https://doi.org/10.17979/spudc.9788497498081.0907

MODELO NEURONAL MULTIVARIABLE DE UN SISTEMA TWIN-ROTOR

Luis Ignacio Ruiz
DISA, EIB, UPV/EHU, Alda. Urquijo S/N, lruiz048@ikasle.ehu.eus

Eloy Irigoyen, Vicente Gómez, Fernando Artaza
DISA, EIB, UPV/EHU, Alda. Urquijo S/N, {eloy.irigoyen,vicente.gomez,fernando.artaza}@ehu.eus

Resumen

Este trabajo tiene como objetivo contribuir al uso de técnicas de Computación Inteligente para la obtención, en este caso, de un modelo perfectamente funcional de un sistema MIMO (Multi-Input Multi-Output) de tipo Twin-Rotor. Dicho sistema presenta en determinados puntos de trabajo un elevado grado de no linealidad debido a las características intrínsecas del mismo, donde sus dos entradas y sus dos salidas quedan fuertemente acopladas, poniendo en compromiso sus condiciones de estabilidad, ante estrategias de control tradicionales. Este sistema consta de un rotor superior y otro lateral de cola, que controlan el Pitch y el Yaw respectivamente. El modelo neuronal NARX obtenido se presenta como una caja negra ("Black-Box") que reproduce de forma aproximada la dinámica del funcionamiento del sistema, manteniendo los comportamientos no lineales más generales, tal y como lo demuestran los resultados presentados.

Palabras Clave: red neuronal NARX, sistema MIMO no lineal, modelado, Computación Inteligente, Twin-Rotor.

1 INTRODUCCIÓN

Los sistemas de control se diseñan con una configuración cuyo objetivo es obtener un comportamiento deseado en un proceso específico. Los más usados actualmente son los PID (Proporcional-Integral-Derivativo) por su bajo coste, simplicidad y eficiencia. Sin embargo, esta tarea se vuelve más compleja cuando dicho proceso o sistema a controlar tiene un grado de no linealidad muy elevado y/o cuando no se conoce el modelo de dicho sistema con exactitud.

Esta no linealidad es producida por las características dinámicas reales que puede reunir un sistema, es decir, que no es posible plantear soluciones linealizadas de igual modo para cada uno de los puntos de operación. Estos sistemas suelen tener más de una entrada y salida en sus modelos reales, acopladas entre ellas, y cuyas inferencias pueden llegar a inestabilizar el sistema si no se conocen de forma adecuada, lo que hace que la obtención de un modelo aproximado sea aún más difícil y costoso.

Por ello, cada vez más empresas y compañías hacen uso de aquellas técnicas de computación inteligente. Estas técnicas, tales como pueden ser las redes neuronales, algoritmos genéticos, métodos de toma de decisiones, o lógica borrosa, permiten entre otras cosas adaptarse a esas no linealidades en esquemas multivariables, obteniendo un modelo que facilite posteriormente la selección de una estrategia de control adecuada.

1.1 ESTUDIOS E INVESTIGACIONES EXISTENTES

En los últimos años, se ha producido un aumento en el uso de las técnicas inteligentes, con objetivos muy variados como su aplicación en modelado, control o Big Data. Por ejemplo, en [1] se utilizan redes neuronales para algo tan actual como es el modelado de imanes superconductores, necesarios para la construcción de reactores de fusión de tipo *Tokamak*, ahorrando de esa forma mucho tiempo y dinero en obtención de dichos modelos.

En [2] hacen uso de redes neuronales para obtener evaluaciones de las curvas de energía de turbinas de forma más exacta. Debido a la relación no lineal que existe entre la energía de salida y sus principales parámetros obtuvieron buenos resultados en el modelado de dichas curvas energéticas, reduciendo considerable-mente el error producido, tanto absoluto, como aleatorio en comparación con los métodos convencionales.

Incluso en [3] utilizan redes neuronales para el modelado directo del proceso de separación líquido-líquido. En dicho trabajo obtuvieron unos resultados muy superiores a los métodos tradicionales por escalas de polaridad, donde su utilidad se ve reducida debido al manejo de grandes cantidades de datos.

En lo concerniente a las aplicaciones en aeronáutica, desde hace tiempo se ha podido ver el futuro que tendrían las técnicas del área de la inteligencia computacional. Un ejemplo de estas soluciones presentadas en este tipo de aplicaciones es la ofrecida en un enfoque global por [4] o de modo más concreto por [5], donde realizan un diseño del tamaño de un helicóptero mediante la obtención de una red neuronal optimizada para tal fin con algoritmos genéticos. Más actualmente se pueden encontrar trabajos que hibridan varias de estas técnicas, como los presentados por Sierra y Santos [6, 7].

Más específicamente, teniendo en cuenta el sistema estudiado en el presente trabajo, se han realizado estudios previos en los que se pueden encontrar el uso de redes neuronales para realizar modelos de controladores para el propio Twin-Rotor, como puede ser el de Rahideh y Bajodah [8] o el de Khakshour y Khanesar [9] donde además la estructura de la red neuronal incluye un componente de lógica borrosa.

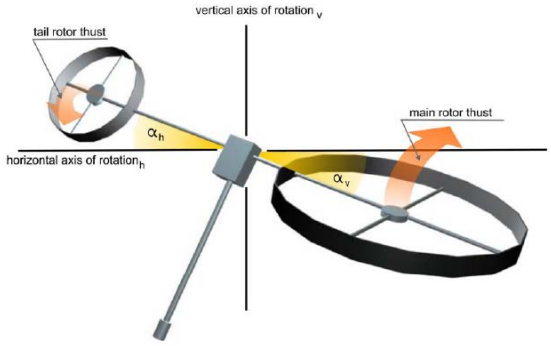

Figura 1: Maqueta Twin-Rotor

En otros trabajos se han tratado de diseñar observadores mediante redes neuronales para reproducir la dinámica del Twin-Rotor, como llevaron a cabo Shaik y Purwar [10].

A continuación se estructura el trabajo presentado explicando primero la metodología básica con que se va a desarrollar el modelo respecto a las especificaciones técnicas del Twin-Rotor. Posteriormente, se desarrollará paso a paso dicha metodología mostrando las entradas extraídas para el entrenamiento de las redes y ejemplos de la estructura de las mismas. Por último, se comentarán los resultados finales, sus razones y comparativas, terminando con un apartado de conclusión que resume tanto el trabajo realizado como los futuros proyectos que se esperan desarrollar.

2 METODOLOGÍA

La metodología aplicada requiere de una selección del tipo de red más viable para su uso en sistemas con inestabilidades y no linealidades tan altas. En este caso, se presenta el tipo de red NarX (Nonlinear-AutoRegressive network with eXogenous inputs) como aquella capaz de obtener dicha dinámica de la forma más eficiente. Esto es posible debido a su configuración *feedforward* que describe de forma discreta el sistema no lineal haciendo uso de entradas y salidas anteriores en función del número de retardos configurados en la red. Para el desarrollo del mencionado estudio se ha utilizado la *toolbox* de MATLAB ® (versión R2012b), denominada *Neural Networks*.

Esta y otras razones de su uso vienen detalladas en la revisión de Hong y Mitchell [11] donde comparan varios métodos para la identificación de este tipo de sistemas. Algunos estudios como el de Sahoo y Dash [12], donde estudian la utilización de este tipo de redes en la identificación de sistemas dinámicos no lineales.

El primer paso en todo proceso de entrenamiento de redes neuronales es obtener una cantidad relevante de datos de entrada y salida obtenidos del sistema, de forma que abarquen un rango de la dinámica del mismo lo suficientemente amplio como para cubrir gran parte de su comportamiento dinámico, siempre teniendo en cuenta el rango de funcionamiento. En nuestro caso, Twin Rotor de Feedback, este rango varía entre -2.5/2.5 voltios a la entrada.

Después de obtener dichos datos, se deben diseñar una serie de configuraciones de la red neuronal a estudiar, de forma que se pueda obtener aquella con una buena adaptación, capacidad de predicción a un horizonte determinado y menor coste computacional. En esta configuración entra tanto la selección del número de neuronas de la capa oculta como el número de retados aplicados a las entradas y a las salidas.

Una vez que se ha decidido cuál es el rango del número de retardos y número de neuronas posibles, se realizará un entrenamiento serie-paralelo a cada una de esas posibles configuraciones. Tras dicho entrenamiento se realizarán las correspondientes validaciones con nuevos datos, para comprobar que la red neuronal ha aprendido correctamente la dinámica del sistema, llevando a cabo dicho estudio mediante la comparación del error medio en cada validación.

Tras haber comparado los errores medios en las validaciones de todas las posibles configuraciones, se obtendrá un conjunto de posibles redes con comportamiento apropiado. Este conjunto se seleccionará de las redes que hayan dado mejores resultados, menores errores, en las diferentes comparaciones a realizar en el proceso de validación.

Por tanto, el último paso sería realizar una prueba de predicción con unos datos distintos a los de entrenamiento y validación, con un horizonte de predicción definido.

Siguiendo este proceso, se obtiene un modelo neuronal equivalente al modelo real no lineal que matemáticamente es desconocido, pudiéndose aplicar después para obtener un control de ese sistema de forma más rápida y sencilla.

3 OBTENCIÓN DEL MODELO NEURONAL

La red neuronal se compone de una o varias capas, cada una con un número de neuronas a estudiar, las cuales interconectan las entradas y salidas de la red. Por cada conexión existe un valor asignado o peso que es modificado cada iteración durante el entrenamiento. De igual modo se establece una serie de condiciones de parada, según las cuales una vez alcanzados unos resultados aceptables o si el error es muy grande y se detecta que no mejora, el entrenamiento se detiene.

Este tipo de redes han resultado ser las que mejor comportamiento han mostrado para tratar, tanto con sistema no lineales, como lineales multi-entrada y salida. Para conseguir los mejores resultados se deben seguir unas pautas:

Paso 1: Preparación de los datos de entrenamiento. A partir del sistema objeto de estudio, se extraen una batería de entradas y sus correspondientes salidas.

Para ello, primero se estudia el rango de funcionamiento de dicho sistema, comprobando qué valores máximos y mínimos admite antes de desestabilizarse.

El Twin-Rotor dispone de sus valores de Pitch y Yaw introducidos en radianes, convirtiendo posterior-mente las señales de control equivalentes en voltios, donde su rango va desde -2.5 a 2.5 voltios, lo que equivale a un rango de 1.67 a -0.87 radianes en el Pitch y 4.5 a -4.5 radianes en el Yaw. El Pitch tiene un rango menor debido a que el sistema se hace menos controlable en posiciones tan alejadas de su punto de reposo inicial, el cual muestra una mínima inclinación del Twin-Rotor.

Los datos extraídos para su entrenamiento se toman teniendo en cuenta que varíe entre todos los límites posibles para tratar de obtener la dinámica completa. Dicha señal viene dada por las siguientes entradas y sus respectivas salidas:

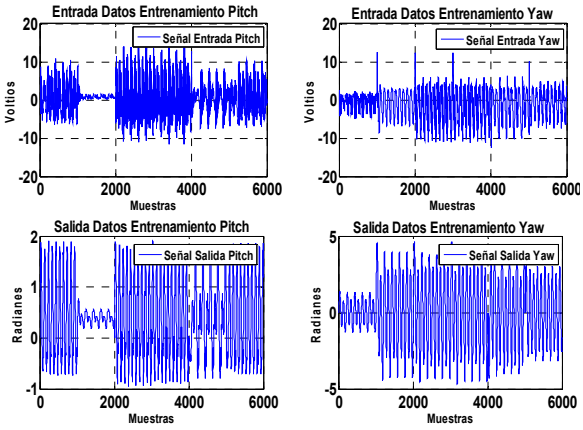

Figura 2: Señales I/O de entrenamiento

También se obtienen otras entradas y salidas aleatorias dentro de los límites de funcionamiento para validar cada configuración en el siguiente paso.

Por supuesto, los picos de más de 2.5 o -2.5 voltios no son tenidos en cuenta y por tanto son limitados antes de introducirlos al modelo mediante un saturador. Esta señal será la usada en el entrenamiento del siguiente paso.

Paso 2: Entrenamiento y validaciones para cada configuración. Con los datos de entrenamiento obtenidos, se define la cantidad de posibles configuraciones para el modelo neuronal. Se establece un número mínimo de neuronas (6) y un máximo (30), con un paso mínimo para que exista una diferencia palpable (6:2:30).

A su vez, se definen los retardos mínimos a entrada y salida así como sus máximos. Se establece un máximo de 5 retardos en entrada y salida, y un mínimo de 1 ya que se considera que al menos un retardo debe ser introducido.

La estructura del funcionamiento de los retardos viene dada por el siguiente esquema ejemplo en el que existen 10 neuronas, 3 retardos en la entrada y 5 en el target:

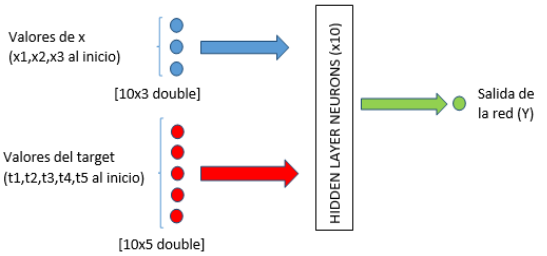

Figura 3: Esquema ejemplo red con retardos en LA

Este diseño se da debido a que en un principio las pruebas se realizan con un modelo de red neuronal en lazo abierto, ya que en lazo cerrado nunca va a dar

mejores resultados que su equivalente en lazo abierto, por lo que primero se selecciona la configuración en lazo abierto y luego se podrá proceder a hacer pruebas con la misma configuración en lazo cerrado. Esto es así debido a que en lazo cerrado, la red no solo tiene que tratar de adecuarse a la dinámica del sistema si no que al ser realimentadas sus estimaciones también tiene que adecuarse al error que su predicción produce, la cual no es la mayoría de los casos constante.

Por tanto, se entrenarán todas las posibles configuraciones, un total de 108 posibilidades, con 10000 iteraciones por entrenamiento cada una. El entrenamiento usa la función por defecto de Levenberg-Marquardt, que es de los más usados cuando se trata de sistemas no lineales y se realizará 10 veces por cada posibilidad para tratar de cubrir todos los óptimos posibles donde pueden terminar dichas redes.

Tras ser entrenadas cada configuración se valida 3 veces con señales distintas a la de entrenamiento y entre sí, incluyendo en una de ellas una perturbación para comprobar que la red no se ha sobreentrenado, obteniendo el error medio cuadrático en cada caso.

Paso 3: Predicción con las redes obtenidas. Se compara a continuación todos los errores de cada validación para cada configuración y se guardan aquellas configuraciones con menor error para cada validación.

Se observan dos posible redes óptimas, por lo que la última prueba que deben pasar será una predicción a un horizonte determinado H=10. Se utiliza una nueva señal de nuevo distinta a las anteriormente usadas para que sirva como una predicción válida.

Para ello, se requiere de un algoritmo que permita guardar por columnas los valores predichos desde 1 a 10, desplazando el puntero desde el que se comienza a predecir de 1 en 1. De esta forma, se tiene la señal para cada horizonte de predicción menos los valores del número de retardos más dicho horizonte. Es decir, si la red tiene 4 retardos y en ese instante predice el octavo valor, la señal de la octava columna guardada tendrá los valores de la señal completa menos 12.

Paso 4: Comprobación de lazo abierto a lazo cerrado. Cuando se decide cuál es la red que mejor predice o que menos se degrada con cada predicción, se realiza una última prueba con esa configuración para dilucidar qué estructura entre lazo abierto y lazo cerrado de esa red es la que muestra un comportamiento optimizado.

Como ya se tienen los resultados en lazo abierto, se realizan pruebas usando distintas posibilidades de esa misma configuración de red pero en lazo cerrado.

La primera es con la función de Levenberg-Marquardt convirtiendo de forma directa la red entrenada en lazo abierto a lazo cerrado. La siguiente sería igual pero con la función Backpropagation con regularización por redes bayesianas, ya que suele funcionar mejor en los casos en los que la red está en lazo cerrado. Para la tercera y cuarta, se crean nuevas redes con la misma configuración pero directamente en bucle cerrado y las respectivas funciones indicadas para cada una.

A continuación, se entrena cada posibilidad de la misma forma que se hizo a su homóloga en lazo abierto. Una vez entrenadas, se realiza de nuevo la prueba de predicción para cada una de ellas de la misma forma que en bucle abierto pero teniendo en cuenta que el funcionamiento en lazo cerrado (LC) varía respecto del de lazo abierto, como se observa en la figura 5, equivalente al anterior esquema en lazo abierto:

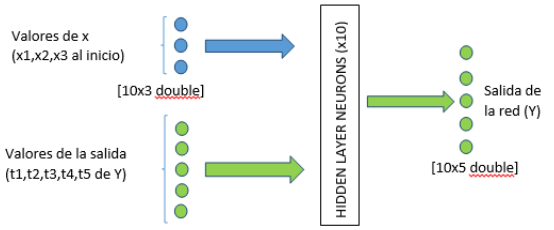

Figura 4: Esquema ejemplo red con retardos en LC

Esto muestra que en vez de introducir valores reales de salida como entradas, realimenta aquellos valores predichos por la red, por lo que se debe tener en cuenta para el algoritmo de predicción.

4 MODELO NEURONAL PROPUESTO Y ESQUEMAS

Como se ha mencionado, del entrenamiento y validación de todas las posibles configuraciones se extraen aquellos errores medios cuadráticos (MSE) más pequeños para cada validación. En este caso, dichos valores corresponden a las siguientes configuraciones con N siendo el número de neuronas, Re los retardos a la entrada y Rt los retardos en el target:

Tabla 1: Menores MSE obtenidos de validaciones.

	Menores MSE	Configuración
Error 1	1.1433e-07	N: 16, Re:3, Rt: 4
Error 2	1.2419e-06	N: 6, Re: 4, Rt: 4
Error3	8.7979e-08	N: 16, Re: 3, Rt: 4

En la prueba realizada para la predicción a un horizonte, se comparan los resultados entre ambas posibilidades, para unos horizontes de H=2, H=4, H=8 y H=10:

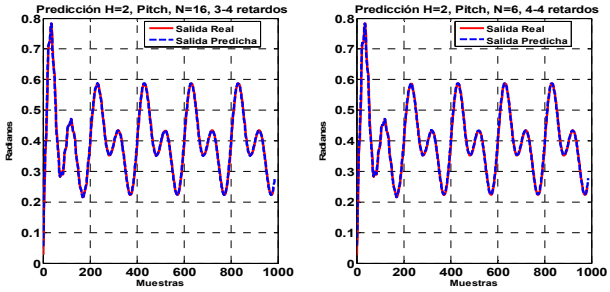

Figura 5: Comparación con H=2, Pitch

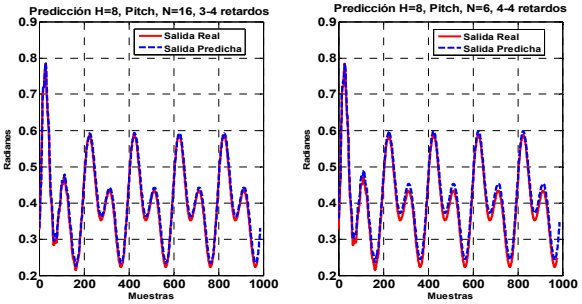

Figura 6: Comparación con H=8, Pitch

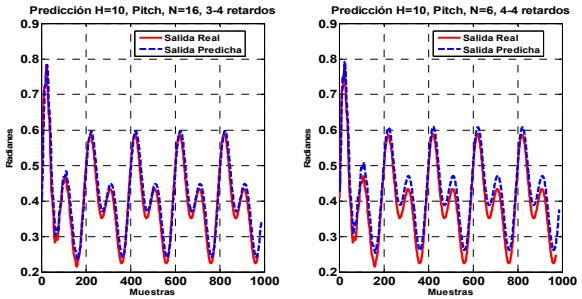

Figura 7: Comparación con H=10, Pitch

Se observa una degradación mayor en la predicción de la configuración con 6 neuronas y 4 retardos a la entrada y en el target, por lo que se decide que el mejor modelo a usar será el de 16 neuronas, 3 retardos a la entrada y 4 en el target. No constituyen una gran cantidad de gasto computacional porque su número de conexiones no es alto, por lo que además es un valor añadido.

El último paso de comparar la predicción de esa configuración elegida en lazo cerrado se divide en cuatro tipos de posibilidades como se ha comentado. Se prueba la configuración en lazo cerrado convirtiendo de forma directa desde lazo abierto y entrenada con Levenberg-Marquadt (L-M) y otra con la función Backpropagation con regularización por redes bayesianas (B-R) y después creando una nueva red con esa misma configuración y entrenándolas de igual forma a la de lazo abierto.

A continuación se compara la predicción con H=2 de lazo cerrado en los cuatro casos de entrenamiento en lazo cerrado con la de lazo abierto:

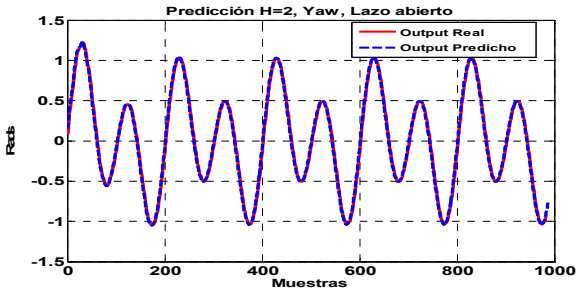

Figura 8: Comparación con H=2, Yaw en LA

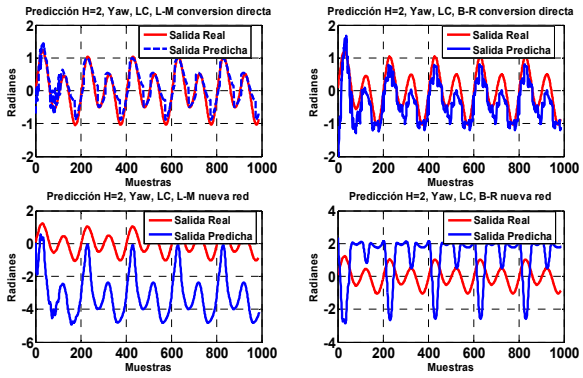

Figura 9: Comparaciones con H=2, Yaw en LC

La predicción incluso solo con H=2 en lazo cerrado con cualquiera de la posibles opciones es mucho peor que los resultados en lazo abierto. Esto se debe entre otras cosas a que a la hora de entrenar la red en lazo cerrado, esta no solo debe tratar de adaptarse a la dinámica del sistema si no al propio error realimentado producido por los valores predichos de la red, por lo que aumenta su gasto computacional hasta alcanzar unas respuestas aceptables.

5 CONCLUSIÓN

En este trabajo se ha propuesto el uso de redes neuronales para el modelado de sistemas MIMO altamente no lineales. Los resultados presentados son

de un prototipo de modelo aún por probar en un entorno real aplicable. La dificultad del modelado gracias a este método se ve enormemente reducida, con lo que se ahorraría en tiempo, que en muchos casos equivale a un mayor coste.

La metodología seguida es de las más usadas cuando se trata de este tipo de técnicas, aunque en muchos casos se puede variar obteniendo igualmente resultados satisfactorios. No es imprescindible conocer el funcionamiento interno de una red neuronal pero desde luego ayuda conocer cómo puede variar y porqué tras distintos entrenamientos puede llegar a una u otra solución satisfactoria, sea o no la que se considera como óptima global.

El siguiente paso será realizar un control adaptativo para este modelo mediante un algoritmo que utiliza un conjunto de técnicas inteligentes, como algoritmos genéticos para obtener las mejores acciones de control y lógica difusa para la selección de aquella que más convenga según el instante en el que se encuentre el sistema. Comprobando su total funcionamiento en una maqueta real de Twin-Rotor de la empresa Feedback.

Agradecimientos

Este trabajo se ha realizado en colaboración con el grupo de investigación de control inteligente (GICI) de la Universidad del País Vasco (UPV/EHU).

Referencias

[1] A. Froio, (2016) “A Design and optimization of Artificial Neural Networks for the modelling of superconducting magnets operation in tokamak fusion reactors”, *Journal of Computational Physics*, 321, pp. 476-491

[2] F. Pelletier, C. Masson, A. Tahan (2016) “Wind turbine power curve modelling using artificial neural network”, *Renewable Energy*, 89, pp. 207 - 214.

[3] M. Moghadam, S. Asgharzadeh (2016) “On the application of artificial neural network for modeling liquid-liquid equilibrium”, *Journal of Molecular Liquids*, 220, pp. 339 - 345.

[4] W. E. Faller, S. J. Schreck (1996) “Neural networks: Applications and opportunities in aeronautics”, *Progress in Aerospace Sciences*, 32, pp. 433-456.

[5] X. Lu, H. Liu, G. Wang, Z. Wu (2006) “Helicopter Sizing Based on Genetic Algorithm Optimized Neural Network”, *Chinese Journal of Aeronautics*, 19, pp. 212 - 218.

[6] J.E. Sierra, M. Santos (2105) “Adaptive Neural Control-Oriented Models of Unmanned Aerial Vehicles”, *Advances in Intelligent Systems and Computing*, SOCO 2015, vol. 368, 329-337.

[7] J.E. Sierra, M. Santos (2104) “Modelado de un vehículo aéreo no tripulado mediante combinación de técnicas paramétricas y neurodifusas”, *Actas del ESTYLF* (XVII Congreso Español sobre Tecnologías y Lógica Fuzzy), pp. 339-344.

[8] A. Rahideh, A.H. Bajodah, M.H. Shaheed (2012) “Real time adaptive nonlinear model inversion control of a twin rotor MIMO system using neural networks”, *Engineering Applications of Artificial Intelligence*, 25, pp. 1289 - 1297.

[9] A. J. Khakshour, M. A. Khanesar (2016) “Model reference fractional order control using type-2 fuzzy neural networks structure: Implementation on a 2-DOF helicopter”, *Neurocomputing*, 193, pp. 268 - 279.

[10] Shaik, (2011) “Real-time implementation of Chebyshev neural network observer for twin rotor control system”, *Expert Systems with Applications*, 38, pp. 13043 - 13049.

[11] X. Hong, R. J. Mitchell, S. Chen, C. J. Harris, K. Li, G. W. Irwin (2008) “Model selection approaches for non-linear system identification: a review”, *International Journal of Systems Science*, 39, pp. 925-946.

[12] H.K. Sahoo, P. K. Dash, N. P. Rath (2013) “NARX model based nonlinear dynamic system identification using low complexity neural networks and robust H∞ filter”, *Applied Soft Computing*, 13, pp. 3324 - 3334.