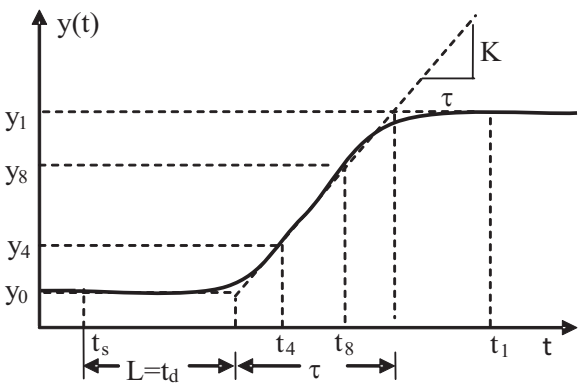

Figura 3 Respuesta típica ante una excitación tipo escalón

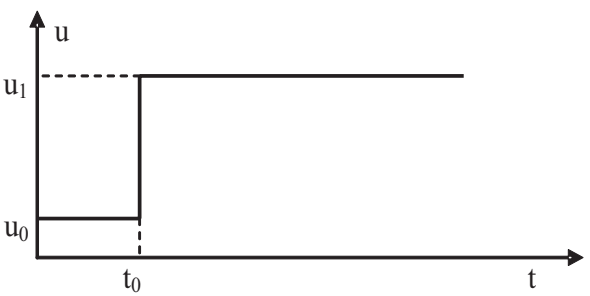

Figura 4 Función de entrada tipo escalón

Al igual que otros algoritmos de optimización, PSO no garantiza el éxito. La probabilidad de éxito se basa en la cantidad de partículas y el tamaño del espacio de búsqueda. Para un

espacio de búsqueda grande, debe utilizarse una población elevada para mejorar la tasa de búsqueda exitosa. No obstante, esto involucra más cálculos, y la velocidad de búsqueda puede reducirse significativamente. Por tanto, es necesario introducir un espacio de búsqueda razonable y eficiente. La determinación de las condiciones iniciales, a partir de las cuales la función objetivo es optimizada a través del uso de PSO, puede realizarse con base en la respuesta al escalón en el dominio del tiempo usando las siguientes relaciones analíticas propuestas en [3]. El valor de *K* puede ser determinado usando (4)

$$K = \frac{y_1 - y_o}{u_1 - u_o} \tag{4}$$

Para estimar el valor de τ, se determinan primero los tiempos t_4 y t_8 que corresponden respectivamente a 0,4 y 0,8 veces el máximo valor alcanzado por la curva de respuesta del escalón.

$$\begin{aligned} y(t_4) &= 0{,}4(y_1 - y_0) \\ y(t_8) &= 0{,}8(y_1 - y_0) \end{aligned} \tag{5}$$

Las ecuaciones (6) pueden emplearse para estimar τ:

$$\begin{aligned} N &= \left[\frac{1{,}075(t_4 - t_s)}{t_8 - t_4} + 0{,}5 \right] \\ \tau &= \frac{(t_4 - t_s) + (t_8 - t_s)}{2{,}16N} \end{aligned} \tag{6}$$

Donde *N* es un variable intermedia. El tiempo de retardo se evalúa usando la ecuación (7)

$$t_d = t_s - t_0 \tag{7}$$

Los valores obtenidos usando las ecuaciones anteriores constituyen el conocimiento a priori, el cual limitará el espacio de búsqueda. Estos valores permiten establecer los límites de búsqueda y con ello mejorar sustancialmente la eficiencia en la determinación de la solución al problema de identificación de los parámetros.

Función objetivo

La función objetivo es el criterio para evaluar el proceso de estimación de los parámetros. Esta función representa el grado de cercanía entre la salida del sistema real comparada con la salida del modelo cuyos parámetros se identifican usando PSO. En este caso la función objetivo se evaluará individualmente para la solución obtenida en cada partícula. La función de ajuste se obtiene usando (8)

$$F(p) = \frac{1}{t_1 - t_0} \int_{t_o}^{t_1} |Y_m(t) - Y(t)| dt \tag{8}$$

Donde $Y_m(t)$ es la salida estimada y $Y(t)$ es el vector que contiene los datos que corresponden a la velocidad del motor DC cuando se somete a una señal tipo escalón unitario.

Herramienta PSOt para Matlab

Para la implementación del algoritmo se utilizó la herramienta PSOt disponible para entorno Matlab desarrollada por Brian Birge [8]. Esta herramienta se puede descargar de [9]. En este trabajo se utilizó la versión actualizada el 20 de marzo de 2006. Los resultados de la optimización se obtuvieron con la función pso_Trelea_vectorized.m de la toolbox descargada de la Internet. Los parámetros de entrada de esta función se tomaron como los valores por defecto, excepto para los siguientes parámetros:

1. Rango de variables: esta matriz se organizó con los valores límite para las variables *K*, t_d y τ, respectivamente (ecuación 3). Las variables de tiempo se escalaron para utilizar valores en muestras. Estos límites se seleccionaron alrededor de una solución inicial determinada de acuerdo con las ecuaciones propuestas en [3]. El código para la especificación de esta matriz corresponde a: VarRange=[10 100;500 4000;10 500]. Si la solución óptima incluye uno de los límites establecidos, entonces este límite se debe aumentar ya que esta solución podría ser un óptimo local.

2. Gradiente de error global mínimo (*minimum global error gradient*): El valor por defecto para este parámetro es 10^{-25} (1e-25). En este trabajo se tomó como 10^{-3} (1e-3) para disminuir el tiempo de ejecución de algoritmo.

A continuación se encuentra el código utilizado para definir la función objetivo y para establecer los parámetros de entrada para la función pso_Trelea_vectorized.m.

```
function [out]=motorcontinua(in)
%Los datos reales se encuentran en el vector yd
global yd
load datosreales.mat
t=0:(length(yd)-1);
%Variables de la partícula
K=in(:,1); %Ganancia
td=in(:,2); %retardo
tao=in(:,3); %Constante de tiempo
P=length(K); %cantidad de partículas
%Valor de función a minimizar en cada partícula para primer orden (N=1)
out=zeros(P,1);
for p=1:P
   ym=datosmodelo(t,K(p),tao(p),td(p));
   out(p,1)=(t(2)-t(1))*sum(abs(ym-yd))/(t(length(t))-t(1));
end
clear global yd
%Funciones utilizadas
function ym=datosmodelo(t,K,tao,td)
ym=K.*(1-(exp(-(t-td)./tao))).*(t>=td);
end
%%% Fin datos reales
end
```

El siguiente código corresponde a la configuración de los parámetros de entrada para la función pso_Trelea_vectorized.m:

```
functname='motorcontinua';
D=3;
mv=4;
VarRange=[10 100;500 4000;10 500];
%K,td,tao
minmax=0;
PSOparams=[100,2000,24,2,2,0.9,0.4,1500,1e-3,150,NaN,0,0];
plotfcn='goplotpso';
%PSOseedValue; [optOUT,tr,te]=pso_Trelea_vectorized(functname,D,mv,VarRange,minmax,PSOparams,plotfcn,0);
```

Fase experimental

Los estudios experimentales en identificación de sistemas requieren modificar y monitorizar las respuestas de un sistema físico ante una o varias señales de entrada. El banco de pruebas utilizado en este trabajo consiste de un motor DC de 48 *V* con una potencia nominal de 24 *W*. El sistema de monitorización y captura de datos es implementado usando Matlab®/Simulink® y como interfaz de hardware se utiliza el sistema dSpace®. Este sistema es compilado usando la herramienta *Real Time Workshop* (RTW®) y luego se implementa en tiempo real con la plataforma dSpace®. Aunque el proceso de identificación de los parámetros del motor es *offline*, el sistema de adquisición de datos propuesto permite muestrear en tiempo real la señal de velocidad y la señal de voltaje de entrada al motor DC. Las señales son capturadas con una frecuencia de muestreo de 5 KHz. La señal de entrada al motor es proporcionada por un generador de funciones el cual suministra un tren de pulsos bipolar. La señal de velocidad del motor, respuesta del sistema, es pre-procesada usando un filtro discreto pasabajas de 6º orden con frecuencia de corte de -3dB igual a 247 Hz. Este filtro se utiliza para eliminar ruido de alta frecuencia introducido por el enconder empleado para censar la velocidad del motor.

Resultados y discusión

Para la identificación del modelo lineal del motor DC se implementó una prueba utilizando como señal de entrada un tren de pulsos rectangulares con frecuencia de 0,5 Hz, con un ciclo de trabajo

del 50% y una amplitud de tensión de ± 30 V. A partir de esta prueba se tomó la respuesta de medio ciclo para obtener el modelo del motor a partir de la respuesta al escalón. En consecuencia se eliminó la componente de continua y se normalizaron las señales para obtener la señal de entrada como se muestra en la figura 5. Los valores iniciales, alrededor de los cuales se establecen los rangos de búsqueda, se determinaron con las ecuaciones (4-7) a partir de los datos medidos de la respuesta al escalón (figura 7) y corresponden a los siguientes valores: $K=11,7,\ t_d=0,64\ [s]$ y $\tau = 0,088\ [s]$. Las pruebas se efectuaron en repetidas ocasiones permitiendo que el proceso aleatorio implementado en el algoritmo de PSO produjera diferentes ubicaciones iniciales para las 24 partículas utilizadas. Como se ilustra en la figura 6, independientemente de las posiciones iniciales de las partículas, el algoritmo de búsqueda presentó convergencia alrededor de 140 épocas. La figura 6 ilustra la evolución del mejor valor de la función objetivo (*gbest*) para los distintos escenarios desarrollados a partir de diferentes posiciones iniciales. Los resultados experimentales de la respuesta en la velocidad del motor, tomando como entrada el escalón unitario, se muestran en la figura 7 junto a los valores obtenidos usando los parámetros determinados con el algoritmo de optimización PSO. La función de transferencia obtenida (con un valor de función objetivo de 0,4588) se muestra en (9).

$$\frac{Y(s)}{U(s)} = \frac{11,226\ e^{-0,6356\ s}}{0,0925\,s+1} \qquad (9)$$

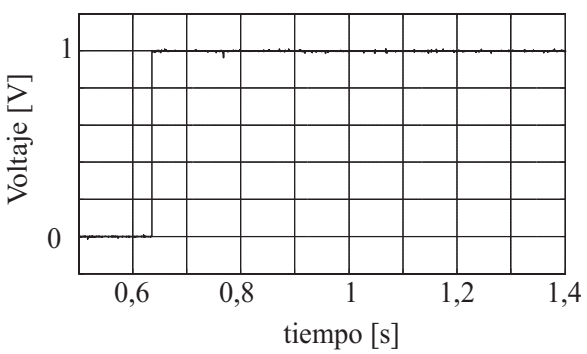

Figura 5 Función de entrada al motor DC

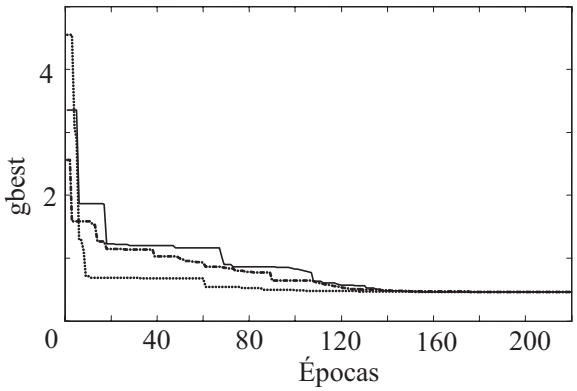

Figura 6 Evolución del *gbest* para diferentes posiciones iniciales de partículas en función del número de épocas

La figura 7 ilustra cómo el modelo proporciona una respuesta al escalón con una mínima desviación con respecto a la respuesta real del motor. A través del uso de esta metodología es posible implementar un sistema de identificación de parámetros a través de la definición de la apropiada función objetivo en combinación con PSO.

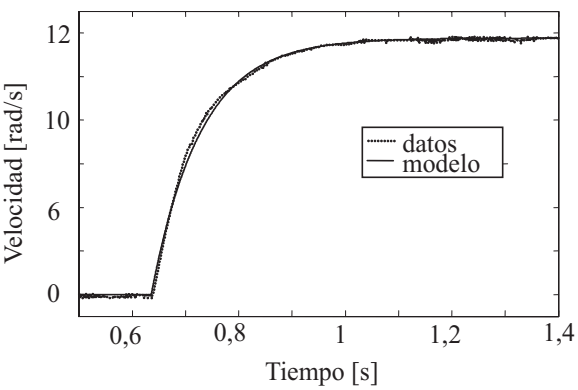

Figura 7 Respuesta al escalón unitario del motor y del modelo encontrado utilizando PSO

En resumen, se puede establecer que PSO es un buen candidato para identificación de sistemas tipo caja gris del tipo ilustrado en la ecuación (2). Aún cuando el modelo del sistema es de mínimo orden (orden 1) la función objetivo presenta mínimos locales. En las figuras 8 y 9 se presentan las curvas de nivel de la función

objetivo, en función de las variables t_d y τ, para dos valores de la constante de proporcionalidad K. Se puede observar que la región donde se puede encontrar el valor mínimo de la función objetivo es diferente en cada caso. Asimismo, se puede notar cómo el valor estos dos mínimos (aproximadamente 46,65 para K=1,59 y 28,98 para K=31,81) no corresponde al valor mínimo obtenido, el cual es de 0,4588 para un valor de K=11,226. De esta manera se puede verificar la efectividad del algoritmo PSO para evitar los mínimos locales.

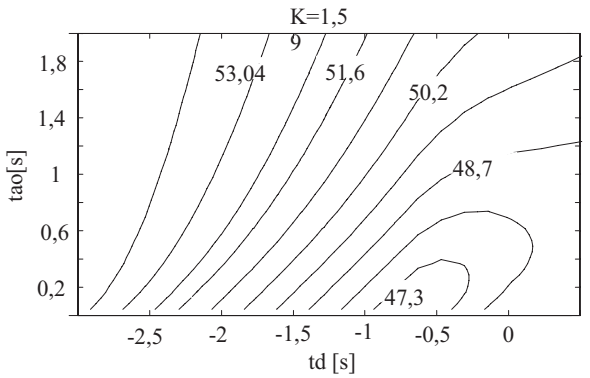

Figura 8 Curvas de nivel de la función objetivo para un mínimo local con K=1,59

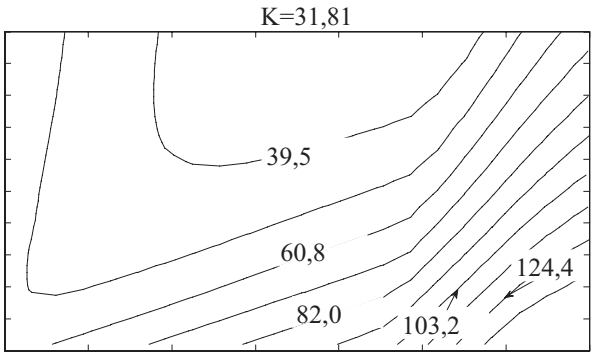

Figura 9 Curvas de nivel de la función objetivo para un mínimo local con K=31,81

Conclusiones

Es este artículo, se ha propuesto el uso del algoritmo PSO para la identificación de parámetros en la función de transferencia de un sistema tipo caja gris. El uso de esta técnica puede simplificar el algoritmo necesario para la identificación del sistema. Los resultados experimentales mostraron la eficacia del sistema propuesto en identificar el sistema consistentemente y en un intervalo de tiempo corto (aproximadamente 20 s). En el futuro, este trabajo se puede extender para procesos de identificación más complejos y de mayor tamaño.

Referencias

1. G. Panda, D. Mohanty, B. Majhi, G. Sahoo. *Identification of nonlinear systems using particle swarm optimization technique*. Evolutionary Computation. CEC. IEEE Congress Singapore. Sept. 2007. pp.3253-3257.
2. X. Peng, G.K. Venayagamoorthy, K. A. Corzine. *Combined Training of Recurrent Neural Networks with Particle Swarm Optimization and Backpropagation Algorithms for Impedance Identification*. Swarm Intelligence Symposium. SIS. IEEE. Honolulu 1-5 April 2007. pp.9-15.
3. Z. Dong, P. Han, D. Wang, S. Jiao. *Thermal Process System Identification Using Particle Swarm Optimization*. IEEE International Symposium on Industrial Electronics. Montreal. Vol.1. 2006. pp.194-198.
4. L. Liu. *Robust fault detection and diagnosis for permanent magnet synchronous motor*s. Ph.D. dissertation. Dept. Mech. Eng. Florida State University. Tallahassee (FL) 2006. pp. 83-105.
5. J. Kennedy, R. Eberhart. "Particle Swarm Optimization," *Proceedings of IEEE International Conference on Neural Networks*. Perth (Australia) Vol. 4. 1995. pp. 1942-1948.
6. Y. Shi. *Particle Swarm Optimization*. Ed. Electronic Data Systems Inc. IEEE Neural Networks Society. Kokomo (USA). 2004. pp. 8-13.
7. J. G. Ziegler, N. B. Nichols. "Optimum settings for automatic controllers". *Trans. A.S.M.E.* Vol. 64. 1942. pp. 759-765.
8. Birge, B., "PSOt - a particle swarm optimization toolbox for use with Matlab," Swarm Intelligence Symposium. 2003. SIS '03. *Proceedings of the 2003 IEEE.* Indianapolis. April 24-26. 2003. pp. 182-186.
9. http://www.mathworks.com/matlabcentral/fileexchange/7506. Consultada el 16 de diciembre de 2008.