Disponible en www.sciencedirect.com
ScienceDirect
Revista Iberoamericana de Automática e Informática industrial 9 (2012) 3–13
www.elsevier.es/RIAI

# Identificación Inteligente de un Proceso Fermentativo Usando el Algoritmo GMDH Modificado.

F. Hernández [a,*], F. Herrera [b]

[a] *ECASA, prof. adjunto Universidad de Camagüey, 1ra paralela #15 e/ Julio Sanguily y G de Quezada, Camagüey, Cuba.*
[b] *Departamento de Automática, Fac. Ingeniería Eléctrica, Universidad Central de Las Villas, Km 7 ½ Carretera de Camajuaní, Santa Clara, Cuba.*

**Resumen**

En este trabajo se aborda, de manera particular, un método para el diseño del algoritmo conocido como Group Method of Data Handling, GMDH, típico con lazo recurrente. Una modificación en una de sus fases de entrenamiento permite ampliar el número de variables utilizadas en cada capa y con ello el área de regresión. Consecuentemente se puede obtener una estructura optimizada en sí misma de mayor complejidad, posibilitando la aparición de lazos recurrentes en las capas intermedias. Lo anterior permite una reducción del error en la modelación de procesos no lineales de lento comportamiento, como el crecimiento celular en biorreactores. El modelo se probó en una fermentación tipo feed-batch de la levadura *Pichia pastoris*. La estabilidad y capacidad de generalización es demostrada. El método propuesto es comparado con el GMDH típico recurrente y con otras estructuras de redes neuronales clásicas. 

*Palabras Clave:*
redes neuronales, recurrente, algoritmo genético, modelación, fermentación.

## 1. Introducción

La identificación de sistemas se ha desarrollado para la determinación de modelos matemáticos a partir de datos conocidos de un proceso. (Mark and Xin, 2002). Las técnicas de la Inteligencia Artificial, IA, se están aplicando significativamente en este campo en las últimas décadas, y dentro de ellas las conocidas como Redes Neuronales Artificiales, RNA, caracterizadas por sus propiedades de aprendizaje y generalización (Leiva, 2006).

Una importante característica a considerar en el empleo de las RNA para la identificación de sistemas, es su potencialidad para la inducción, la cual puede ser implementada mediante software. (Miroslav Šnorek, 2006). En este sentido existen algunos métodos para la construcción de modelos inductivos, uno de ellos es comúnmente conocido como Group Method of Data Handling, GMDH, (Ivakhnenko A. G., 1971).

Solla (1989) demuestra que la probabilidad de que una RNA muestre un determinado comportamiento depende no sólo del algoritmo de aprendizaje, sino también de su arquitectura. Generalmente recae sobre el método de aprendizaje el lograr una correcta correspondencia del modelo con el sistema a modelar, pero tal y como argumentan Happel and Murre (1990), un método potencialmente más versátil para cambiar esta correspondencia modelo – proceso se basa en imponer restricciones sobre la topología de la red, la cual debe corresponder en la mayor medida posible a la estructura del proceso que se modele. En este sentido, para procesos complejos, con variables intermedias o de salida que se retroalimenten en uno o varios puntos, se requiere una RNA con lazos internos de realimentación, o sea una RNA recurrente, obteniéndose de esta forma modelos predictivos con una mejor eficiencia en la predicción a largo plazo (Yuan and Vanrolleghem, 1999). Múltiples bibliografías muestran, además, el empleo combinado de otras herramientas para mejorar el desempeño de la red, a partir de cambios en su estructura. Una de ellas, de uso creciente en la actualidad, es el Algoritmo Genético, AG, basado en la teoría de la evolución biológica aplicada en la optimización de modelos matemáticos de sistemas (Holland, 1975; Ferreira, 2001, 2004).

Muchos de estos métodos encuentran aplicación en diversos campos como la bioingeniería, donde los modelos utilizados constituyen problemas complejos en razón de las características de los procesos a estudiar (Passoni, 2005). Este concepto deviene del hecho de que la actividad biológica genera información con características particulares, destacándose las siguientes:

* Autor en correspondencia.
*Correos electrónicos*: lozano@cmw.ecasa.avianet.cu (F. Hernández), herrera@uclv.edu.cu (F. Herrera)
URL: www.uclv.edu.cu (F. Hernández)


doi:10.1016/j.riai.2011.11.001

- Heterogeneidad debida, fundamentalmente, a la complejidad estructural de los objetos vivos que se modelan.
- La información obtenida presenta una dinámica compleja, asociada a las propiedades de los fenómenos que se desean estudiar.

La metodología de identificación de un proceso fermentativo, estudiada en este trabajo, se basa en la introducción de lazos recurrentes en las capas intermedias de una red GMDH típica, a partir de la modificación de una de las fases de diseño del algoritmo, combinando el concepto de identificación evolutiva con las metodologías desarrolladas por varios autores: Redes organizadas en sí mismas o GMDH (Ivakhnenko A. G., *et al*., 1998, 1999; Ivakhnenko G.A., 2001), programación genética (Mark and Xin, 2002; Ferreira, 2001, 2004; Nariman-Zadeha and Jamali, 2007) y GMDH-Type con realimentación negativa (Kondo and Ueno, 2006, 2009).

La estructura de este artículo es la siguiente: En la sección 2 se describen las características fundamentales del algoritmo GMDH y la utilización de lazos recurrentes. La sección 3 explica las modificaciones que se hacen en una de las fases del diseño mediante la aplicación de AG. Posteriormente se describen los pasos a seguir para diseñar el modelo propuesto en la sección 4. La sección 5 constituye un ejemplo de aplicación del modelo en una fermentación tipo feed-batch. Estos resultados son comparados con otros algoritmos en la sección 6. Finalmente una sección de conclusiones resume los resultados obtenidos.

## 2. Descripción del Algoritmo GMDH

Las RNA que utilizan el algoritmo GMDH se conocen como Redes Neuronales Polinómicas (RNPo). Las primeras investigaciones en torno a este método fueron hechas por R. Shankar en 1962 el cual presentó el GMDH como un algoritmo que permitía describir de forma sucesiva un sistema complejo de relaciones a partir de simples operaciones matemáticas. De hecho, es un buen método para solucionar problemas del estilo: identificación, predicción a corto y a largo plazo de procesos aleatorios, reconocimiento de patrones en entornos complejos, etc. La teoría matemática fue desarrollada de forma conjunta por muchos investigadores, siendo su máximo exponente Aleksey Gregory Ivakhnenko, hacia los años setenta.

El contenido del algoritmo se desarrolló como vehículo para identificar relaciones no lineales entre entradas y salidas. De esta forma se crea una estructura óptima a partir de un proceso sucesivo de varias generaciones de descripciones parciales de los datos, mediante la incorporación de nuevas capas. El número de neuronas en cada capa, el número de capas y las variables de entrada son determinados automáticamente de forma tal que se minimice un criterio de error de predicción. Así se organiza una arquitectura de red neuronal óptima utilizando una heurística organizada en sí misma, la cual es la base del GMDH (Ivakhnenko A. G., 1971). Este método es particularmente satisfactorio en la solución de problemas de modelación de múltiples entradas para una sola salida (Mutasem, 2004).

### *2.1. Fases de diseño del algoritmo GMDH.*

Los datos utilizados en el diseño de esta red están formados por un conjunto de vectores que agrupan las variables independientes de entrada, ($x_1$, $x_2$, … $x_n$), y la variable de salida S. Antes de iniciar el proceso estos se dividen en dos grupos uno de entrenamiento, utilizado para calcular los parámetros de la red, y otro de prueba que se emplea para evaluar los resultados. El proceso de diseño se divide tres fases fundamentales (Figura 1).

*Primera fase:* Selección de variables de entrada. Consiste en agrupar por parejas todas las posibles combinaciones de variables independientes de la siguiente forma:

$$\{(x_1, x_2), (x_1, x_3), \cdots, (x_1, x_n), \cdots, (x_{n-1}, x_n)\}$$

Siendo *n* el número de variables de entrada. De esta manera se crean n(*n*-1)/2 combinaciones.

*Segunda fase:* Generación de ecuaciones de regresión. Utilizando cada una de las parejas formadas se crean igual número de ecuaciones de regresión. Estas se representan de forma general utilizando la ecuación (1).

$$y = a_0 + \sum_{i=1}^{m} a_i x_i + \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij} x_i x_j \tag{1}$$

Donde: ($a_0$, $a_i$, $a_{ij}$) son los coeficientes o pesos, $x_i$ y $x_j$ constituyen las variables de entrada, siendo $i, j \in \{1, 2, \ldots, n\}$, y *m* representa el orden del polinomio. En nuestro caso de estudio, la ecuación utilizada se reduce a un polinomio de orden 2, por lo que su representación puede simplificarse, quedando de la siguiente forma:

$$y(x_i, x_j) = a_0 + a_1 x_i + a_2 x_j + a_4 x_i x_j + a_3 x_i^2 + a_5 x_j^2 \tag{2}$$

Cada una de las ecuaciones desarrolladas representa una neurona de la red. Así se obtienen tantas ecuaciones de segundo orden como combinaciones de entrada sean posibles. Los coeficientes son calculados aplicando un análisis de regresión empleando el juego de datos de entrenamiento. Mediante el uso de la expresión cuadrática descrita por la ecuación (2) es posible representar la siguiente ecuación matricial para cada par ($x_i$, $x_j$):

$$A * a = Y \tag{3}$$

Donde:

*a* es el vector de coeficientes desconocidos del polinomio cuadrático.

$$a = \{a_0, a_1, a_2, a_3, a_4, a_5\} \tag{4}$$

*Y* es el vector de valores de salida.

$$Y = \{y_1, y_2, y_3, ..., yN\}^{\mathrm{T}} \tag{5}$$

Quedando la matriz *A* representada como:

$$A = \begin{bmatrix} 1 & x_{1i} & x_{1j} & x_{1i}x_{1j} & x_{1i}^2 & x_{1j}^2 \\ 1 & x_{2i} & x_{2j} & x_{2i}x_{2j} & x_{2i}^2 & x_{2j}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{Ni} & x_{Nj} & x_{Ni}x_{Nj} & x_{Ni}^2 & x_{Nj}^2 \end{bmatrix} \tag{6}$$

Aplicando la técnica de mínimo cuadrado se puede alcanzar la solución de las ecuaciones de la siguiente forma:

$$a = \left(A^{\mathrm{T}} A\right)^{-1} A^{\mathrm{T}} Y \tag{7}$$

*Tercera fase:* Selección de las mejores salidas. Se evalúa cada una de las ecuaciones obtenidas empleando el juego de datos de prueba y se determina el Error Cuadrático Medio, MSE, conocido también como Criterio de Regularidad, *CR*

$$CR = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{8}$$

Siendo N el número de vectores que componen el juego de datos de prueba, yn la salida real y ŷn la salida obtenida del modelo.

El valor de *CR* más pequeño $CR_{min}$ es comparado con el valor más pequeño obtenido en la última iteración $CR_{ant}$. En caso de ser la primera iteración (capa de entrada) se adopta un valor mínimo preestablecido $CR_{inic}$. Esto permite determinar la necesidad de más iteraciones o capas. Si no hay mejora en el resultado, o sea, $CR_{min} > CR_{ant}$ o $CR > CR_{inic}$ el proceso ha terminado, quedando como neurona de salida la que obtuvo el $CR_{ant}$. Otra capa es necesaria si al menos 2 neuronas cumplen con $CR < CR_{ant}$, de esta manera se inicia nuevamente el ciclo hasta encontrar una representación optima del proceso. Al pasar nuevamente a la primera fase se utilizan como variables de entrada las salidas $y(x_i, x_j)$ que mejor *CR* obtuvieron en la fase anterior. Aquellas neuronas que no alcanzan un menor criterio de regularidad son rechazadas y sus salidas no son utilizadas en la siguiente capa.

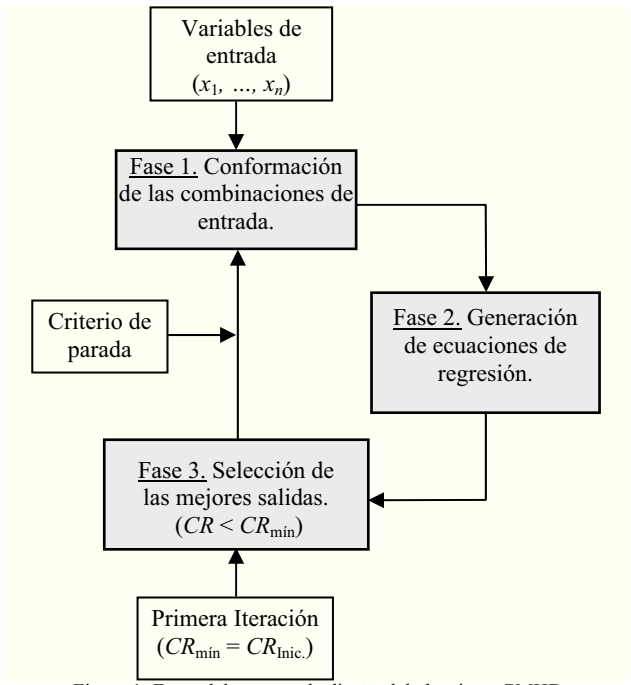



Figura 1: Fases del proceso de diseño del algoritmo GMHD.

La tarea de organizarse en sí misma de las redes de neuronas activas, mediante selección, permite estimar el número de capas y de neuronas activas, así como el conjunto de entradas y salidas potencialmente posibles de cada neurona. Esta característica define el número óptimo de capas y neuronas para cada modelo separadamente. Durante el aprendizaje las neuronas activas organizan en sí mismas la estructura de la red entera. La figura 2 muestra un ejemplo de como se organiza la red tipo GMDH a partir de la selección de aquellas unidades que mejor describen el proceso. La salida *S* corresponde a la neurona que mejor representa el proceso, o sea la que presenta menor *CR*.

Es bien conocido el potencial que poseen las redes neuronales polinómicas organizadas en sí mismas, para la representación de procesos con patrones de comportamiento no lineal, debido a la proliferación de aplicaciones presentes en la literatura. Este tipo de arquitectura es considerada una función de aproximación con buena capacidad de generalización (debido a la superficie del error que genera) y se clasifica dentro del grupo de las redes alimentadas hacia delante (feedforward). Esto significa que las neuronas de una capa solo pueden establecer conexiones con las neuronas de la capa siguiente.

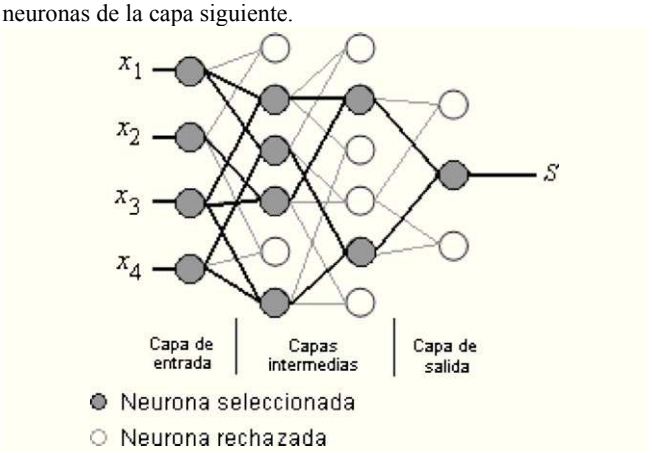



Figura 2: Ejemplo de estructura de red neuronal obtenida mediante algoritmo GMDH.

### *2.2. Algoritmo GMDH recurrente.*

Las Redes Neuronales Alimentadas en Sentido Directo, RNASD, tienen varias limitaciones inherentes a su diseño que pueden ser mejoradas con un cambio de su arquitectura. Sin embargo, existen redes neuronales que permiten establecer simultáneamente conexiones hacia delante y hacia atrás (recurrentes o retroalimentadas), formando ciclos dentro de su arquitectura. De esta manera pueden conservar una memoria interna del comportamiento de los datos, lo que facilita el aprendizaje de relaciones dinámicas complejas (Sánchez, 2008). A estas redes se les conoce como Redes Neuronales Recurrentes, RNR

Las RNR presentan varias características que las hacen superiores a las RNASD. (Drchal, 2006). En primer lugar pueden converger más rápidamente a un valor para una meta de desempeño dada con un determinado margen de error, significando un menor número de iteraciones. Otra ventaja es que tienen un mejor comportamiento no lineal durante el aprendizaje. Sin embargo estas ventajas tienen un costo importante, el tiempo de procesamiento es mayor. Esta característica hace a las RNR más lentas en aplicaciones donde el número de neuronas es grande, tanto en las capas de entrada y salida como en las capas ocultas. De ahí que el uso de esta arquitectura debe ser un compromiso entre desempeño y rapidez.

La modelación usando RNR implica la construcción de dos componentes separados: una o más capas recurrentes, las cuales

almacenan la memoria de corto plazo; y un estimador, que propaga hacia delante el comportamiento aprendido. La memoria a corto plazo retiene las características relevantes de la salida previa a la tarea de estimación; por lo tanto la salida de la red depende no sólo de la entrada actual, sino también de su comportamiento previo.

Para estimar los parámetros del algoritmo GMDH utilizando un lazo recurrente se emplea como variable de entrada realimentada $S_{(t\text{-}1)}$ los valores de salida $S(t)$ del juego de datos de entrenamiento desfasados en el tiempo $t$ en $t$-1. Una vez seleccionadas las neuronas que mejor cumplen con el criterio de regularidad, se evalúa el modelo utilizando el juego de datos de prueba (Kondo, 2003; Kondo and Ueno, 2006). Dicha metodología solo puede establecer un lazo recurrente entre la neurona de salida y la capa de entrada. Sin embargo, por la naturaleza de las variables que se utilizan en las capas intermedias, se puede establecer una relación a corto plazo con la salida en el instante de tiempo anterior ($t$-1) lo que ayuda a una mejor estimación. Esto implica el uso de lazos recurrentes en las capas intermedias. Para lograr este objetivo resulta necesario modificar la fase 1 del proceso de diseño de manera que se amplíe el área de regresión y se pueda incorporar una entrada $S_{(t\text{-}1)}$ entre las variables de las capas intermedias.

## 3. Modificación de la Fase 1 del Diseño de una Red GMDH Utilizando Algoritmo Genético

Como se había explicado anteriormente esta fase está vinculada con la formación de las combinaciones de entrada de cada capa, aspecto que se necesita incrementar incorporando nuevas variables. Ante tal situación los modelos evolutivos constituyen una alternativa muy utilizada.

### *3.1. Codificación genética.*

Los métodos evolutivos, como el algoritmo genético, han sido muy utilizados en diferentes aspectos del diseño de las redes neuronales. Algunos métodos estocásticos son comúnmente usados en su entrenamiento en la determinación de pesos asociados o coeficientes, con resultados satisfactorios que superan los métodos tradicionales basados en el gradiente. La literatura muestra que un amplio número de diseños evolutivos son usados en la determinación de la arquitectura o en la obtención de los pesos de conexión, de manera separada. Aunque se realizan esfuerzos por obtenerlos simultáneamente.

En la mayoría de los algoritmos GMDH se alcanza una arquitectura definida por el propio modelo, donde las neuronas de cada capa solo se pueden conectar con las neuronas de la capa adyacente. Tomando esto como ventaja, es posible obtener un simple esquema de codificación para el genotipo de cada individuo de la población. Sin embargo para representar el modelo de una manera más general es necesario eliminar esta restricción en su arquitectura (Bagheri, *et al*., 2007). De manera que las neuronas de una capa intermedia o de la capa de entrada puedan conectarse con una neurona lejana, que no solo esté ubicada en la capa adyacente (Figura 3).

La codificación genética de las redes neuronales tipo GMDH, en función de desarrollar su estructura, comienza representando cada neurona como una cadena alfabética de entradas. La habilidad de cambiar bloques de información que provoca el cruzamiento o mutación de cromosomas puede ser utilizada para formar nuevas generaciones. Tal como aparece descrito por Nariman-Zadeh, *et al*. (2003).

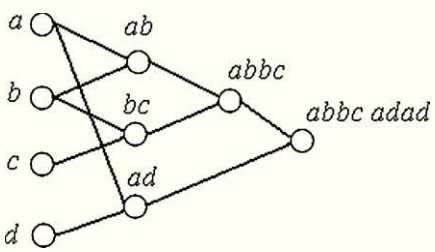
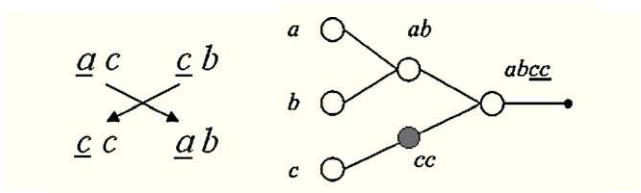



Figura 3: Ejemplo de codificación donde se representa una neurona virtual.

Las conexiones neuronales en la red GMDH, generalizada por este método, pueden ocurrir entre diferentes capas, las cuales no son necesariamente adyacentes, a diferencia de la red neuronal GMDH convencional. Por ejemplo, una distribución de red como la que aparece dibujada en la figura 3, muestra la conexión de una neurona intermedia directamente con la neurona de salida. Consecuentemente, esta generalización de la estructura extiende el desarrollo de las redes GMDH en la modelación de procesos complejos del mundo real. Tal generalización se logra repitiendo el nombre de la neurona, la cual pasa directamente a la siguiente capa. En esta figura la neurona *ad* de la primera capa oculta es conectada a la capa de salida, pasando directamente a través de la segunda capa oculta. De esta forma es fácil notar que el nombre de la neurona de salida incluye *ad* dos veces *abbcadad*. En otras palabras una neurona virtual llamada *adad* ha sido construida en la segunda capa oculta y usada junto con *abbc* como entrada de la neurona de salida (Nariman-Zadeha, *et al*., 2005). Tal repetición ocurre siempre que una neurona pase algunas capas ocultas adyacentes y se conecte a otra neurona en la 2da, 3ra, 4ta o enésima capa oculta. En este esquema de codificación el número de repeticiones de la neurona depende del número de capas ocultas por donde pasa, $\tilde{n}$, y es calculada como $2^{\tilde{n}}$. Ello significa, por ejemplo, que si una neurona *a* pasa directamente por dos capas, su representación a la entrada de la tercera capa sería *aaaa*. Resulta evidente, además, que si se formara un cromosoma virtual como el *abab* combinado con otro cromosoma virtual, como por ejemplo *bcbc* se genera un cromosoma *abab bcbc*, lo que puede representarse como *abbc* de forma simplificada. Esto demuestra que dos neuronas virtuales no pueden combinarse para formar una neurona en la capa adyacente, pues su combinación genera un cromosoma no válido para dicha capa.

Si llevamos este método a una sola capa intermedia, aplicando operadores genéticos de cruzamiento y mutación se puede simplificar su representación y obtener nuevas combinaciones que incrementen el número de ecuaciones utilizadas en dichas capas.

En la figura 4 se puede observar otro ejemplo donde existen tres entradas *a*, *b* y *c*. De ellas se pueden obtener las combinaciones *ab*, *ac* y *bc* por el método clásico. Al aplicar el operador de cruzamiento sobre los padres *ac*, *bc* se obtiene una descendencia formada por *ab* y *cc*. La combinación *ab* ya formaba parte del dominio de combinaciones posibles. Sin embargo, tal y como se había descrito anteriormente, resulta evidente la formación de una neurona virtual *cc*. De esta forma la entrada *c* queda conectada directamente con la neurona de salida.




(a). Cruzamiento (b). Representación de la Neurona Virtual *cc*.

Figura 4: Ejemplo de neurona virtual obtenida por cruzamiento.

El método natural de la ruleta puede ser utilizado como método para seleccionar dos padres que formarán dos descendencias. Sin embargo, tomando en cuenta que existe un número finito de combinaciones, es posible conocer entonces el número máximo de descendencia y por consiguiente el número de neuronas virtuales, cuyo valor coincide con la cantidad de entradas. Siendo esta una representación generalizada del proceso de diseño se le conoce como algoritmo GMDH generalizado. Para comprender mejor este procedimiento se puede observar en la figura 5 como las variables de entrada se reflejan directamente sobre el conjunto de variables que pasan a la siguiente capa.

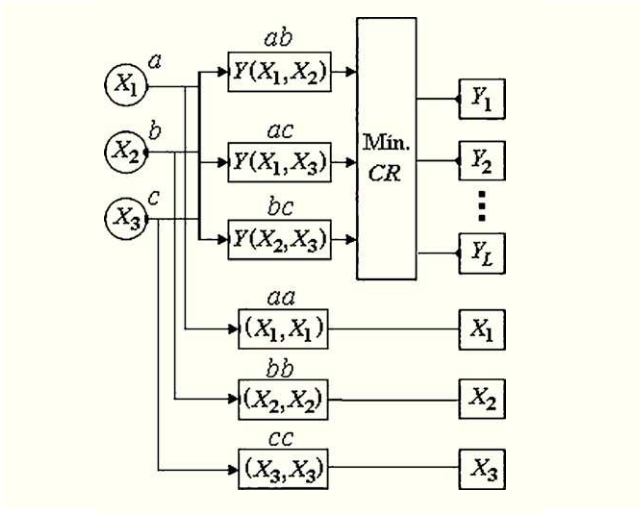



Figura 5: Diagrama de conformación del conjunto de variables que pasan a la siguiente capa.

### *3.2. Forma generalizada de representación.*

Las redes neuronales pueden ser descritas como una matriz, con unidades de neuronas activas en varias capas. Las neuronas de cada capa difieren unas de otras por sus conjuntos de variables de entrada y de salida. Las variables de salida de cada capa de neuronas activas son usadas como variables de entrada de la siguiente capa. La extensión del área de regresión siempre perfecciona el resultado de la misma (Ivakhnenko G.A., 2001). En la red neuronal considerada a continuación dicha extensión es obtenida de manera sencilla, tomando en cuenta los métodos evolutivos planteados anteriormente.

Por ejemplo, si la primera capa de neuronas activas contiene un conjunto de variables de entrada ($X_1$, $X_2$,…, $X_M$) y estas se combinan entre sí formando ecuaciones que generan un conjunto de variables de salida ($Y_1$, $Y_2$,…, $Y_L$) entonces las neuronas de la segunda capa obtienen ambos conjuntos como entrada, tal como se observa en la figura 6. Esto se debe a la formación de neuronas virtuales, {($X_1$, $X_1$), ($X_2$, $X_2$),…, ($X_M$, $X_M$)} derivadas de la aplicación de operadores genéticos sobre las combinaciones de entrada, tal como se observa en la figura 5. Este proceso se repite en cada una de las capas siguientes.

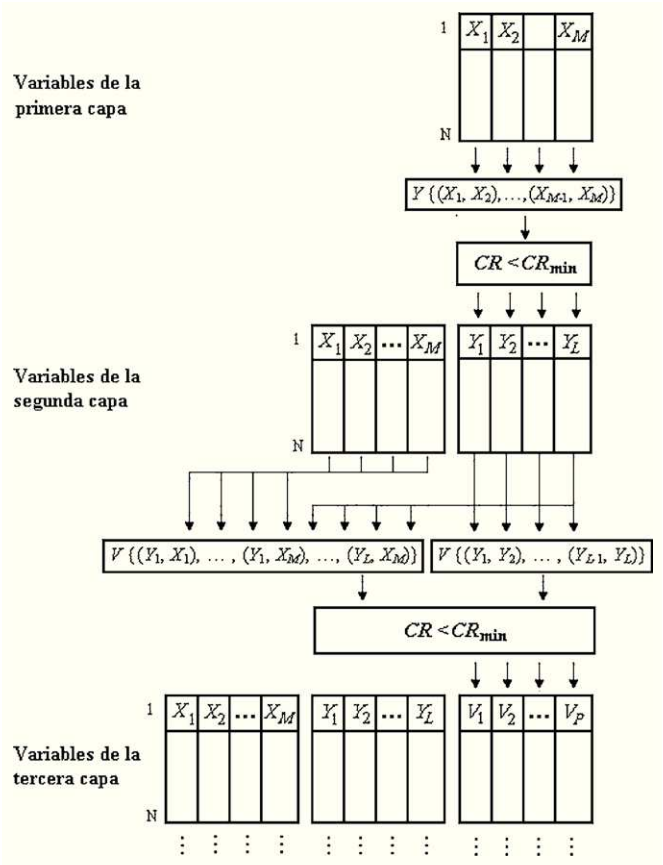



Figura 6: Representación generalizada del diseño de una red neuronal GMDH.

La extensión del conjunto de variables y con ello del área de regresión siempre debe estar acompañada de un estrechamiento razonable del número de combinaciones para evitar exceder las habilidades computacionales. Con este objetivo se definen tres reglas para la formación de combinaciones tomando en cuenta el análisis realizado con operadores genéticos y la organización en sí misma del algoritmo GMDH. Ellas son:

1. Las salidas de la capa anterior que son seleccionadas por CR para pasar a la capa siguiente, generan un conjunto de variables que se combinan entre sí para formar pares que integran las ecuaciones de regresión.
2. Las entradas de la capa anterior pasan directamente a formar parte del conjunto de entradas de la siguiente capa.
3. El conjunto de entradas derivadas de las entradas de la capa anterior no pueden combinarse entre sí, pues formarían una población que no pertenece al dominio de esta capa. Solo pueden combinarse con las salidas de la capa anterior

Estas reglas reducen el número de combinaciones posibles, *k*, del conjunto de variables de entrada. Partiendo del ejemplo que se muestra en la figura 6, el valor de *k* para el conjunto de variables de la segunda capa puede determinarse de la siguiente forma:

$$k = L(L-1)/2 + L*M \quad (9)$$

Donde $L$ es el número de salidas de la capa anterior seleccionadas por el criterio de regularidad para pasar a la siguiente etapa, y $M$ es el número de entradas de la capa anterior.

De la figura 6 resulta evidente que el número de combinaciones de entrada de cada capa aumenta de una a otra debido a que el conjunto de variables está formado por las salidas seleccionadas y las entradas de la capa anterior. Para evitar un sobredimencionamiento de la red es necesario imponer dos criterios de parada. Ellos son:

- Cuando se ha alcanzado un valor mínimo de *CR*.
- O cuando se ha alcanzado un número máximo de capas *NC*.

Una vez concluido este proceso queda definida la arquitectura de la red por selección de aquellos regresores que mejor representan el proceso. La utilización de un lazo recurrente entre las variables de entrada, bajo estas condiciones, permite que la señal retroalimentada se represente en todas las capas intermedias. De esta manera se puede lograr un lazo recurrente en cada capa como se muestra en el ejemplo de la figura 7.

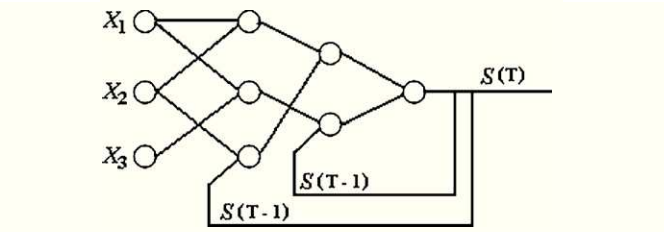



Figura 7: Ejemplo de formación de un lazo recurrente en cada capa intermedia.

## 4. Pasos para el Diseño del Modelo

Antes de iniciar el proceso de diseño se establecen los valores iniciales de la variable de realimentación $S_{(t-1)}$ para entrenamiento y prueba, siendo iguales a la salida $S$ del juego de datos de entrenamiento y de prueba respectivamente, desfasados en un instante atrás. En la figura 8 puede observarse un diagrama detallado del algoritmo propuesto.

Para el diseño de este modelo se establecen los siguientes pasos:

*Paso 1:* Establecer los criterios de parada *CR* mínimo inicial y el número máximo de capas *NC*.

*Paso 2:* Introducir lazo recurrente. Agregar la salida $S$ desfasada en ($t$-1) al conjunto de entradas.

*Paso 3:* Inicializar los operadores de diseño:

Contador de capa $q = 1$.

No. de entradas de la capa anterior $M = 0$.

No. entradas de la capa inicial $L$.

*Paso 4:* Generar todas las combinaciones de entradas posibles siguiendo el modelo generalizado del GMDH.

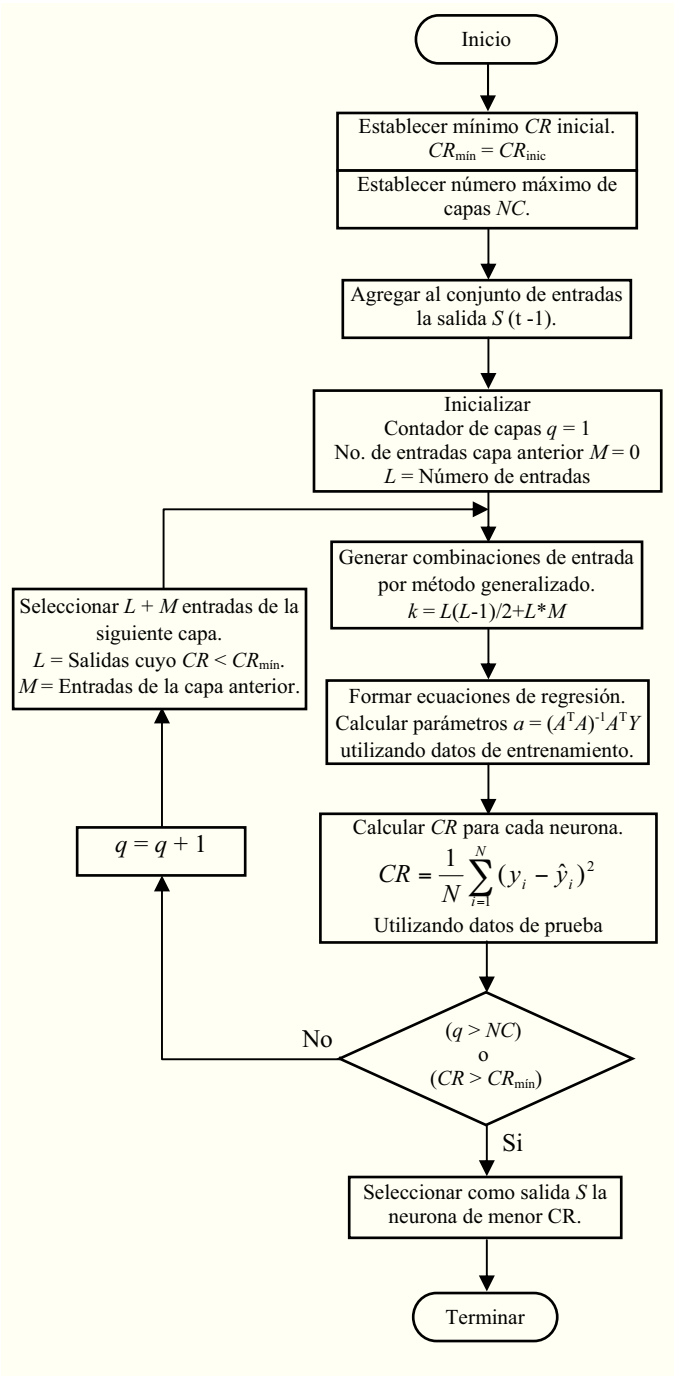
*Paso 5:* Formar las ecuaciones de regresión y calcular sus parámetros utilizando los datos de entrenamiento según ecuación (7).

*Paso 6:* Calcular *CR* para cada unidad neuronal según la ecuación (8).

*Paso 7:* Evaluar los criterios de parada. Si se cumple que ($q > NC$ ) o ($CR > CR_{min}$) el proceso termina quedando como salida la neurona que alcanzó menor *CR*. En caso contrario el proceso continúa al *Paso 8*.

*Paso 8:* Incrementar el contador de capas $q = q + 1$.

*Paso 9:* Seleccionar las entradas de la siguiente capa. Este conjunto está formado por las $L$ salidas cuyo $CR < CR_{min}$ y las $M$ entradas de la capa anterior. Para formar las combinaciones y repetir así el proceso de diseño de la siguiente capa saltar al *Paso 4*.



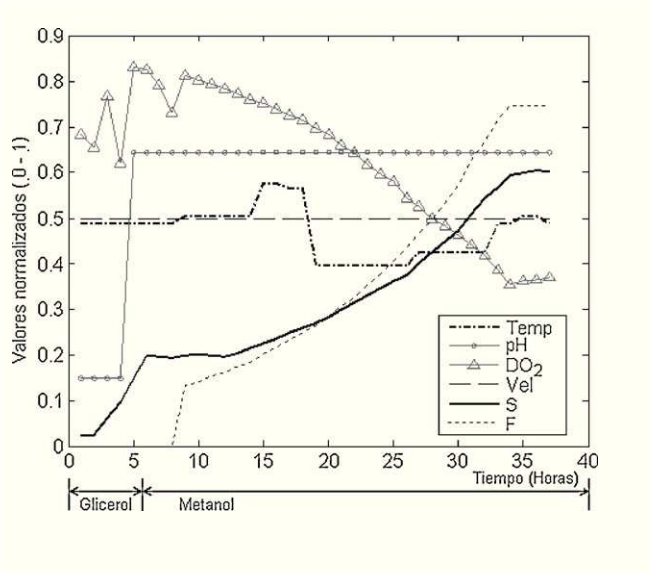
Una vez terminado el proceso, la neurona de menor CR será la salida S y el resto de la red puede obtenerse a partir de los enlaces que de ella se derivan.

## 5. Aplicación Práctica en la Modelación de un Proceso de Fermentación

### 5.1. Descripción del proceso.

Obtener una alta concentración de biomasa es uno de los principales objetivos de la producción biotecnológica. La medición de este parámetro permite controlar el crecimiento celular y optimizar así el proceso. La determinación exacta, continua y en línea (on-line) de la concentración de biomasa, es uno de los "principales sueños" en la biotecnología. Entre los métodos más comunes para determinar la concentración de la biomasa están la extracción de muestras para gravimetría (peso seco de células) o bien la prueba de espectrometría (densidad óptica) (Reed, *et al*., 2000; Känsäkoski, *et al*., 2006). Dichos métodos no permiten acciones de control oportunas ya que en la mayoría de los casos se realizan fuera del proceso (off-line), implicando esto: pérdida de densidad de información, retraso en la obtención de resultados y esfuerzo humano (Royce, 1993). Por esta razón se requiere de un sistema de estimación robusto y con una buena capacidad de generalización, debido a que las fermentaciones nunca son iguales, aun teniendo un control estricto de las condiciones iniciales, substrato e instrumentación (Jenzsch, *et al*., 2006). La utilización de modelos constituye una alternativa de importante aplicación en estos casos.

Los datos que se utilizan para implementar la red neuronal propuesta se tomaron de un proceso de fermentación tipo feed-batch para la producción de una vacuna recombinante utilizando la levadura *Pichea pastoris*, la cual cuenta con dos etapas en su crecimiento, atendiendo al suministro de substrato. La primera etapa del crecimiento celular se produce en presencia de glicerol, donde no se le agrega substrato y la segunda se desarrolla en metanol donde existe un flujo controlado de adición de substrato en correspondencia con la razón de crecimiento. Este cambio de medio produce una variación en la pendiente de crecimiento, tal como se muestra en la figura 9, donde el comportamiento de la biomasa es descrito por la curva *S*.




Las variables que se miden en el proceso son: Flujo de substrato (*F*), Temperatura (*Temp*), pH, Oxígeno disuelto ($DO_2$) y Velocidad de agitación (*Vel*) (McNeil and Harvey, 1990). Considerando $S_{(t-1)}$ el valor de salida realimentado un paso atrás, entonces las variables de entrada quedan expresadas de la siguiente forma:

$x_1 = F$ $x_2 = Temp$ $x_3 = pH$
$x_4 = DO_2$ $x_5 = Vel$ $x_6 = S_{(t-1)}$

La concentración de biomasa es obtenida utilizando la técnica de peso húmedo, determinado por métodos analíticos fuera del proceso (*off-line*). Tomando en consideración el tiempo existente entre cada muestra utilizada, el número de datos queda reducido a 24 muestras por fermentación.

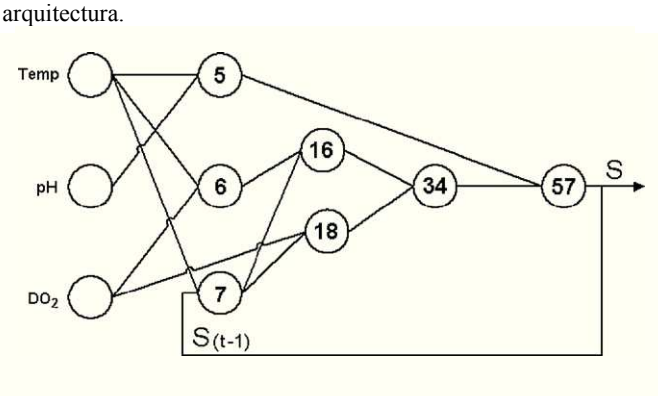
### 5.2. Diseño de la red.

Después de seleccionar las variables de entrada, estableciendo una realimentación $S_{(t-1)}$, se separaron los datos de entrenamiento y prueba respectivamente. Como criterio de parada inicial se adoptaron los siguientes valores: $CR_{inic}$ = 0.01 y *NC* = 5. A continuación se formaron todas las combinaciones posibles de entrada utilizando las reglas definidas en el método propuesto. El diseño se realizó siguiendo los pasos descritos anteriormente. La figura 10 representa la estructura de red obtenida. En ella solo aparecen representadas aquellas neuronas que fueron seleccionadas por cumplir con el criterio de regularidad. En esta figura se puede observar como se produce un enlace directo entre la neurona 5 y la 57 pasando por dos capas intermedias. Como se observará a continuación parte de los resultados obtenidos por la red se deben a esta característica.

Tabla 1: Coeficientes de cada neurona

| Ne | $a_0$ | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ |
|---|---|---|---|---|---|---|
| 5 | 3.22E-01 | 1.25E+00 | -8.59E-01 | -1.22E+00 | 0.00E+00 | 6.08E-01 |
| 6 | 1.62E-03 | 5.50E-01 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |
| 7 | 1.62E-02 | -6.57E-02 | 1.56E+00 | -5.11E-01 | 0.00E+00 | -2.85E-01 |
| 16 | -9.69E-04 | 1.42E-02 | 1.00E+00 | -6.82E-02 | 0.00E+00 | 3.08E-02 |
| 18 | 1.08E-03 | 9.92E-01 | 0.00E+00 | 0.00E+00 | 1.11E-02 | 0.00E+00 |
| 34 | 2.79E-02 | 4.57E+01 | -4.49E+01 | -3.28E+04 | 1.63E+04 | 1.65E+04 |
| 57 | 1.73E-04 | 9.57E-01 | 4.35E-02 | 9.35E-01 | -3.95E-01 | -5.41E-01 |

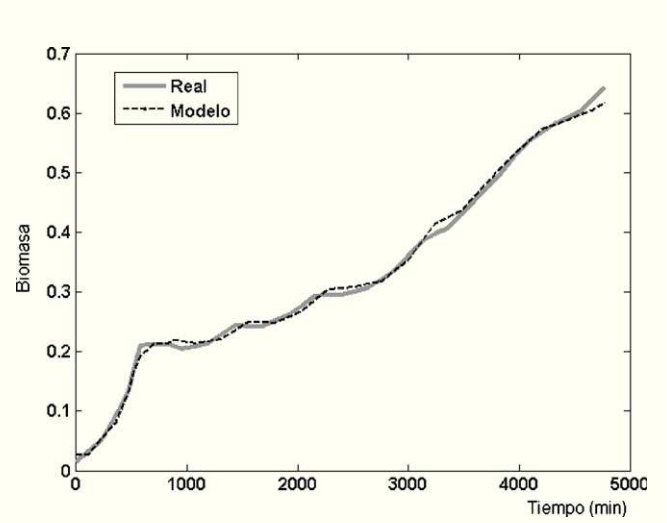
La arquitectura mostrada corresponde solamente al proceso de fermentación utilizado como ejemplo. La aplicación de este método en un nuevo proceso puede implicar una diferencia en su arquitectura.



Las variables de entrada que no aparecen en el modelo fueron rechazadas durante el propio proceso de diseño. En la tabla 1 aparecen reflejados los coeficientes de las ecuaciones que describen cada una de las neuronas que se observan en la figura 10.

### *5.3. Análisis de los resultados.*

La figura 11 muestra el comportamiento de la red durante un proceso de fermentación donde se observa un desempeño satisfactorio del modelo. Existen diferencias entre un proceso de fermentación y otro aun cuando se trate del mismo microorganismo con iguales condiciones. La estabilidad en la modelación y su capacidad para asimilar las no linealidades propias de los procesos de fermentación puede apreciarse en la figura 12, donde, a pesar de existir características peculiares que lo distinguen del ejemplo de la figura 11, esta red es capaz de seguir su comportamiento.



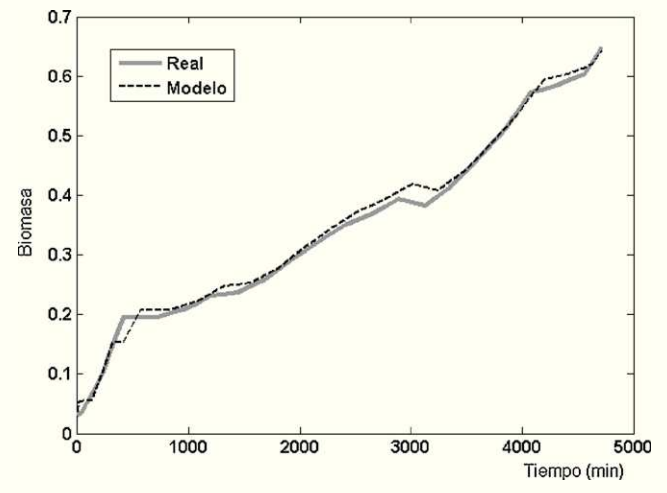




Figura 12: Respuesta del modelo ante otro proceso con características peculiares.

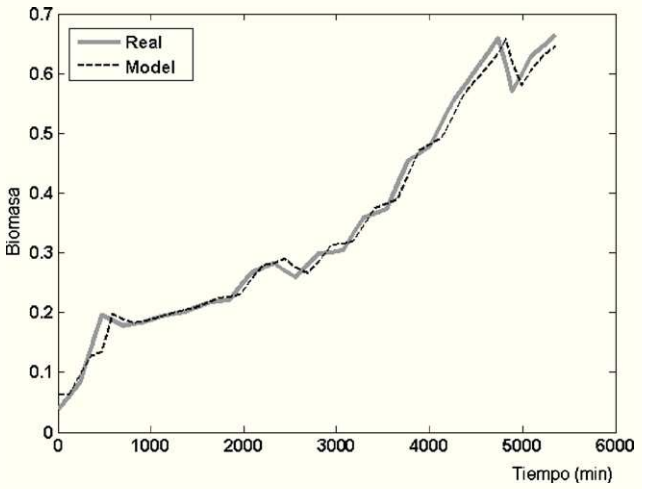



Figura 13: Respuesta del modelo en presencia de ruido en una de sus variables de entrada.

La estabilidad que se aprecia en el modelo le permite absorber las no linealidades y las diferencias existentes en un mismo proceso. Este aspecto posibilita extender su uso como herramienta para estimar el crecimiento celular en un birreactor en tiempo real y resolver así la ausencia, en muchos casos, de sensores con este fin. A pesar de la relativa complejidad de la red, el crecimiento celular es un proceso lento en comparación con su tiempo de estimación. Aun cuando existan variaciones bruscas de las condiciones ambientales, la razón de cambios en la biomasa dista mucho de las potencialidades existentes hoy en día en los medios de cómputo utilizados para estos fines. Por otra parte se incorpora al algoritmo un criterio de parada que permite regular su costo computacional a partir de definir un número máximo de capas. Otro aspecto a tener en cuenta para valorar su uso en procesos reales es que en ellos el diseño o reajuste de la red se realiza, en la mayoría de los casos, fuera del proceso ya que, generalmente, no se cuenta con sensores en línea midiendo el crecimiento celular. Estos datos se obtienen, en la mayoría de los casos, mediante técnicas analíticas y sus resultados son utilizados posteriormente en el entrenamiento de la red.

Solo es necesario tener en cuenta un aspecto muy importante y es que cuando se realizan cambios en el protocolo o se desarrolla un nuevo producto es necesario entrenar la red con las nuevas características. Generalmente cada vez que se efectúan estos cambios se realizan fermentaciones de estandarización que pueden ser utilizadas para entrenar la red que posteriormente podrá ser utilizada en este proceso.

## 6. Comparación con Otros Algoritmos

Para poder establecer una medida del comportamiento del algoritmo descrito, es necesario realizar una comparación con sus antecesores. En este caso el GMDH típico de segundo orden utilizando un lazo recurrente puede servir como referencia para evaluar su desempeño. Un criterio muy empleado en la comparación entre diversos algoritmos es el error cuadrático medio (mse) (Ramsey, 1994; Pappas and Ekonomou, 2006). La figura 14 muestra el comportamiento de mse de ambos sistemas durante el proceso de entrenamiento.

Una prueba más evidente de su estabilidad se aprecia al introducir ruido blanco en una de sus variables de entrada

utilizando un proceso con marcadas diferencias respecto a las primeras, como se aprecia en la figura 13.

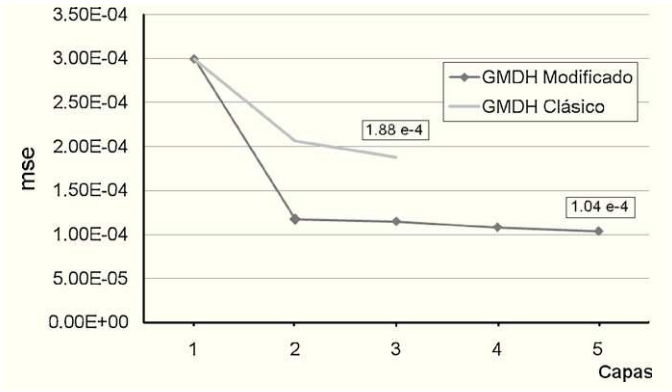



Figura 14: Convergencia del error durante el entrenamiento.

Como puede apreciarse el algoritmo típico alcanza su mínimo error en la tercera capa. Esto se debe a que el método no encuentra un error menor en la siguiente capa. Sin embargo, el otro método alcanza un menor error en la quinta capa debido a que se incrementó el número de variables utilizadas a la entrada de cada capa y con ello el número de combinaciones, lo que en consecuencia amplía el área de regresión y las probabilidades de alcanzar una representación óptima del proceso. Es notable, además, que desde la primera regresión el error que se alcanza en el modelo generalizado es menor que el típico, con lo que se puede afirmar que el algoritmo converge hacia un menor error más rápidamente.

Por otra parte existe una diferencia interesante entre ambos modelo, dada por la forma en que se ha detenido el proceso de diseño. El modelo típico alcanzó más rápidamente su valor mínimo por lo que resulta imposible para este algoritmo continuar adicionando más capas que le permitan reducir aun más el error. Mientras que el método generalizado se detuvo tras alcanzar el máximo número de capas. Esto significa que si se eleva este valor el modelo puede continuar convergiendo hacia un error menor hasta tanto alcance un valor mínimo. Lo cual genera un compromiso, pues al elevarse el número de capas se incrementa también, el costo computacional. Es necesario tener en cuenta además que en ocasiones la disminución del error que se produce entre una capa y otra es muy pequeña por lo que no reporta una ventaja considerable continuar elevando la complejidad de la red.

La figura 15 muestra los resultados a la salida del modelo obtenido por el algoritmo GMDH típico con lazo recurrente. En ella puede observarse una ligera separación del modelo con respecto al comportamiento real en comparación con los resultados que se aprecian en la figura 11.

En la figura 16 puede observarse una comparación del modelo propuesto y el típico con otros métodos. Entre ellos la red neuronal de Elman que contiene un lazo recurrente, al igual que los algoritmos GMDH analizados anteriormente y una red neuronal feed-forward clásica. Para el proceso utilizado como caso de estudio, resulta evidente que el algoritmo GMDH con realimentación negativa propuesto, tiene mejor desempeño al mostrar un menor error que el resto de las variantes utilizadas.

Las redes neuronales desarrolladas utilizando el algoritmo GMDH, tienen una importante diferencia respecto a los modelos cuya distribución neuronal está definida previamente y es que sus unidades de procesamiento juegan un papel activo en la definición de su estructura. Esto se debe a que el diseño es ejecutado dentro de la propia unidad, representando una nueva variable que es generada por la selección independiente de las entradas más importantes, necesarias para encontrar una distribución óptima (Balanza, *et al*., 1998). Este aspecto influye en los resultados obtenidos en la figura 16 donde se puede apreciar que el algoritmo GMDH presenta un error pequeño en ambos casos, ya que está optimizado en base a este criterio.

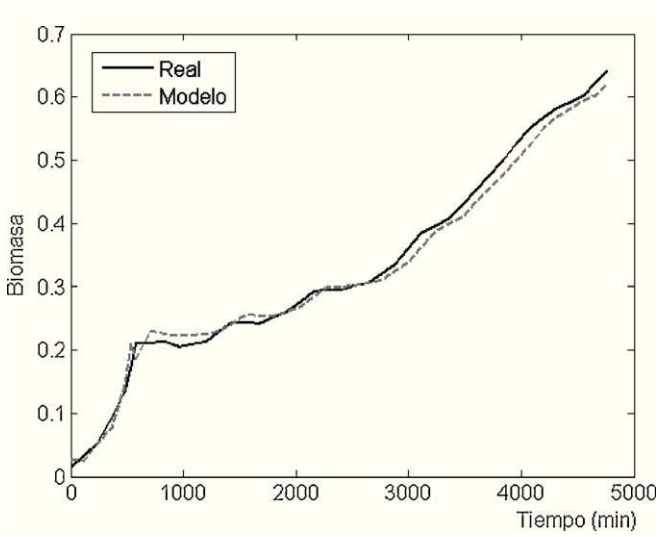



Figura 15: Modelo con algoritmo GMDH típico con lazo recurrente.

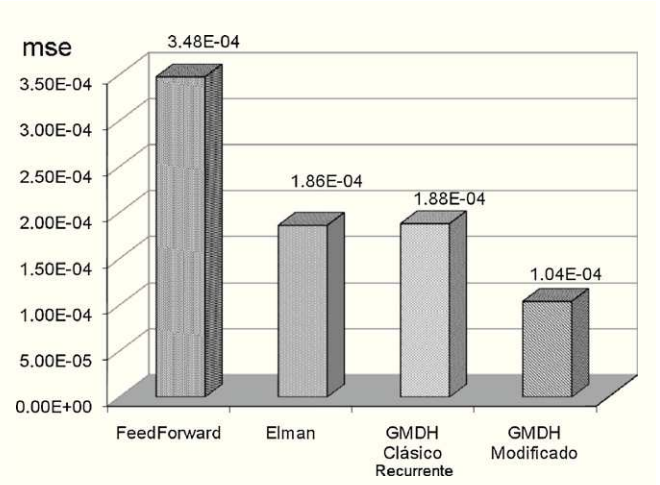



Figura 16: Comparación aplicando diferentes modelos neuronales al caso de estudio.

## 7. Conclusiones

En este trabajo se ha desarrollado una red neuronal utilizando el algoritmo GMDH, proponiendo un método para incrementar el número de combinaciones de entrada de cada capa intermedia, modificando una de las fases del proceso de diseño. Los aspectos abordados anteriormente demuestran su capacidad para estimar el crecimiento celular en un proceso de fermentación de tipo feed-batch. Esto permite la implementación de un sensor virtual (Soft-Sensor) capaz de estimar, en línea (On-Line), la concentración de

biomasa y poder establecer, de esta forma, un control oportuno de las variables para mejorar sus resultados.

La complejidad en el proceso de diseño se incrementa gradualmente debido al aumento del número de variables a utilizar en cada capa intermedia. Esto interviene directamente en su estructura final, permitiendo la utilización de más capas, con lo cual se logra una disminución del error. Aun cuando se incrementa su costo computacional sus capacidades para el uso en tiempo real en procesos de fermentación siguen siendo buenas, ya que los mismos no contienen variables críticas en el tiempo.

Al ser comparado con otros métodos neuronales, puede observarse un mejor desempeño en la modelación de un proceso de fermentación por lotes.

El uso de lazos recurrentes combinados con métodos de algoritmos genéticos, aplicados al diseño de redes neuronales mediante el algoritmo GMDH, abre nuevos horizontes. Con ello se puede hablar no solo de un lazo recurrente de salida hacia la entrada, sino de múltiples lazos interiores, lo que permite definir neuronas capaces de enlazarse en varias direcciones hacia delante y hacia atrás. De esta forma se puede desarrollar sistemas complejos con mayores potencialidades para la identificación de procesos no lineales como es el caso de los procesos de fermentación.

## English Summary

**Intelligent identification of a fermentative process using modified GMDH Algorithm.**

### Abstract

One of the variables of more interest in the biotechnological processes is the biomass concentration. The continuous, on-line and exact determination of this parameter it is very difficult and expensive. In this work a neural network was used to estimate the biomass concentration in a Feed-batch fermentation process. In the design, the Group Method of Data Handling algorithm, GMDH, is applied with a new structure based on the employment of genetic algorithm and incorporating a feedback loop. The neural equations are restricted to order 2. The pattern was proved in the fermentation of the Pichia pastoris yeast for the production of a vaccine for recombinant methods. The stability and generalization capacity is demonstrated. The proposed method was compared with other neuronal networks attending to behavior of the Mean Square Error. (mse).

*Keywords:*

Neural networks, recurrent, Genetic Algorithms, modeling, fermentation.

## Agradecimientos

Este trabajo ha sido realizado gracias al apoyo del Centro de Ingeniería Genética y Biotecnología de Camagüey y el departamento de Automática de la Facultad de Ingeniería Eléctrica de la Universidad Central de las Villas.

## Referencias

Bagheri A., Nariman-Zadeh N., Babaei M., and Jamali A., 2007. Polynomial Modeling of the Controlled Rack-Stacker Robot Using GMDH-type Neural Networks and Singular Value Decomposition. International Journal of Nonlinear Sciences and Numerical Simulation, 8(3), 301-310.

Balanza García José, Cano I. J. M., López C. J., Alvarez C. A., Luna M. M., 1998. Estudio y Desarrollo de Sensores Software Basados en Sistemas Neuro-Difusos. Aplicación en Procesos Petroquímicos. Departamento de Ingeniería de Sistemas y Automática de la UPCT. TIC99-0446-C02-01.

Drchal, J., 2006. Evolution of Recurrent Neural Networks., Czech Technical University, Faculty of Electrical Engineering, Prague.

Ferreira, C., 2001. Gene Expression Programming in Problem Solving. Complex Systems.

Ferreira, C., 2004. Designing Neural Networks Using Gene Expression Programming. Paper presented at the 9th Online World Conference on Soft Computing in Industrial Applications.

Happel, B. L. M. and Murre, J. M. J., 1990. Structure Identification of Non Linear Dynamic Systems - A survey on input/output Approaches. Automática, 26(4), 651-667.

Holland, J. H., 1975. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. University of Michigan Press, (second edition: MIT Press, 1992).

Ivakhnenko A. G., K. V. V., Tetko I. V., Luik A.I., Ivakhnenko G.A., Ivakhnenko N.A., 1999. Self-Organization of Neural networks with Active Neurons for Bioactivity of Chemical Compounds Forecasting by Analogues Complexing GMDH Algorithm. Paper presented at the Poster for the ICANN'99 Conference.

Ivakhnenko A.G., 1971. Polynomial theory of complex systems. IEEE Transactions on Systems, Man, and Cybernetics, SMC-1(1).

Ivakhnenko A.G., D. W., Ivakhnenko G.A., 1998. Inductive Sorting-Out GMDH Algorithms with Polynomial Complexity for Active Neurons of Neural Network. from http://come.to/GMDH

Ivakhnenko G.A., 2001. Inductive Self-Organizing Algorithm for Maximum Electrical Load Prediction. International Centre of Informational Technologies and Systems of the National, Ac. Sci. Ukrania, Kyiv.

Jenzsch, M., Simutis, R., Lübbert, A., 2006. Optimization and Control of Industrial Microbial Cultivation Processes. Eng. Life Sci., 6(2), 117-124.

Känsäkoski, M., Kurkinen, Marika, von Weymarn, Niklas, Niemelä, Pentti, Neubauer, Peter, Juuso, Esko, Eerikäinen, Tero, Turunen, Seppo, Aho, Sirkka & Suhonen, Pirkko., 2006. Process analytical technology (PAT) needs and applications in the bioprocess industry. VTT Technical Research Centre of Finland, 60, 99.

Kondo, T., 2003. Revised GMDH-type neural networks with radial basis functions and their application to medical image recognition of stomach. Systems Analysis Modeling Simulation, 43(10), 1363-1376.

Kondo, T. and Ueno, J., 2006. Revised GMDH-Type Neural Network Algorithm With a Feedback Loop Identifying Sigmoid Function Neural Network. International Journal of Innovative Computing, Information and Control, 5(2), 985—996.

Kondo, T. and Ueno, J., 2009. Medical Image Recognition of Abdominal Multi-Organs by Rbf Gmdh-Type Neural Network. International Journal of Innovative Computing, Information and Control, 5(1), 225-240.

Leiva, G. A. (2006). Redes Neuronales como Herramienta para la Automatización de Sistemas Complejos. Paper presented at the EVIC2006.

Mark S. Voss, Xin Feng., 2002. A new methodology for emergent system identification using Particle Swarm Optimization (PSO) and the Group Method of Data Handling (GMDH). Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2002.

McNeil B., Harvey L. M., 1990. Fermentation a Practical Approach (1ra ed.). Oxford: Oxford University Press.

Miroslav Šnorek, P. K., 2006. Inductive Modelling World Wide the State of the Art. Report of investigation, Dept. of Computer Science and Engineering, Karlovo nam.

Mutasem Hiassat, N. M., 2004. An evolutionary method for term selection in the Group Method of Data Handling. Automatic Control & Systems Engineering, University of Sheffield, 11-14.

Nariman-Zadeh, N., Darvizeh, A., Ahmad-Zadeh, R., 2003. Hybrid Genetic Design of GMDHType Neural Networks Using Singular Value Decomposition for Modelling and Prediction of the Explosive Cutting Process. Journal of Engineering Manufacture, 217, 779-790.

Nariman-Zadeha N., Darvizeha A., Jamali A., Moeinib A., 2005. Evolutionary design of generalized polynomial neural networks for modeling and prediction of explosive forming process. Paper presented at the 13th International Scientific Conference on Achievements in Mechanical and Materials Engineering.

Nariman-zadeha N., Jamali A., 2007. Pareto Genetic Design of GMDH-type Neural Networks for Nonlinear Systems: Department of Mechanical Engineering, University of Guilan.

Pappas S. Sp., Ekonomou L., 2006. Comparison of Artificial Intelligence Methods for Predicting the Time Series Problem. Proceedings of the 6th WSEAS International Conference on Simulation, Modelling and Optimization, 22-24.

Passoni, L. I., 2005. Modelos en Bioingeniería: Caracterización de Imágenes Estáticas y Dinámicas. Tesis del Doctorado en Ingeniería, Universidad Nacional de Mar del Plata.

Ramsey, A., 1994. Assessment of the modeling abilities of Neural networks. U. of Massachusetts, US.

Reed G., Rehm J., Puhler A., Stadler P., 2000. Biotechnology, Measuring modelling and control. VHC, 4, 181.

Royce, P. N., 1993. A discussion of recent developments in fermentation monitoring and control from a practical perspective. Critical reviews in biotechnology, 13, 117-149.

Sánchez, P. A., 2008. Redes Neuronales Recurrentes en el Modelado de la Tasa de Cambio Colombiana. III Congreso Colombiano de Computación. Abril 23-25 Medellín.

Solla, S. A., 1989. Learning and Generalization in Layered Neural Network: the Contiguity Problem. Neural Networks: from Models to Aplications. In L. Personnas and G. Dreyfus Paris: I.D.S.E.T.

Yuan J.Q., Vanrolleghem P. A., 1999. Rolling learning-prediction of product formation in bioprocesses. Journal of Biotechnology (Elsevier Science B.V.), 69, 47–62.