UNIVERSIDAD MICHOACANA DE

SAN NICOLAS DE HIDALGO

FACULTAD DE INGENIERIA QUIMICA

DIVISION DE ESTUDIOS DE POSGRADO

CONTROL DE TEMPERATURA DE UN REACTOR BATCH USANDO REDES NEURONALES EVOLUTIVAS

TESIS presentada por:

FRANCISCO JAVIER SANCHEZ RUIZ

Asesor:

Dr. Luis Ignacio Salcedo Estrada

A la División de Estudios de Posgrado de la
Facultad de Ingeniería Química como
requisito parcial para obtener el
grado de:
DOCTOR EN CIENCIAS
EN
INGENIERIA QUIMICA

Morelia, Mich. Marzo 2013

CONTENIDO

CONTROL DE TEMPERATURA DE UN REACTOR BATCH USANDO REDES NEURONALES EVOLUTIVAS

Doctorado en Ciencias en Ingeniería Química

Presenta: Francisco Javier Sánchez Ruiz

Asesor: Dr. Luís Ignacio Salcedo Estrada.

Resumen

Las redes neuronales son sistemas inteligentes que son adecuados para el control de sistemas no lineales y de comportamiento poco predecible, en la actualidad estos sistemas son en su mayoría aplicados a la solución de problemas de imágenes, comunicaciones y dentro de la robótica. Las características de las redes neuronales pueden considerase como sistemas alternos para el control de proceso, en especifico de temperatura en reactores de tipo batch. Mediante un entrenamiento y arquitectura adecuada ajustable a la dinámica del proceso, se puede generar un sistema de control inteligente que sea capaz de ajustar las variables de temperatura de acuerdo a la dinámica del proceso. En el presente trabajo se plantea a las redes neuronales evolutivas como una alternativa para el control de temperatura de un proceso de dinámica poco predecible y altamente no lineal, el uso de la red neuronal evolutiva da como resultado un fenotipo y genotipo el cual está en función de una regla de evolución o estrategia de evolución, para el caso de estudio, se establece una estrategia de evolución basada en la pendiente de la función de excitación de la neurona y la modificación del número de salidas de la neurona de la capa oculta, para establecer el cumplimento de la regla heurística de 2^n que determina los estados de evolución que puede presentar la red neuronal. La arquitectura de la red neuronal para el control del proceso batch consta de una neurona de entrada, una neurona en la capa oculta con diferente número de salidas y una neurona de salida, se utilizaron tres tipos de funciones de excitación (tangencial, logarítmica y radial) con diferentes magnitudes de pendiente (0.25-2.0), los resultados obtenidos de las simulaciones demuestran que la red con mejor desempeño fue la constituida por una arquitectura de tres salidas y una función de excitación de tipo base radial (RFB) que usa como estrategia de evolución la

magnitud de la pendiente de 0.5, además de presentar un valor de la integral del cuadrado del error de 0.001887 con respecto a los otros modelos de control, 0.3662; 0.0198 para el control PID *(Proporcional Integral Derivativo por sus siglas en inglés)* y GMC *(Generic Model Control por sus siglas en inglés)* respectivamente. El sistema de control obtenido fue aplicado teóricamente y experimentalmente a una reacción de polimerización que se lleva a cabo por el método de radicales libres en disolución en un reactor batch, se parte de un monómero como el metil metacrilato en presencia de peróxido de benzoilo el cuál actúa como iniciador, utilizando tolueno como solvente, la reacción presenta un proceso no lineal y con dinámica no predecible. En la parte teórica se realizo la simulación de la red neuronal evolutiva en función a la estrategia de evolución, mediante la modificación de la pendiente de la función de excitación y el número de salidas de las capas ocultas, y se obtuvo un control dentro del intervalo de 67.2 ºC – 67.5 ºC, eliminando de manera sustancial el sobretiro que presentaban los controles convencionales PID y GMC En la parte experimental se aplicaron los parámetros obtenidos en la parte teórica, los cuales se implementaron en la red neuronal evolutiva del sistema de control experimental, los cuales representan los valores iniciales de evolución de la red, el ajuste dinámico del sistema de control se realiza por medio del entrenamiento en línea de la red neuronal evolutiva, alcanzando los estados evolutivos necesarios para la dinámica del sistema de reacción. En el reactor batch de laboratorio se monitoreo las variables de control mediante la programación modular en LabView de la redes neuronales, considerando como punto de partida la red neuronal obtenida en la simulación con mejor desempeño, realizando la programación de la misma en la plataforma de LabView con el modelo de entrenamiento en línea no supervisado mediante los datos de temperatura del reactor y del entorno adquiridos durante la experimentación, la temperatura de control es de 67.5 ºC con un tiempo de reacción de 3.5 h, obteniendo un rendimiento del 60-65%. La aplicación de la red neuronal evolutiva en sistemas no lineales y dinámicos a nivel laboratorio abre el camino hacia una nueva vertiente del control de procesos basado en sistemas de inteligencia artificial complementado así las líneas ya existentes como la lógica difusa.

A mis Padres

María Felícitas Ruiz Hernández
Francisco Sánchez Montejano

A mi chaparro

Francisco Emanuel Sánchez Ruiz

LISTA DE TABLAS

LISTA DE FIGURAS

NOMENCLATURA

A = Área de transferencia.

Cp_m =Capacidad calorífica del Monómero.

Cp_s =Capacidad calorífica del solvente.

Cp_p =Capacidad calorífica del polímero.

Cp_i =Capacidad calorífica del iniciador.

Cp_r = Capacidad calorífica de la mezcla.

F_i = función de salida de la neurona i.

M_i = Masa del iniciado.

M_m = Masa del monómero.

M_p = Masa del polímero.

M_s = Masa del solvente.

M_r = Masa de la mezcla.

T_j = Temperatura de la chaqueta.

T_{jsp} = Temperatura de la chaqueta set-point

T_r = Temperatura del reactor.

U = Coeficiente de transferencia de calor.

V_j = Volumen de la chaqueta.

a_i = estado de activación de la neurona i.

f_i = función de excitación.

k_i = Constante de velocidad reacción i.

k_j = Constante de velocidad reacción j.

t = tiempo.

w_{ij} = pesos sinápticos.

x_i = entradas de la neurona i.

y_i = Neurona i.

ΔH_i = Calor de reacción del iniciador.

ΔH_m = Calor de reacción del monómero

ρ_j = Densidad.

σ_i = umbrales de la neurona i.

AGRADECIMIENTOS

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por el apoyo económico bajo la beca 169828 para mis estudios de Doctorado.

A la Universidad Michoacana de San Nicolás de Hidalgo por haberme albergado durante mis estudios de posgrado y su apoyo al inicio de los mismos estudios, soy orgullosamente NICOLAITA.

A mi asesor Dr. Luis Ignacio Salcedo Estrada por su contribuciones que enriquecieron este trabajo, y además de ser mi asesor se convirtió en un amigo. También quiero agradecer a la mesa evaluadora Dr. Medardo Serna González, Dr. José María Ponce Ortega, Dra. Ma. Carmen Chávez Praga y al Dr. Pedro Alberto Quintana Hernández este ultimo le agradezco las enseñanzas que mostro en las presentaciones de mi trabajo. A toda la división de posgrado de la Facultad de Ingeniería Química por su preocupación y apoyo en mi trabajo de investigación.

A Dios por haberme dado vida para realizar mis estudios y alcanzar una meta mas en mi vida.

A mis padres que son parte fundamental en mi vida, que han soportado todos mis estudios tal vez estén cansados de tener un hijo que quiere ser el eterno estudiante, gracias mamá y gracias papá.

A mis hermanos que siempre me han apoyado en cada uno de los proyectos que he emprendido gracias Pedro, Alma y Alejandra. A mis sobrinos, gracias por su cariño.

A la persona que me alegra los fines de semana haciendo que cada día sea mejor a mi chaparro espero que cuando estés grande puedas leer esto.

A mis amigos que son pocos pero buenos, por su apoyo en los días difíciles de mis estudios, gracias.

Capítulo 1

Introducción

En este capítulo se plantea de forma general los antecedentes e importancia del control de procesos, también se hace la justificación del trabajo de investigación que está basado en sistemas de control inteligente.

1.1 Control de Procesos

El control de procesos tiene una función vital en el avance de la ingeniería y de la ciencia, ya que permite la producción con la confiabilidad y calidad, que las altas exigencias del proceso y del mercado continuamente imponen.

En lo referente a la industria química, valor radica principalmente en la importancia de controlar variables de estado como: presión, temperatura y flujos, siendo estas la principales variables manipulables para el control.

De los diversos equipos con los que se cuenta en las plantas de proceso, son los reactores químicos, sin lugar a dudas, los que representan un mayor reto para poder implementar un sistema de control, ya que son sistemas altamente complejos.

Dentro de la gran variedad de reactores químicos, los de tipo por lotes (batch), constituyen un ejemplo de reactores donde su modelo matemático exhibe no linealidad, el modelo de dichos reactores se asemeja bastante al modelo obtenido en reactores continuos de mezcla completa (CSTR, en sus siglas en ingles), aunque con la limitante de que en los reactores por lotes no existen salidas de producto, por lo cual su complejidad hacia el control se ve incrementada (Luyben, 1989).

Para obtener los parámetros de los controladores de estos sistemas altamente no lineales, algunos investigadores han propuesto que el proceso se caracterice mediante un modelo de primer orden o de segundo orden con tiempo muerto (Smith-Corripio, 1996).

Sin embargo hay muy pocas propuestas sobre lo referente a la sintonización del control de temperatura de reactores por lotes ya que no es un problema trivial, y la utilización de métodos tradicionales genera respuestas pronunciadamente oscilatorias e inestables en muchos de los casos creando así un problema de sintonización muy complejo en comparación con otros reactores (Luyben, 1998).

En la industria química, para este tipo de reactores los controladores más comúnmente usados son los controladores humanos, pues requieren de cierta experiencia para su control. Usando control automático, los controladores usados son controladores clásicos como el control proporcional integral (PI) o proporcional integral derivativo (PID), pero en gran medida los parámetros de ajuste para el controlador son obtenidos con sintonización empírica, a fin de alcanzar coeficientes razonables de amortiguamiento a lazo cerrado, por no contar con métodos convencionales para su determinación (Galván y Zaldivar, 1992).

Para estos equipos, el objetivo primordial es mantener la conversión en un valor deseado, sin embargo esto se realiza normalmente manteniendo el control de la temperatura. Cuando la temperatura está bien controlada, las variaciones en la conversión del reactivo van muy de acuerdo con las variaciones de la temperatura. El control de la conversión de un producto no es tan convencional en muy pocos casos se maneja como variable a controlar debido a su alto costo en el control.

1.2 Planteamiento del problema

Las redes neuronales son sistemas inteligentes que son adecuados para el control de sistemas no lineales y de comportamiento poco predecible, se consideran sistemas que pueden ser una alternativa real para el control de temperatura en reactores de tipo batch teniendo un entrenamiento adecuado al sistema, además de contar con una arquitectura mínima que pueda realizar tareas complejas y ajustarse a la dinámica del proceso, el entrenamiento y la arquitectura de la red neuronal es un factor importante de alto peso específico para la comprensión del sistema, el peso del entrenamiento radica principalmente en la

obtención de los vectores de entrenamiento de las neuronas, los cuales especifican los máximos y mínimos de temperatura en el sistema a controlar; la arquitectura de la red neuronal también establece un parámetro sensible en los sistemas de control de procesos batch, esto se debe a la razón de número de neuronas presentes, lo que significa que la arquitectura con un número mínimo de neurona deberá dar excelentes resultados, aunque la evolución de la misma puede mejorarlos. La bibliografía menciona que la redes evolutivas son una alternativa para sistemas que presentan sobre aprendizaje y arquitecturas grandes las cuales generan dicho la propagación del error como vector de peso de la red neuronal lo cual genera un sobre aprendizaje, la neuro-evolución es la alternativa más indicada para procesos de dinámica no predecible, en este trabajo se plantea el uso de la neuro-evolución para la generación del fenotipo y genotipo adecuado para el control de temperatura de forma teórica y experimental de un reactor batch exotérmico, en donde se lleva a cabo la reacción de polimerización exotérmica por radicales libres del metil metacrilato, el cual es un proceso no lineal y con dinámica no predecible.

1.3 Hipótesis

La aplicación de técnicas evolutivas en los sistemas de control neuronal reducirá la cantidad de neuronas y capas ocultas para un sistema de control de temperatura en un reactor tipo batch, en comparación con los sistemas de control neuronal no evolutivo y mejorando la respuesta con respecto al control convencional, evitando el sobre-aprendizaje del sistema inteligente, generando de esta manera lo denominado como control neuro-evolutivo.

1.4 Objetivo general

Implementar teórica y experimentalmente un sistema neuronal evolutivo en el control de temperatura de un reactor batch en el que se lleve a cabo una reacción exotérmica de polimerización por radicales libre del metil metacrilato en presencia de Peróxido de Benzoilo como iniciador y utilizando tolueno como solvente.

1.5 Objetivos particulares

- Establecer una red neuronal evolutiva para el control de temperatura del reactor batch, mediante estrategias evolutivas.
- Comprobar teóricamente la cinética de reacción necesaria para el modelo matemático.
- Generar un fenotipo y genotipo teórico para la red neuronal, mediante una estrategia evolutiva.
- Obtener a nivel laboratorio los parámetros necesarios para el modelo matemático, que se usaran en la parte de implementación de la red neuronal evolutiva.
- Mejorar la Instrumentación del equipo a nivel laboratorio.
- Implementar una red neuronal evolutiva en un reactor batch de forma teórica y experimentalmente para controlar la temperatura.

Capítulo 2

Antecedentes

En este capítulo se abordan los temas relacionados al concepto de redes neuronales artificiales convencionales aplicadas al control de temperatura, así como su funcionamiento con respecto a los sistemas de control convencional, y el proceso de polimerización por vía de radicales libres, de forma conjunta se establece también las diferentes técnicas de evolución comúnmente usadas para dar origen a la nueva tendencia de la inteligencia artificial como lo es las redes neuronales evolutivas.

2.1 Redes Neuronales de tipo Biológico

El cerebro humano contiene aproximadamente 12 billones de células nerviosas o neuronas. Cada neurona tiene de 5600 a 60000 conexiones dendríticas provenientes de otras neuronas mientras que en el sistema nervioso hay 1014 sinapsis; teniendo cada neurona más de 1000 a la entrada y a la salida. Es importante destacar que aunque el tiempo de conmutación de la neurona es casi un millón de veces menor que las computadoras actuales, ellas tienen una conectividad miles de veces superior que las actuales supercomputadoras.

La principal aplicación de estas redes, es el desarrollo de elementos sintéticos para verificar las hipótesis que conciernen a los sistemas biológicos.

Las neuronas y las conexiones entre ella, llamadas sinapsis, son la clave para el procesado de la información (ver Figura 2.1).

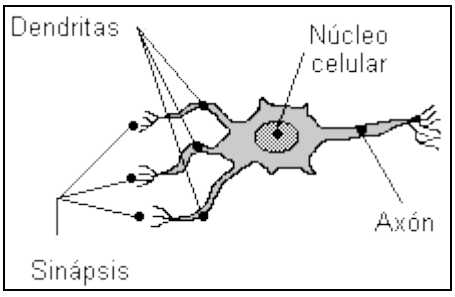


Figura 2.1 Neurona biológica.

La mayoría de neuronas tienen una estructura parecida a la de un árbol llamadas dendritas que reciben las señales de entrada que vienen de otras neuronas a través de las sinapsis.

Una neurona consta de tres partes (ver Figura 2.2):

1. El cuerpo de la neurona
2. Ramas de extensión (dendritas) para recibir las entradas
3. Un axón que lleva la salida de una neurona a las dendritas de otras neuronas

La interacción entre dos neuronas no es del todo conocida pero el proceso del traspaso de información es modelado como una regla de propagación representada por la red *u(t)*. Mientras que la neurona puede ser modelada como una simple función escalón *f(t)*.

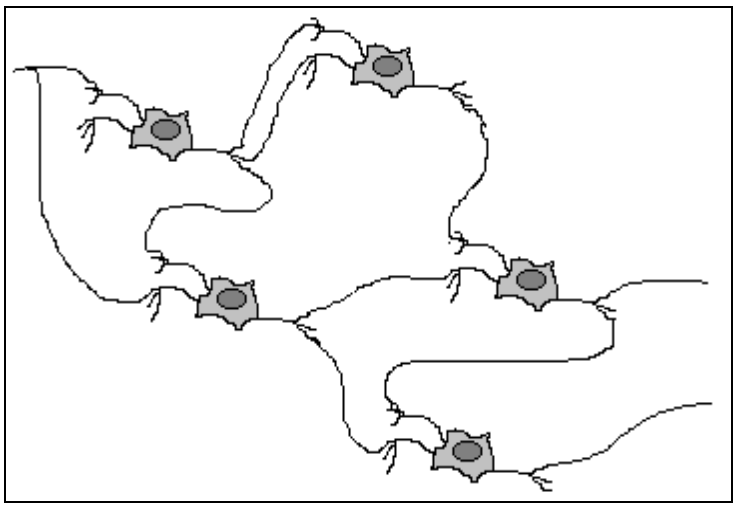


Figura 2.2 Red neuronal biológica

2.2 Redes Neuronales Artificiales

Las redes neuronales son sistemas inteligentes y dinámicos que de acuerdo con el entrenamiento puede adecuarse a sistemas no lineales y lineales de solución compleja y difícil.

Las Redes Neuronales Artificiales (RNA) surgieron de la observación del funcionamiento del cerebro y de su comparación con la forma de trabajar de los ordenadores digitales. Los elementos estructurales constituyentes del cerebro, las neuronas biológicas (Ramón y Cajal, 1911), son unas seis órdenes de magnitud

más lentas que la programación lógica, ofreciendo tiempos de respuesta del orden del milisegundo frente a los vertiginosos tiempos de respuesta del orden del nanosegundo que tienen ciertos dispositivos digitales actuales.

Esta relativa lentitud no impide al cerebro realizar ciertas funciones habituales, como son las tareas de reconocimiento, percepción y control motriz, con una eficacia y rapidez fuera del alcance del mejor ordenador digital. Esta supremacía le viene dada por la alta complejidad de su estructura, formada por un elevado número de células nerviosas (neuronas) masivamente interconectadas y funcionando en paralelo. Se estima que el número de neuronas en el cerebro humano es del orden de diez millares de millones (10^{10}), y que el número de sinapsis o conexiones es del orden de 60 billones ($60x\ 10^{12}$). La alta complejidad de esta estructura biológica no sería operativa si no fuese además eficiente. La eficiencia energética del cerebro es aproximadamente de 10^{-16} J por operación por segundo, mientras que los mejores ordenadores no superan los 10^{-6} J por operación por segundo (Faggin, 1991).

Al nacer, el cerebro ya tiene gran parte de su estructura formada, y lo que es más importante, está dotado de los mecanismos necesarios para construirse sus propias reglas a través de lo que conocemos como “experiencia”. Este proceso de aprendizaje se extiende a lo largo de los años, aunque el desarrollo más espectacular tiene lugar durante los dos primeros años de vida, durante los cuales se forman alrededor de un millón de sinapsis por segundo. La adaptación del sistema nervioso a su entorno sigue dos mecanismos distintos en un cerebro adulto: la creación de nuevas conexiones sinápticas y la modificación de las conexiones existentes.

Se puede establecer que el cerebro humano es una estructura compleja, no lineal y paralela de tratamiento de información que almacena su conocimiento en las conexiones que ligan a sus elementos de proceso (neuronas), y que es capaz de adaptarse a su entorno. Las RNA están inspiradas de la estructura del cerebro, y fueron concebidas para resolver cierto tipo de problemas especialmente mal resueltos por las técnicas de programación tradicionales. Formalmente, computación neuronal es la disciplina tecnológica que trata sistemas paralelos y adaptativos de procesamiento de información distribuida, que desarrollan sus capacidades bajo su exposición a un entorno de información.

Esta definición muestra claramente el paralelismo entre la estructura cerebral biológica y las RNA. Este paralelismo ha motivado a muchos investigadores de diversos campos de la ciencia a profundizar en la interpretación biológica de las RNA, proponiendo nuevas estructuras conexionistas y estrategias de aprendizaje directamente inspiradas de la modelización del cerebro. Estos estudios han dado resultados interesantes en el campo de las RNA, pero desde un punto de vista subjetivo, lo más importante de ellos es el conocimiento que directa o indirectamente está aportando para esclarecer los complejos procesos de aprendizaje, percepción y gestión del conocimiento que tienen lugar en el cerebro humano.

Una RNA es una estructura de procesamiento paralelo de información distribuida, bajo forma de grafo orientado, con las siguientes subdefiniciones y restricciones:

- Los nodos del grafo son llamados *elementos de proceso* o *neuronas*.
- Las uniones del grafo son llamadas *conexiones* (unidireccionales e instantáneas).
- Se admite cualquier número de conexiones de entrada.
- Cada elemento de proceso sólo puede tener una señal de salida, que puede ramificarse en cualquier número de conexiones de salida.
- Los elementos de proceso pueden tener *memoria local*.
- Cada elemento de proceso tiene una *función de transferencia* que puede utilizar y alterar la memoria local, puede usar las señales de entrada, y produce la señal de salida. La función de transferencia puede actuar de forma continua o discreta en el tiempo. Si actúa de forma discreta, existirá una entrada de estímulo que active la aplicación de la función de transferencia proveniente de la unidad de control de la RNA.

La Figura 2.3 muestra el esquema general de un elemento de proceso, que consta de dos funciones, una de transferencia y una memoria local, donde la función de transferencia es la parte principal del modelo neuronal para el proceso.

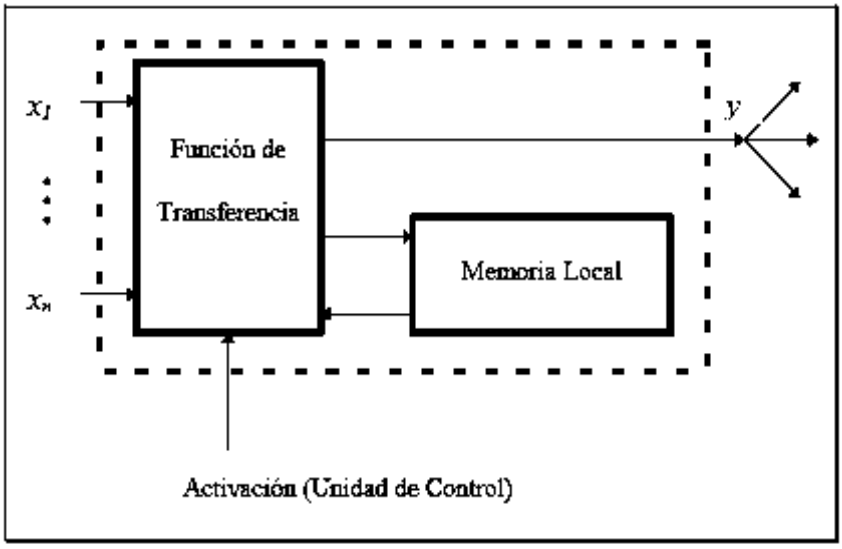


Figura 2.3 Estructura de un elemento de proceso

Podemos concluir que las RNA son estructuras adaptativas de procesamiento de información inspiradas en la estructura cerebral, donde el procesamiento se lleva a cabo mediante la interconexión de elementos de proceso muy sencillos a los que llamaremos neuronas. Esta arquitectura da lugar a estructuras altamente paralelizables donde el flujo de información no sigue un camino secuencial, sino que se distribuye a través de las conexiones de los elementos de proceso donde la información es tratada.

Durante la fase de aprendizaje, la RNA expone a un entorno de información para que pueda adaptar sus pesos (parámetros libres que dan forma a las funciones de transferencia de sus elementos de proceso) y posiblemente también su estructura.

Durante la fase de evaluación, los pesos de la red se mantienen fijos y la RNA se limita a tratar la información de entrada que se le suministra.

La estructura de la red neuronal puede variar esto depende de las necesidades requeridas y de los datos con los que se cuenta para el entrenamiento de la red neuronal, las más comúnmente usadas son de dos tipos:

a) Redes Multicapas Estáticas.

b) Redes Recurrentes Dinámicas.

a) Redes Multicapas Estáticas.

Las redes multicapas, son las más conocidas y usadas, aun cuando existen redes que pueden representar de mejor manera la capacidad de conocer el funcionamiento dinámico de un proceso (Dirion *y col*, 1996).

El perceptrón multicapa (MLP) es el tipo de red estática más común. Tiene una topología de alimentación hacia delante totalmente conectada como se aprecia en la Figura 2.4.

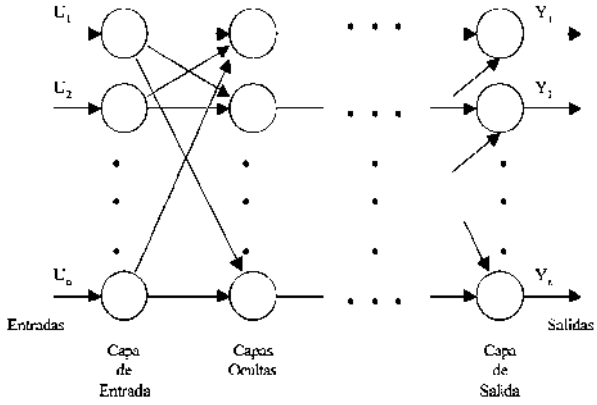


Figura 2.4 Estructura de la red neuronal estática

Para este tipo de redes, la salida en un instante determinado no depende de otros instantes, si las neuronas no tienen una dinámica interna.

Red estática de funciones de base radial (RBF)

La red de funciones de base radial (RBF) también es una red estática (Figura 2.5). Está formada por dos capas. Una de ellas está compuesta por neuronas con función de activación radial (usualmente gaussianas). Cada neurona posee un peso asociado a cada entrada, determinando la posición o centroide de la gaussiana. El resto de pesos representan las varianzas o dispersiones de cada gaussiana. La salida de cada neurona viene dada por:

$$u_{i,j} = exp\left[-\frac{(x-w_{1,j})'(x-w_{1,j})}{2\sigma_j^2}\right] \qquad j = 1,2,\ldots,N_1 \tag{2.1}$$

Donde *u* es la salida de la neurona *i,j, x* es el vector de referencia en el estado igual a 1, *w* son los vectores de peso de entrenamiento *i,j,* y *σ* es la varianza de la función de excitación o de transferencia.

Dirion y colaboradores, en 1996, sintetizaron una red neuronal usando el algoritmo de multicapas obteniendo resultados en el control de un reactor por lotes

buenos, en donde la desviación con los resultados obtenidos de planta es muy pequeña.

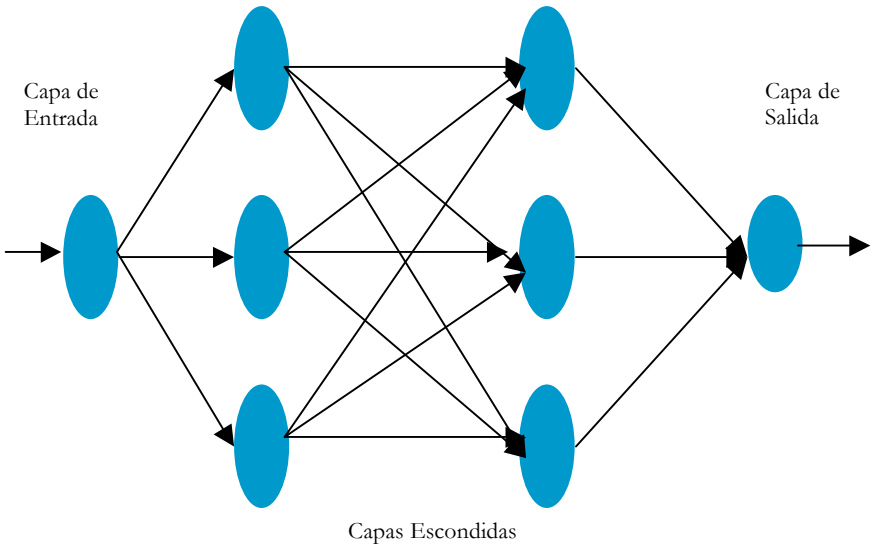


Figura 2.5 Arquitectura de red multicapas.

Koza en 1992**,** sintetizo una red neuronal multicapas, para el control de temperatura en un reactor CTRS por sus siglas en inglés, entrenando la red con datos de simulaciones y datos de planta, mostrando así una disminución del error y un mejor control de la reacción

b) Redes Recurrentes Dinámicas

En este tipo de redes el funcionamiento se describe mediante ecuaciones en diferencia o diferenciales dependiendo de la naturaleza discreta o continua de las variables. Estas redes son importantes porque muchos de los sistemas que se pueden modelizar en el mundo real tienen características dinámicas.

Red neuronal dinámica con retardo en el tiempo (RNDRT)

Es una red dinámica simple que se puede considerar como una red estática en la que se introducen valores de la entrada en instantes anteriores (Hertz, 1991), como aparece en la Figura 2. 6.

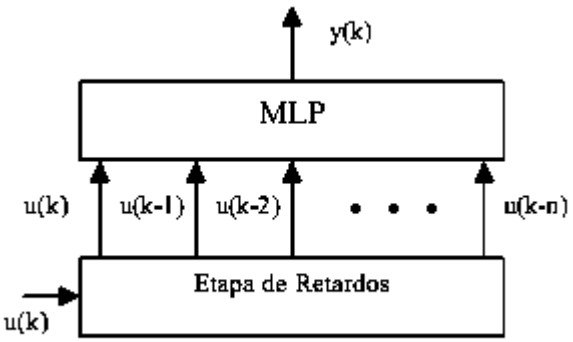


Figura 2.6 Red Neuronal de tipo RNDRT

La salida está dada por la siguiente ecuación:

$$y(k) = F(u(k), u(k-1), \ldots, u(k-n) \tag{2.2}$$

Lo que significa que para cada valor de salida existe una dependencia con el retardo para cada uno de los puntos de la red neuronal.

Este tipo de redes son más difíciles de estructurar y el entrenamiento más lento, por esto ha sido mucho menos utilizadas, aunque para la estructuración de dichas redes se pueden estructurar de acuerdo a los resultados que deseemos obtener esto utilizando lazos de retroalimentación sobre los nodos internos (Drakopoulos, 2005).

Una de las ventajas de la redes recurrentes es que tienen una mejor capacidad predictora que la red multicapa, ya que su diseño es independiente al sistema y por lo tanto no acarrea la acumulación del error de ajuste y esto la hace más adecuada para calcular la evolución temporal de las variables de salida de un proceso dadas sus entradas.

Una de las estructuras más utilizadas en las redes recurrentes es la red recurrente diagonal (EDR por sus siglas en inglés) (Figura 2.7).

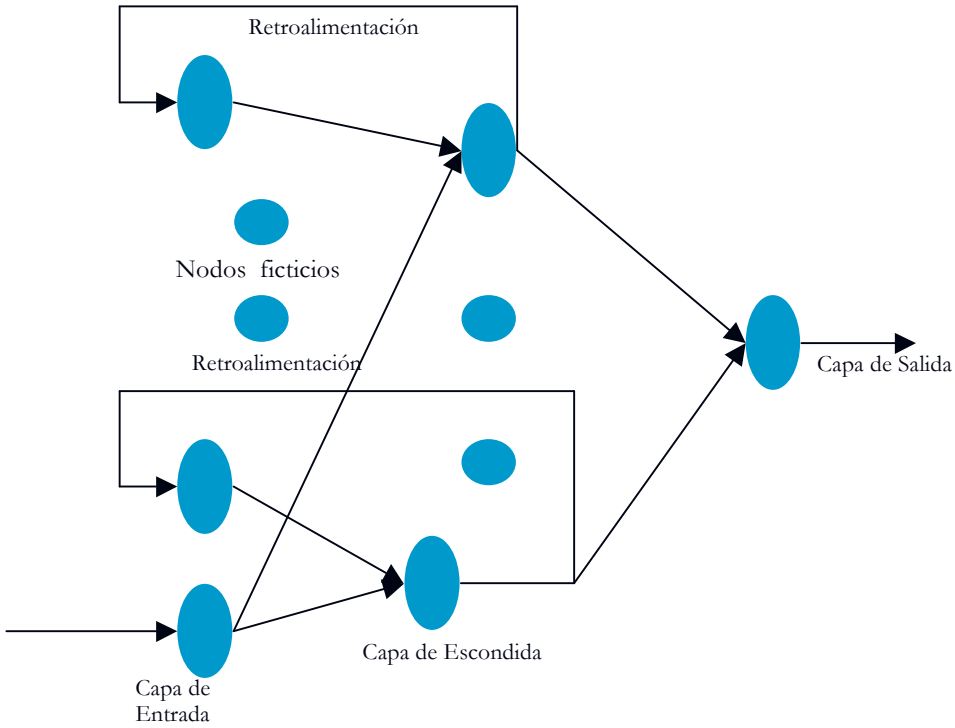


Figura 2.7 Red recurrente diagonal (EDR).

Luus, en 1994, sintetizo una red recurrente modificada para el control de pH en un reactor de saponificación, obteniendo buenos resultados comparados con trabajos semejantes en donde se utilizaron datos de planta y redes multicapas.

Daosud y colaboradores, en 2005, utilizaron las redes neuronales recurrentes para el control de temperatura en procesos que presentan una reacción exotérmica y además para linealizar el modelo matemático obteniendo resultados muy satisfactorios y mejores que usando otro tipo de controlador.

2.3 Entrenamiento de una red neuronal

Las redes neuronales son modelos matemáticos diseñados para emular algunas capacidades del cerebro humano y están compuestos por muchos elementos computacionales simples llamados nodos o neuronas (que imitan el comportamiento de la neurona biológica), y que están altamente conectados entre sí. Las redes neuronales pueden aproximar arbitrariamente bien cualquier función no lineal, dentro de un conjunto compacto y acotado, y son por tanto aproximadores universales (Cybenko, 1989).

Los modelos neuronales están especificados por una topología (diseño estructural de conexiones y nodos) de la red, características de los nodos, y reglas o algoritmos de entrenamiento o aprendizaje. El entrenamiento, utilizando datos de operación de un sistema dado, permite especificar el conjunto de pesos o conexiones sinápticas de la red, para dar a esta capacidad de representar al sistema en cuestión. La red constituye entonces un modelo empírico del sistema, que permite predicción de sus variables de salida, o en el caso de un sistema dinámico, la inferencia de su comportamiento a lo largo del tiempo. Esta habilidad de representar sistemas es una de las características más importantes de las redes neuronales y es utilizada para la identificación de sistemas y para la síntesis de controladores (Hunt y col., 1992).

Las redes neuronales pueden tener factores de pesos fijos y adaptables. Las que tienen pesos adaptables emplean leyes de aprendizaje para ajustar el valor de la fuerza de una interconexión con otras neuronas. Si las neuronas utilizan pesos fijos, entonces su tarea deberá estar previamente definida. Por otra parte, los pesos adaptables son esenciales si no se conocen previamente cual deberá ser su valor correcto.

El aprendizaje de la neurona lo podemos clasificar en dos tipos un aprendizaje supervisado y otro aprendizaje no supervisado, el primero ocurre cuando se le proporciona a la red tanto la entrada como la salida correcta, y la red ajusta sus pesos tratando de minimizar el error de su salida calculada. Este tipo de entrenamiento se aplica por ejemplo en el reconocimiento de patrones. El entrenamiento no supervisado se presenta cuando a la red se le proporcionan únicamente los estímulos, y la red ajusta sus interconexiones basándose únicamente en estímulos y la salida de la propia red. Las leyes de aprendizaje determinan como la red ajustara sus pesos utilizando una función de error o algún otro criterio. La ley de aprendizaje adecuada se determina en base a la naturaleza del problema que se intente resolver (Hopfield, 1982).

Propagación hacia atrás

Uno de los grandes avances logrados con la propagación hacia atrás es que esta red aprovecha la naturaleza paralela de la redes neuronales para reducir el

tiempo requerida para el procesamiento secuencial para determinar lo patrones entre un punto dado. Además el tiempo de desarrollo de cualquier sistema al cual se esté tratando de analizar se puede reducir como secuencia de que la red puede aprender el algoritmo correcto sin que alguien tenga que deducir por anticipado el algoritmo en cuestión.

La propagación hacia atrás es un tipo de red de aprendizaje supervisado, que emplea un ciclo de propagación – adaptación de dos fases una vez que se ha aplicado un patrón a la entrada de la red como estimulo, este se propaga desde la primera capa a través de las capas superiores de la red, hasta generar una salida. La señal de salida se compara con la salida deseada y se calcula una señal de error para cada una de las salidas.

Las salidas del error se propagan hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa oculta que contribuyen directamente a la salida. Sin embargo, las neuronas de la capa oculta solo reciben una fracción de la señal total del error, basándose aproximadamente en la contribución relativa que haya aportado cada neurona a la salida original. Este proceso se repite capa por capa, hasta que todas las neuronas de la red hayan recibido una señal de error que describa su contribución relativa al error total. Basándose en la señal de error que percibida, se actualizan los pesos de conexión de cada neurona, para hacer que la red converja hacia un estado que permita clasificar correctamente todos los patrones del entrenamiento.

La importancia de este proceso consiste en que, a medida que se entrena la red, las neuronas de las capas intermedias se organizan a sí mismas de tal modo que las distintas neuronas aprenden a reconocer distintas características del espacio total de entrada. Después del entrenamiento, cuando se les presente un patrón arbitrario de entrada que contenga ruido o que este incompleto, las neuronas de la capa oculta de la red responderán con una salida activa si la nueva entrada contiene un patrón que se asemeje a aquella característica que las neuronas individuales hayan aprendido a reconocer.

Varias investigaciones han demostrado que, durante el proceso de entrenamiento, la red de propagación hacia atrás tiende a desarrollar relaciones internas entre neuronas con el fin de organizar los datos de entrenamiento por

clases. Esta tendencia se puede extrapolar, para llegar a la hipótesis consiste en que todas las unidades de la capa oculta de una propagación hacia atrás son asociadas de alguna manera a características específicas del patrón de entrada como consecuencia del entrenamiento. Lo que sea o no exactamente asociado puede no resultar evidente para el observador humano, lo importante es que la red ha encontrado una representación interna que le permite generar las salidas deseadas cuando se le dan las entradas, en el proceso de entrenamiento. Esta misma representación interna se puede aplicar a entradas que la red no haya visto antes, y la red clasificará estas entradas según las características que comportan con los ejemplos de entrenamiento (Galván y Zaldivar, 1999).

El diseño de un controlador neuronal basado en el entrenamiento de propagación hacia atrás, puede considerarse como una arquitectura compleja en donde el principal factor de peso es el entrenamiento del neúron, en donde, se ha demostrado que un buen entrenamiento puede llevar a un tiempo de computo menor que en el caso de utilizar un entrenamiento diferente al usado con propagación hacia atrás.

El *backpropagation* fue creado mediante la generalización de la regla de aprendizaje Widrow-Hoff para redes multicapas y las funciones de transferencia diferenciables no lineales. Los vectores de entrada y los correspondientes vectores deseados se usan para entrenar una red hasta cuando ella pueda aproximar una función, asociando los vectores de entrada con los vectores de salida específicos, ó clasificar los vectores de entrada de una manera apropiada. Las redes con bases, una capa sigmoide, y una capa de salida lineal son competentes para aproximar cualquier función con un número finito de discontinuidades.

El *backpropagation* estándar es un algoritmo de gradiente descendente, como lo es la regla de aprendizaje Widrow-Hoff, en la cual los pesos de la red son movidos a lo largo del negativo del gradiente de la función de ejecución. El término *backpropagation* se refiere a la manera como el gradiente es calculado para redes multicapa no lineales. Existe un cierto número de variaciones en el algoritmo básico, las cuales están basadas en otras técnicas de optimización, tales como el gradiente conjugado y los métodos de Newton. Con el Toolbox de Redes Neuronales de Matlab^{R} podemos implementar estas variaciones.

Las redes *backpropagation* entrenadas orientan a dar respuestas razonables cuando se les presentan entradas que aún no han sido consideradas. Típicamente, la entrada conduce a una salida similar a la salida correcta para vectores de entrada que se han empleado en el entrenamiento, los cuales a su vez son similares a las nuevas entradas que están siendo presentadas. Esta generalización con propiedad hace posible entrenar una red sobre un conjunto representativo de las parejas entrada / salida deseada, y se obtienen buenos resultados sin entrenar la red sobre todas las posibles parejas de entrada/salida. Existen dos características del Toolbox de Redes Neuronales de Matlab^R las cuales están diseñadas para mejorar la generalización y regularización de la red y la detención remota.

2.4 Aplicaciones de Redes Neuronales al control de procesos

Las redes neuronales tienen una amplia aplicación a sistemas con características robóticas, además de presentar ajuste para sistemas de reconocimiento de imágenes, aunque en la actualidad existen diferentes vertientes de aplicación de la redes neuronales en el control de procesos químicos, mediante diferentes configuraciones del control neuronal.

Neurocontrolador libre de modelo

Ante la ausencia de un controlador existente, algunos investigadores han desarrollado la opción para que un controlador aprenda como actuaría un controlador humano con poco o nada de conocimiento detallado sobre la dinámica de proceso. También se ha intentado diseñar controladores que por adaptación y aprendizaje puedan resolver problemas de control difíciles en la ausencia de los modelos del proceso o de experiencia humana al respecto. La clave en esta estructura de control adaptivo es que el modelo del proceso no es conocido previamente y tampoco es muy explícito durante el diseño del controlador. Esta técnica para el diseño de controladores es frecuentemente denominada como aprendizaje reforzado. Sin embargo, a esta clase de controladores se les conoce como neurocontroladores libres de modelo, ya que desde la óptica particular es más apropiado en el contexto. La Figura 2.8 es una representación clásica de esta clase de controladores.

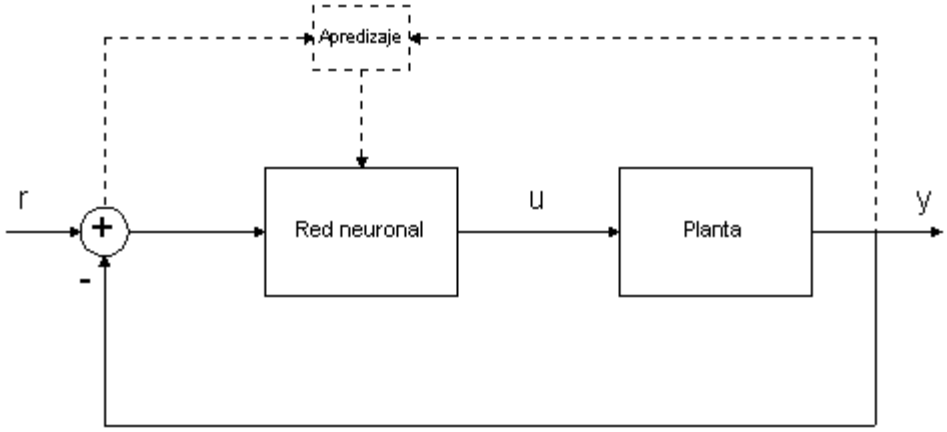


Figura 2.8 Arquitectura de controladores libres de modelo

El primer trabajo en esta área fue el algoritmo de adaptivo crítico propuesto por Werbos en 1988. Este algoritmo puede ser visto como una versión aproximada de la programación dinámica. En el trabajo se desarrolla el problema del péndulo invertido mostrando el concepto del diseño de control (Werbos, 1988). En esta clase de diseño la información es limitada o pobre, y es frecuentemente adoptada como un indicador para los criterios de rendimiento del sistema. Por ejemplo, el objetivo del problema del péndulo invertido es simplemente mantener el péndulo cercano a su posición vertical balanceada. La retroalimentación en función de instrucciones es limitada a una señal de falla cuando el controlador no logra mantener el péndulo en la posición vertical. El problema de balancear el péndulo invertido ha llegado a ser popular como prueba para explorar nuevos conceptos de diseño para controladores libres de modelo.

La solución propuesta por Werbos fue construir un esquema de control el cual está compuesto por dos elementos adaptivos: un Elemento de Búsqueda Asociativa (ASE por sus siglas en ingles) y un Elemento Crítico Adaptivo (ACE por sus siglas en ingles). El Elemento de Búsqueda trata de reproducir la señal de control óptimo que satisface los objetivos de rendimiento definidos, mientras que el Elemento Crítico trata de monitorear el rendimiento interno del controlador y proveer una señal de reforzamiento interna la cual es usada para entrenar al ASE como se ve en la Figura 2.9.

El ACE es entrenado utilizando señales de fallas/éxitos externas. Este entrenamiento interno continuo del elemento de control pretende implementar mejoras en el rendimiento de todo el sistema.

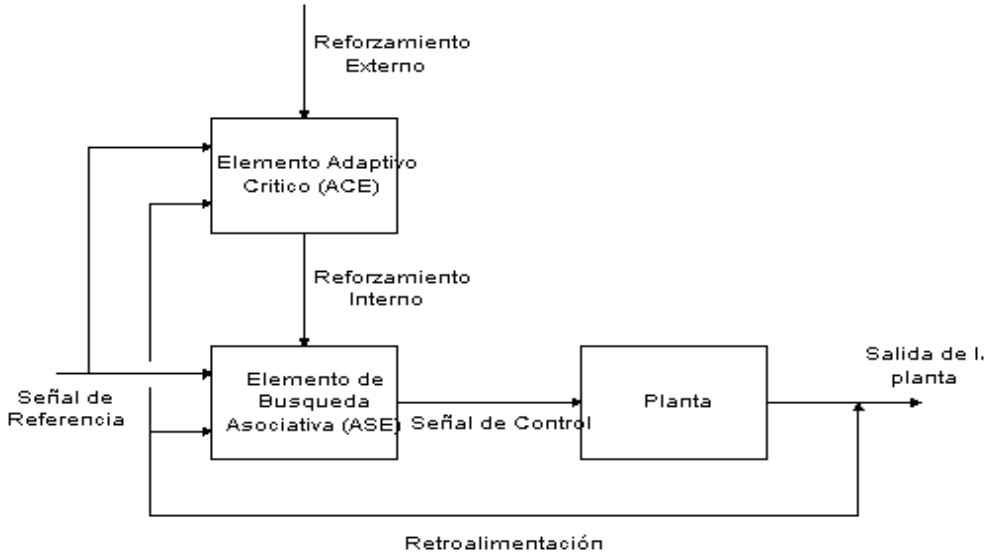


Figura 2.9 Arquitectura de aprendizaje reforzado planteada por Werbos, utilizando el concepto de ACE y ASE.

Aún con su importancia histórica y sus conceptos intuitivos, los neurocontroladores adaptivos libres de modelo no son apropiados para la mayoría de las aplicaciones del mundo real. La planta deberá de pasar por estados fuera de control durante el proceso de aprendizaje y pocos procesos industriales pueden tolerar esa larga cantidad de fallas necesarias para adaptar el controlador.

Neurocontrolador basado en modelo

Desde una perspectiva práctica, uno prefería dejar las fallas para un ambiente simulado utilizando un modelo más que una planta real, aun cuando las fallas sea desastrosa o no causen pérdidas sustantivas en las etapas tempranas del aprendizaje neuronal. Esta clase de neurocontroladores es llamada frecuentemente "neurocontroladores basados en modelo".

Si el modelo del proceso no es disponible, uno puede primero entrenar una segunda red neuronal para modelar la dinámica de planta. En el curso del modelado de la planta esta debe de ser operada "normalmente" en lugar de llevarla fuera de

control. Después de la etapa de modelado, el modelo puede ser usado para el diseño de controladores. Si un modelo de la planta puede ser desarrollado, entonces en una simulación, en la cual las fallas no pueden causar ninguna pérdida más de allá del tiempo de cómputo, un controlador basado en redes neuronales puede instalarse en el sistema de control real, después de un extenso entrenamiento en alguna plataforma de simulación (Figura 2.10).

En efecto, estos diseños de neurocontroladores basados en modelo, no solamente han probado su eficiencia en distintos estudios (Dirion *y col*, 1996) sino que también ya han probado generar beneficios económicos notables. Dichos elementos pueden ser usados para diseños fuera de línea ó en adaptaciones en línea.

Demostraciones exitosas han sido desarrolladas para el clásico problema de control del vuelo multivariable. Quizás el mejor éxito comercial del neurocontrol a la fecha, es aquella propuesta que trata de un horno de arco inteligente, el cual utiliza una red neuronal para regular la posición del electrodo en hornos eléctricos de arco. Publicaciones comerciales reportan ahorros típicos de sobre 2 millones de dólares en un horno de cocimiento de cerámica (Sanz, 2005). El controlador inteligente del horno de arco incluye una interesante combinación para el desarrollo de un neurocontrolador. Inicialmente un controlador neuronal es entrenado para actuar como un controlador exitoso para la planta. Después de entrenar, la red neuronal reemplaza dicho controlador. En esta última etapa, una segunda red neuronal pre-entrenada se utiliza como modelo del proceso y el controlador neuronal continúa adaptándose en línea para contener las principales fallas de la planta.

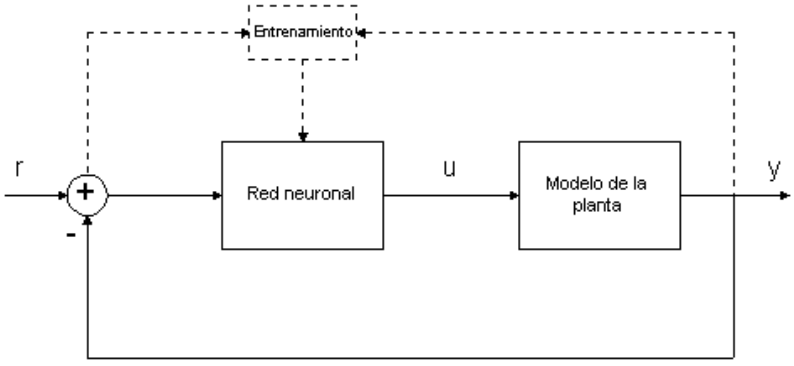


Figura 2.10 Un modelo reemplaza a la planta en el sistema de aprendizaje durante la fase de diseño del controlador.

Neurocontrolador robusto basado en modelo

El tipo de neurocontrolador descrito en la sección pasada tiene todavía una desventaja común hasta ahora: una red neuronal debe entrenarse cada vez que se tenga una nueva aplicación. El re-entrenamiento de la red es necesario aún cuando son pequeños los cambios del criterio de control, tanto para cambios en los pesos de la red relativos a la energía del control como para la respuesta de seguimiento, ó si el controlador va a utilizarse para procesos homólogos pero distintos. Con el objetivo de evitar estas desventajas, el concepto de controlador robusto es naturalmente incluido dentro del diseño de neurocontroladores. En el diseño de sistemas de control neuronal basados en un modelo robusto, se consideran familias de los modelos del proceso en lugar de un solo modelo. Frecuentemente, una familia se agrupa por modelos que presentan los mismos niveles de ruido o bien pueden agruparse atendiendo el criterio del número de parámetros del proceso. Dos aspectos acerca de que tan robusto es un sistema se señalan frecuentemente: La estabilidad que refiere si un sistema es estable cualitativamente y el rendimiento robusto que opera con respecto a criterios cuantitativos (Werbos, 1988). Optimizar un controlador neuronal basado en un modelo preciso, puede alcanzarse un alto rendimiento, tanto como el proceso permanece invariable, pero con altos costos de fragilidad. Un procedimiento para el diseño robusto, por otro lado, no es capaz de alcanzar el mismo nivel de rendimiento nominal pero será menos sensible a estados obsoletos del proceso, ruido y otras fuentes de error en la relación modelo-proceso.

2.5 Algoritmos Genéticos

El algoritmo genético es una técnica de búsqueda basada en la teoría de la evolución de Darwin en 1859, que ha cobrado tremenda popularidad alrededor del mundo durante los últimos años.

Fue en las décadas de 1950 y 1960 cuando varios científicos, de modo independiente, comenzaron a estudiar los sistemas evolutivos, guiados por la intuición de que se podrían emplear como herramienta en problemas de optimización en ingeniería. La idea era *evolucionar* una población de candidatos a ser solución de un problema conocido, utilizando operadores inspirados en la selección natural y la variación genética natural.

Con esta idea nace en el año 1993 la computación evolutiva, que retoma conceptos de la evolución y la genética para resolver principalmente problemas de optimización. Esta rama de la inteligencia artificial tiene sus raíces en tres desarrollos relacionados pero independientes entre sí:

- Algoritmos genéticos
- Programación evolutiva
- Estrategias Evolutivas

De estas tres áreas parten los caminos hacia todos los campos de investigación inspirados en nuestros conocimientos sobre Evolución. Las estrategias evolutivas son métodos computacionales que trabajan con una población de individuos que pertenecen al dominio de los números reales, que mediante los procesos de mutación y de recombinación evolucionan para alcanzar el óptimo de la función objetivo. Entre 1965 y 1973 Rechenberg las introdujo como método para optimizar parámetros reales para ciertos dispositivos. La misma idea fue desarrollada poco después por Schwefel en 1965. El campo de las estrategias evolutivas ha permanecido como un área de investigación activa, cuyo desarrollo se produce, en su mayor parte, de modo independiente al de los algoritmos genéticos (aunque recientemente se ha visto como las dos comunidades han comenzado a colaborar).

La programación evolutiva (PE) es prácticamente una variación de los algoritmos genéticos, donde lo que cambia es la representación de los individuos. En el caso de la PE los individuos son ternas (tripletas) cuyos valores representan estados de un autómata finito. Cada terna está formada por el valor del estado actual, un símbolo del alfabeto utilizado y el valor del nuevo estado.

Estos valores se utilizan como en un autómata finito, teniendo el valor del estado actual en el que nos encontramos, tomamos el valor del símbolo actual y si es el símbolo de nuestra terna, nos debemos mover al nuevo estado. Además las funciones de selección, cruce (*crossover*) y mutación deben variar para adaptarse y funcionar con una población de individuos de este tipo.

Fogel y colaboradores fueron los creadores en 1966 de la programación evolutiva, una técnica en la cual las candidatas a soluciones a tareas determinadas, eran representadas por máquinas de estados finitos, cuyos diagramas de estados

de transición se evolucionaban mediante mutación aleatoria, seleccionándose el que mejor aproximará.

La primera mención del término de algoritmo genético, y la primera publicación sobre una aplicación del mismo, se deben a Bagley en 1967. Este investigador diseñó algoritmos genéticos para buscar conjuntos de parámetros en funciones de evaluación de juegos, y los comparó con los algoritmos de correlación, procedimientos de aprendizaje modelizados después de los algoritmos de pesos variantes de ese periodo. Sin embargo el que es considerado como el creador de los algoritmos genéticos es John Holland, que los desarrolló, junto a sus alumnos y colegas, durante las décadas de 1960 y 1970.

En contraste con las estrategias evolutivas y la programación evolutiva, el propósito original de Holland no era diseñar algoritmos para resolver problemas concretos, sino estudiar, de un modo formal, el fenómeno de la adaptación tal y como ocurre en la naturaleza, y desarrollar vías de extrapolar esos mecanismos de adaptación natural a los sistemas computacionales. El libro que Holland escribió en 1975 *Adaptación en sistemas naturales y artificiales* presentaba el algoritmo genético como una abstracción de la evolución biológica, y proporcionaba el entramado teórico para la adaptación bajo el algoritmo genético. El algoritmo genético de Holland era un método para desplazarse, de una población de cromosomas (bits) a una nueva población, utilizando un sistema similar a la selección natural junto con los operadores de *cruces, mutaciones e inversión* inspirados en la genética. En este primitivo algoritmo, cada cromosoma consta de genes (bits), y cada uno de ellos es una muestra de un alelo particular (0 o 1).
Holland adapta los operadores de selección, cruces, mutaciones e inversión a su algoritmo:

- **Selección:** este operador escoge entre los cromosomas de la población aquellos con capacidad de reproducción, y entre estos, los que sean más *compatibles*, producirán más descendencia que el resto.

- **Cruce:** extrae partes de dos cromosomas, imitando la combinación biológica de dos cromosomas aislados (gametos).

- **Mutación:** se encarga de cambiar, de modo aleatorio, los valores del alelo en algunas localizaciones del cromosoma.

- **Inversión:** invierte el orden de una sección contigua del cromosoma, recolocando por tanto el orden en el que se almacenan los genes.

La mayor innovación de Holland fue la de introducir un algoritmo basado en poblaciones con cruces, mutaciones e inversiones. Holland fue el primero en intentar colocar la computación evolutiva sobre una base teórica firme. Hasta hace poco, esta base teórica, fundamentada en la noción de *esquemas*, fue la estructura sobre la que se edificaron la mayoría de los trabajos teóricos sobre algoritmos genéticos en las décadas siguientes.

Además de los investigadores de los que se han hablado muchos otros investigadores desarrollaron su trabajo en los algoritmos para la optimización y el aprendizaje inspirados en la evolución. Cabe resaltar nombres como los de Box, Friedman, Bledsoe, Bremermann, y Baricelli (Jong, 1975). Sin embargo, su trabajo no ha tenido, ni con mucho, la atención que han recibido las estrategias evolutivas, programación evolutiva, y los algoritmos genéticos. Hay que recordar además a los biólogos evolucionistas que han utilizado el ordenador para simular la evolución para realizar experimentos controlados (Beasley y col, 1993). Pero habría que esperar hasta que la computación electrónica se desarrollara, para poder apreciar la consolidación definitiva de la computación evolutiva.

En estos últimos años se ha generado una amplia interacción entre los investigadores de varios métodos de computación evolutiva, rompiéndose las fronteras entre algoritmos genéticos, estrategias evolutivas y programación evolutiva. Como consecuencia, en la actualidad, el término “algoritmo genético” se utiliza para designar un concepto mucho más amplio del que concibió Holland.

2.6 Redes Neuronales Evolutivas

Los Algoritmo Evolutivos resulta una herramienta fundamental para resolver problemas para los cuales la respuesta esperada varía en función de estados anteriores. Su combinación con las redes neuronales (RNA) ha dado lugar a un nuevo paradigma denominado Redes Neuronales Artificiales Evolutivas (RNAE) o simplemente Neuroevolución (Bruce, 2001).

Las redes neuronales artificiales evolutivas se refieren a un tipo de redes neuronales artificiales a la que se le aplican técnicas evolutivas dentro del proceso de diseño y aprendizaje de la red.

De forma general se puede afirmar que los algoritmos evolutivos se utilizan en el contexto de las redes neuronales en tres diferentes niveles: estimación de pesos de las conexiones, arquitecturas e implantación de reglas de aprendizaje (Yao, 1999). La estimación de los pesos de las conexiones introduce un método que sustituye o complementa a los métodos clásicos de optimización basados en el gradiente descendente que a menudo suelen quedar atrapados en mínimos locales y que ofrecen algunas dificultades cuando se aplican a redes neuronales recurrentes (Yao, 1999). Por otra parte, el diseño de la estructura de la red permite a la red adaptar su topología a diferentes tareas permitiendo que la red se adapte fácilmente en entornos dinámicos, por lo que la intervención humana es mínima dentro del proceso de aprendizaje. La implantación de las reglas de aprendizaje permite que el propio sistema “aprenda a aprender” y el proceso de búsqueda sea más eficiente y eficaz.

La utilización de una RNA para resolver un problema requiere establecer una serie de parámetros, que influyen decisivamente en la rapidez del aprendizaje y en la capacidad de aproximación de la red. En el caso de una red neuronal con una función con base radial los parámetros principalmente buscados son el número de neuronas presentes en la capa oculta y la configuración de dichas neuronas. Cada neurona se caracteriza por un punto central, radio de aplicación y una función con base radial a aplicar. Una vez fijados estos parámetros se pueden determinar analíticamente los pesos de las conexiones entre las neuronas de la capa oculta y las neuronas de la capa de salida.

La búsqueda de una red neuronal que permita resolver un problema dado puede considerarse como un problema de optimización en el sentido de que, de todas las posibles redes que se puedan crear, habrá un grupo de ellas que resuelvan de forma inmejorable el problema. Para las redes neuronales, además del mecanismo de aprendizaje propio de cada una de ellas, se puede diseñar un algoritmo de evolución (AE) que suponga un mecanismo adicional en el proceso de configuración y adaptación de las mismas.

Las técnicas de búsqueda global, como los AE, exploran amplias zonas del espacio de soluciones para determinar donde se hallan las más adecuadas. En general son menos eficientes en términos de tiempo y memoria necesaria que las técnicas de búsqueda local encontrando óptimos locales, por lo que muchas ocasiones es adecuado dejar que el AE seleccione soluciones en buenas áreas del espacio de búsqueda, para posteriormente localizar los óptimos locales en dichas áreas.

Una característica distinguible de las RNAE es su capacidad de adaptabilidad a un ambiente dinámico ya que, además del entrenamiento, cuentan con la evolución como mecanismo de aprendizaje. Han demostrado ser una excelente herramienta para la resolución de tareas complejas, entendiendo como tales aquellas cuya solución no es directa sino que involucra el aprendizaje de una estrategia para lograr el objetivo esperado. Tal es el caso de un robot que debe trasladar distintos tipos de elementos dentro de un escenario con obstáculos (Blickle y Thiele, 1995).

También es importante considerar que existen situaciones que no pueden ser resueltas por un único agente, como ocurre en el juego de fútbol robótico donde varios jugadores combinan sus acciones para lograr un único objetivo. Es importante notar que, más allá de las diferencias entre los agentes, es el grupo el que debe llevar a cabo la estrategia. Diversas investigaciones han demostrado que este tipo de situaciones pueden ser resueltas dividiendo el problema original en partes más simples, llamadas subtareas, permitiendo de esta forma un aprendizaje gradual de la respuesta buscada. En esta dirección se han desarrollado distintas soluciones que combinan técnicas de Evolución Incremental con Redes Neuronales Evolutivas (Tomassini, 1995) con el objetivo de proveer un mecanismo adaptivo que minimice el conocimiento previo necesario para obtener un buen desempeño dando lugar a controladores formados por varias redes (Bruce, 2001). Otro aspecto a tener en cuenta es la forma de determinar cuál es la red neuronal que debe ejecutarse en cada instante de tiempo; en esta línea existen diferentes alternativas que van desde el uso de un árbol de decisión diseñado ad-hoc hasta mecanismos que organizan la estructura en forma automática.

Estimación de los pesos de las conexiones

Dada una estructura de red, el entrenamiento de los pesos de las conexiones se formula en base a la minimización de la función de error. Los principales algoritmos de entrenamiento se basan en el descenso del gradiente de la función de error: retropropagación (BP) y en el gradiente conjugado (Hertz y col., 1991). Dichos algoritmos se han utilizado con éxito en distintas aplicaciones y en diferentes áreas (Knerr y col., 1992; Lang y col., 1990).

Aunque son muy eficientes, suelen quedar atrapados en mínimos locales dependiendo fuertemente del punto de partida de la búsqueda. Además, estos algoritmos no se pueden utilizar cuando la función de error es multimodal y/o no diferenciable (Hertz y col., 1991). Por ejemplo, en el caso de que la función de error tenga en cuenta tanto el error cometido entre los datos de salida y los estimados como la complejidad de la red.

Por su parte, los algoritmos evolutivos realizan una búsqueda global del óptimo de forma más efectiva y pueden trabajar con superficies de error multimodal y no diferenciable. Al no necesitar ninguna información relacionada con el gradiente de la función de error, resultan métodos bastante útiles cuando esta información no está disponible. Este hecho es el que ha motivado la utilización de redes neuronales evolutivas en la resolución de numerosos problemas reales en los que con frecuencia la función de error es multimodal y presenta problemas de continuidad y diferenciables. Por su parte, los algoritmos basados en el gradiente (retro-propagación y gradiente conjugado) suelen ser más rápidos en la búsqueda del óptimo que el entrenamiento evolutivo.

Sin embargo, los métodos evolutivos son en general menos sensibles a las condiciones iniciales del entrenamiento.

Recientemente, ha aparecido una nueva metodología que combina los algoritmos evolutivos y los métodos de búsqueda local (Kinnebrock, 1994; Skinner y Brougthon, 1995). En el contexto de las redes neuronales, esa metodología lleva a cabo un entrenamiento híbrido, combinando la capacidad de buscador global de un algoritmo evolutivo con la capacidad de afinar la solución que tienen los algoritmos de búsqueda local como los basados en el gradiente.

Representación real

Por otra parte, otra posibilidad consiste en la representación de los pesos de las conexiones de la red directamente mediante números reales, (Angeline y col., 1994; García-Pedrajas y col., 2002; Yao y Liu, 1997). Esta codificación se denomina codificación real. De esta forma, la red se suele representar mediante un vector de números reales.

Operador de mutación

El operador de mutación, como se ha comentado anteriormente, está directamente condicionado por el tipo de representación que se utilice. En el caso de utilizar representación real no se pueden aplicar los operadores de mutación y cruce como se entiende tradicionalmente desde el punto de vista de los algoritmos genéticos.

Los paradigmas de la computación evolutiva más adecuados cuando se utiliza la representación real son la Programación Evolutiva y las Estrategias Evolutivas ya que son especialmente eficientes en la optimización en dominios continuos. El principal operador de búsqueda es la mutación Gaussiana aunque también se utilizan otros tipos de mutación como puede ser la mutación de Cauchy (Fogel, 1994).

Básicamente, se trata de añadir valores aleatorios a los pesos de las conexiones siguiendo distribuciones Normales o de Cauchy y aceptar los cambios en caso de mejorar la aptitud de la red. No obstante, se utilizan técnicas de enfriamiento simulado. Para permitir con cierta probabilidad aceptar cambios aunque no se mejore la aptitud de la red, evitando así que el algoritmo quede atrapado en mínimos locales (Souto y col., 2002).

Operador de cruce

El operador de cruce genera descendientes recombinando el material genético (genotipo) de dos individuos de la población sin tener en cuenta el significado de dicho material.

De esta manera, la aproximación al problema se hace desde una perspectiva dual. Por un lado, las soluciones se muestran mediante cadenas de números y es en este espacio donde se realiza la búsqueda. Por otro lado, la evaluación de los

individuos se realiza en el espacio específico del problema. Esta característica, que resulta muy interesante en muchos problemas, puede tener efectos negativos en el problema de la evolución de la arquitectura de una red neuronal.

El primer efecto negativo que presentan es la restricción del espacio de búsqueda intrínseca a la representación de la red neuronal como una cadena finita de números. El subconjunto del espacio de búsqueda será más o menos amplio dependiendo de la representación que se elija. Generalmente, la representación se elige para que el operador de cruce funcione efectivamente. No obstante, existen entornos en los que el operador de cruce no es efectivo. Dichos entornos se denominan engañosos.

Para algunos autores el entorno de evolución de las redes neuronales es un entorno engañoso. Por lo tanto, no es apropiado para la aplicación del operador de cruce. Se argumenta que existen tres formas de engaño en la evolución genética de las redes neuronales (Angeline y col., 1994).

La primera forma de engaño radica en la existencia de redes que comparten su estructura y pesos y sin embargo su forma de representación es diferente. La función de interpretación no es biyectiva, sino que es de muchos uno. El cruce de dos redes iguales con diferente representación tiende a repetir componentes en vez de complementar. Este problema se conoce como el problema de la permutación. No obstante, aunque la mayoría de los estudios apuntan la gravedad del problema, otros no lo consideran tan grave (Hancock, 1992). Incluso se han desarrollado esquemas de codificación que evitan dicho problema.

La segunda forma de engaño viene dada por la existencia de redes con la misma topología pero pesos diferentes. Para una tarea dada, cada topología de red puede implementar soluciones múltiples, cada una correspondiente a una representación distribuida diferente a lo largo de los nodos ocultos (Hinton y col., 1986). Aunque se pueden eliminar nodos sin alterar drásticamente el rendimiento de la red, el papel que juega cada nodo en la representación global está determinado únicamente por el peso de sus interconexiones. Es decir, una unidad oculta no se puede considerar aislada y ser intercambiada libremente, porque su rendimiento depende tanto de sus propios pesos, como de las interconexiones con otras unidades de la red en la que se encuentra.

La tercera fuente de engaño aparece cuando se tiene redes topológicamente diferentes. Los tipos de representación distribuida que pueden adoptar redes diferentes topológicamente varían enormemente. Es probable que estas representaciones sean incompatibles reduciendo mucho la probabilidad de obtener individuos viables si se cruzan. Por ello, la evolución ha de evitar diversificar demasiado el tipo de redes de la población. Esto reduce de forma drástica la variedad de topologías que se pueden considerar en la búsqueda

Diseño de la arquitectura de la red

En la sección anterior se realizó el supuesto en el cual se tiene una arquitectura de red fija y óptima. A partir de dicha arquitectura se intenta ajustar los pesos de las conexiones de la red. No obstante, otro elemento importante dentro del entrenamiento de las redes es el diseño de la estructura de la red y la definición de las funciones de transferencia de los nodos. Esta tarea tradicionalmente la ha realizado el experto, mediante su experiencia y ensayando con diferentes estructuras hasta encontrar una adecuada. Esta tarea aparte de ser tediosa, no es sistemática, y por tanto, está abocada a caer en mínimos locales; redes pueden ser demasiado pequeñas con poca capacidad de aprendizaje, o redes demasiado complejas que sobreentrenan.

Se han realizado diferentes esfuerzos para automatizar esta tarea y hacer el proceso de aprendizaje menos supervisado de lo que es. Una alternativa ha sido el diseño de algoritmos constructivos que intentan, a partir de una red sencilla, añadir nodos en las capas ocultas y conexiones para ir mejorando la red (Frean, 1990). Otro enfoque ha sido justo el contrario, es decir, algoritmos destructivos que, a partir de redes complejas, van quitando nodos y conexiones que sean innecesarios hasta llegar a una red más simple y con capacidad de aprendizaje.

No obstante Angeline y colaboradores afirma que estos métodos están abocados a caer en mínimos locales, entre otras cosas porque al partir de una red determinada se reduce el espacio de búsqueda (Angeline y col., 1994).

Desde el punto de vista evolutivo, el diseño óptimo de la arquitectura de la red puede ser abordado como un proceso de búsqueda en el espacio de las arquitecturas en el que cada punto se refiere a una estructura distinta, y dado un

determinado criterio, como el de la complejidad y/o error de entrenamiento formar una superficie de error. Por tanto, buscar la mejor estructura será buscar el mejor punto de la superficie. Miller y colaboradores describen las características de esta superficie de error (Miller y col., 1989).

En primer lugar, la superficie es infinita ya que el número de nodos y conexiones posibles en la red no es finito. La superficie no es continua y por tanto no derivable ya que los cambios en la estructura son discretos. La superficie es compleja, ya que la relación entre la estructura y su aptitud es un proceso complejo y depende del método de evaluación, entrenamiento utilizado y de las condiciones iniciales para el proceso de entrenamiento. El entorno es engañoso ya que arquitecturas parecidas pueden tener distinta aptitud. Además, la superficie de error es multimodal ya que estructuras distintas pueden tener aptitud parecida.

Como en el caso de la estimación de pesos, se nos presentan dos problemas a la hora de diseñar la estructura de una red neuronal. Por una parte, la representación elegida y por otra los operadores de mutación y/o de cruce que se van a utilizar. En el caso de la representación de la estructura hay que decidir la información a codificar en el cromosoma. Es posible codificar todos los detalles de la red, en tal caso hablaríamos de codificación directa o, sin embargo, codificar ciertos parámetros de la red como puede ser el número de conexiones, número de capas ocultas, número de nodos en la capa oculta, etc. En tal caso hablaríamos de codificación indirecta.

Codificación directa

Numerosos trabajos utilizan codificación directa para diseñar arquitecturas de red (Angeline y col., 1994; García-Pedrajas y col., 2002; Yao y Liu, 1997).
Este diseño se puede realizar conjuntamente con la estimación o aprendizaje de los pesos, o hacerlo independientemente. La codificación utilizada para representar la topología de una red se hace mediante matrices binarias.

Se supone que se cuenta con una red de N nodos, la matriz $C = (cij)_{N\times N}$ representaría la topología de la red de tal manera que si $c_{ij} = 1$ indica que existe una conexión del nodo *i* al nodo *j* . En caso de que $c_{ij} = 0$ indica la ausencia de conexión del nodo *i* al nodo *j*. Para los casos en que se determina la estructura y los pesos de

las conexiones de manera conjunta, es posible codificar el valor de los pesos directamente en la matriz. De esta manera $c_{ij} = \alpha$ donde $\alpha \in R$, es el valor del peso de la conexión. Si α = 0 no existe conexión del nodo *i* al nodo *j*. Cada matriz *C* representa biunívocamente la estructura de una red.

Este tipo de representación no evita el problema de la permutación, por tanto se evita el uso del cruce a la hora de evolucionar redes codificadas mediante este método (Angeline y col., 1994). No obstante, Hancock asegura que el problema de la permutación no es tan importante como pudiera parecer, puesto que a lo más produce una cierta ineficiencia (Hancock, 1992). Existen algunos trabajos que intentan evitar este problema aunque no se han llegado a aplicar a problemas reales (Thierens, 1996).

Codificación indirecta

Con el objetivo de minimizar la longitud del cromosoma que representa a la estructura de la red, se han utilizado otros métodos de codificación denominados indirectos ya que su objetivo no es representar fielmente la estructura de la red sino algunas características importantes que la identifiquen.

Un tipo de representación indirecta puede ser la representación paramétrica, donde una red se puede representar como un conjunto de parámetros: número de capas ocultas, número de nodos en cada capa, número de conexiones entre capas, etc. Un ejemplo de este tipo de representación lo podemos en encontrar en publicaciones referentes al tema (Harp y col., 1989). Otro tipo de representación indirecta puede ser mediante gramáticas, compuestas por reglas de producción para construir arquitecturas. En la publicación de Kitano, de1990 se puede ver un ejemplo de este tipo de representación donde se utiliza un operador de cruce, que minimiza el problema de la permutación.

Evolución de las funciones de transferencia de los nodos

Hasta ahora, hemos considerado fijo el comportamiento de los nodos de la red, dejando al experto la elección de la función de transferencia a utilizar en cada nodo o en cada capa de la red. Sin embargo, es posible evolucionar igualmente el comportamiento de cada nodo de la red. En la publicación de White y Ligomenides,

de 1993 se pueden observar métodos donde se evoluciona tanto la estructura de la red como las funciones de transferencia de los nodos de la red. De esta forma, para cada red se establece inicialmente un 80% de nodos que utilizan la función de transferencia sigmoide, y un 20% que utilizan función de transferencia gaussiana. Durante la evolución se van generando redes con mezcla de nodos. Cohen y Intrator en 2002, introducen una metodología de entrenamiento para una red híbrida de redes MLP/RBF, donde se utiliza una aproximación de selección de modelos utilizando el principio MDL19. Otros autores mediante co-evolución también han co-evolucionado tanto estructuras, como pesos de conexiones y funciones de transferencia (García-Pedrajas y col, 2002).

2.7 Polímeros

Los polímeros son un tipo particular de macromolécula, que se caracteriza por tener una unidad que se repite a lo largo de la molécula.

Las pequeñas moléculas que se combinan entre sí mediante un proceso químico, llamado reacción de polimerización, para formar el polímero se denomina monómero. La unión de todas estas pequeñas moléculas da lugar a una estructura de constitución que se repite regularmente a lo largo de toda la molécula, se conoce con el nombre de unidad constitucional repetitiva (ucr) o unidad monomérica.

La longitud de la cadena del polímero viene determinada por el número de ucr que se repiten en la cadena. Esto se llama grado de polimerización (X), y su peso molecular viene dado por el peso de la unidad constitucional repetitiva multiplicado por el grado de polimerización.

En un determinado polímero, si todas las unidades estructurales son idénticas este se llama homopolímero, pero si este procede de dos o más monómeros recibe el nombre de copolímero (Morrison y Boyd, 1998).

Síntesis y mecanismos de reacción

Las reacciones de polimerización son muy variadas y sus mecanismos de reacción obedecen a la estructura química de los monómeros que les dan origen. Por lo tanto, la mayoría de esto mecanismos, son los mismos que se observan en

las reacciones químicas de moléculas orgánicas sencillas.

Clasificación y características de los procesos de polimerización

Los procesos de polimerización fueron clasificados originalmente por Carothers en 1929 como polimerización por condensación y adición, basándose en la comparación de la fórmula molecular de los polímeros obtenidos con la de los monómeros de los cuales fueron formados.

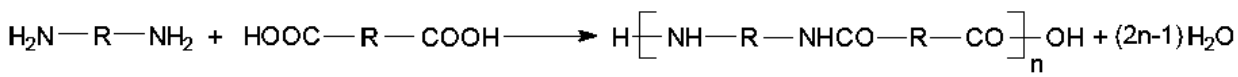
Posteriormente Flory en 1953 proporcionó una nueva base para la clasificación, de acuerdo al mecanismo de la polimerización, definiéndolos como polimerización en etapas y polimerización en cadena (Wade, 1993).

En la actualidad los términos condensación y etapas así como adición y cadena son usados sinónimamente.

Las características generales de la polimerización en etapas (condensación) son las siguientes:

a. La polimerización transcurre mediante reacción entre grupos funcionales, usualmente de distinta naturaleza, tales como hidroxilo (-OH), cloruros de acilo (-COCl), carboxilo (-COOH), amina ($-NH_2$), etc y por lo general con eliminación de una molécula pequeña.
b. El grupo funcional resultante de la reacción de los grupos funcionales de los monómeros forma parte de la cadena principal del polímero, repitiéndose ininterrumpidamente a lo largo de ella.
c. En cualquier instante a lo largo de la polimerización, la mezcla de reacción consiste en una distribución continua de tamaños moleculares que comprende desde el mismo monómero hasta polímero de elevado peso molecular.

A continuación se muestra un ejemplo de este tipo de reacción:



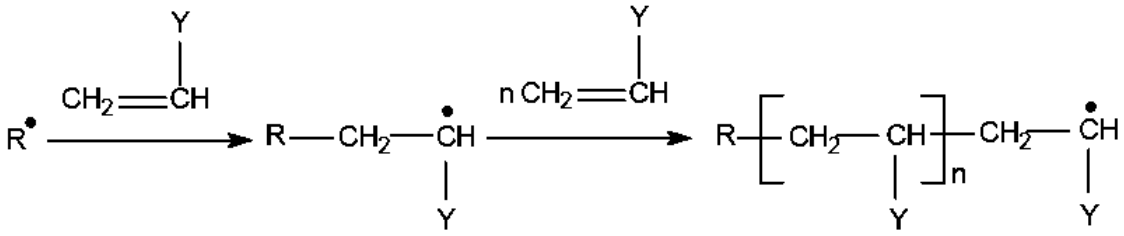
Por su parte las características más relevantes de la polimerización en cadena (adición) se resumen a continuación:

- La polimerización transcurre mediante la adición continua de monómero a una cadena en crecimiento, que contiene un extremo activado hasta el

momento de su terminación.

- La reacción transcurre sin pérdida de materia, por lo que la unidad constitucional repetitiva del polímero y el monómero presentan una estequiometría idéntica.
- En cualquier instante a lo largo de la polimerización, la mezcla de reacción tiene una composición constituida por monómero y polímero de elevado peso molecular.

Un ejemplo es la polimerización vinílica:



Dependiendo del tipo de mecanismo, la evolución del peso molecular promedio del polímero que se genera es claramente diferente. En la poliadición, las cadenas adquieren sus tamaños finales desde el comienzo de la reacción, por lo que el peso molecular apenas varía con la conversión. En la policondensación, las cadenas están continuamente creciendo por combinación de otras más cortas, es decir, los primeros productos son los dímeros, después los trímeros, los tetrámeros y finalmente después de una serie de pasos los polímeros, por lo que el peso molecular crece exponencialmente con la conversión. Así por ejemplo, en la polimerización en cadena a cualquier tiempo de polimerización se encuentra polímero de alto peso molecular y monómero, mientras que en la polimerización por etapas solo es posible encontrar polímero de alto peso molecular cerca del final de la polimerización, cuando las conversiones son mayores del 92%.

Polimerización por radicales libres

Los monómeros vinílicos, compuestos que contienen dobles enlaces, pueden polimerizar en presencia de peróxidos en condiciones en que estos puedan generar radicales libres. Algunos ejemplos de ellos se muestran en la Figura 2.11 (Muñoz, 1997):

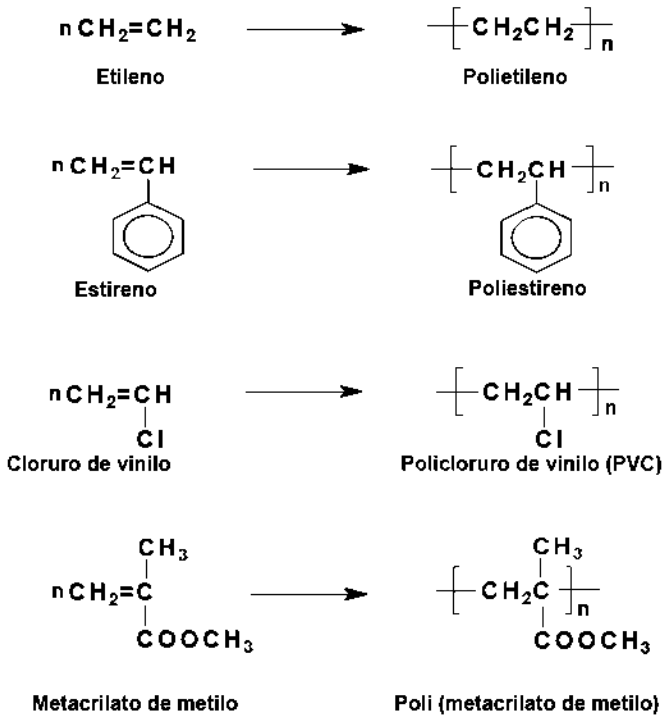


Figura 2.11 Modelos de Polimerización.

Mecanismos de polimerización

La polimerización implica la adición de radicales libres al doble enlace del monómero y se lleva a cabo mediante tres etapas bien diferenciadas: iniciación, propagación y terminación.

a) Iniciación:

Esta fase involucra la creación del centro activo del radical libre y normalmente tiene lugar en dos pasos. El primero es la formación de radicales libres a partir del iniciador y el segundo es la adición de uno de estos radicales libres a una molécula de monómero:

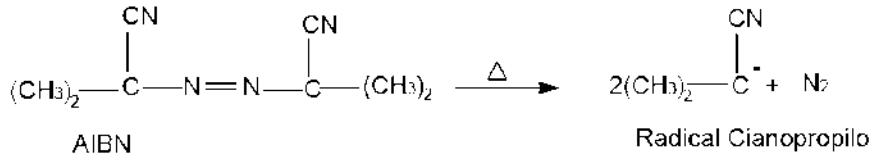
$$I \xrightarrow{k_d} nR^{\bullet}$$

$$R^{\bullet} + M \xrightarrow{k_a} RM^{\bullet}$$

Donde I representa el iniciador, R· al radical libre formado en la descomposición del primero, M el monómero y k_d y k_a las constantes de descomposición del iniciador y de iniciación respectivamente.

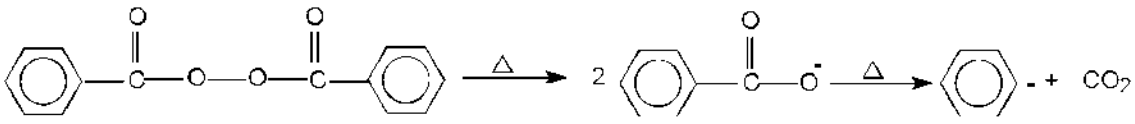
Los radicales se pueden generar mediante la descomposición térmica o

fotoquímica de sustancias como peróxido de benzoílo (PB) o del azobisisobutironitrilo (AIBN) tal como se muestra a continuación:



AIBN

Radical Cianopropilo



Peróxido de Benzoilo

Radical Benzoiloxilo

Radical Fenilo

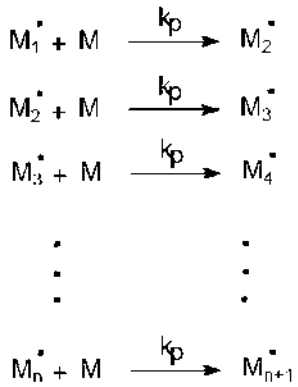
En la Tabla 2.1 se recogen algunas sustancias empleadas como iniciadores en la polimerización radical así como sus intervalos de aplicación óptimos en función de la temperatura (Odian, 1970).

Tabla 2.1 Tiempo de vida media en función de la temperatura.

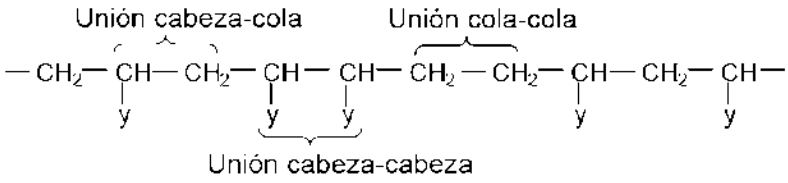
Iniciador	50°C	60°C	70°C	85°C	100°C	115°C	130°C
Azobisisobutironitrilo	74 h	17 h	4,8 h		7,2 min		
Peróxido de benzoilo	86 h	7 h	7,3 h	1,4 h	19,8 mi		
Peróxido de acetilo	158 h		8,1 h	1,1 h			
Peróxido de laurilo	47,7 h	12,8 h	3,5 h	31			
t-butil peracetato				88 h			
Peróxido de cumilo						113 h	1,7 h
peróxido terc-butilo					218 h	34 h	6,4 h
Hidroperóxido terc-					338 h		

b) Propagación

En esta etapa se van añadiendo moléculas de monómero al monómero radical formado en la etapa de la iniciación y la cadena va creciendo tal como se indica; siendo k_p la constante de propagación:



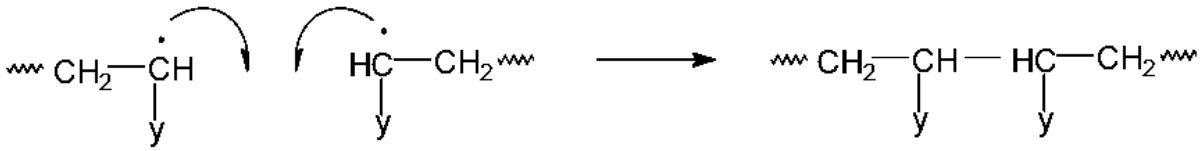
Si se examina la estructura de un polímero generado por un monómero vinílico, como los aquí descritos, se pueden evidenciar varias posibilidades de arreglos con características en las configuraciones. Estos arreglos son la consecuencia de que las unidades monoméricas pueden incorporarse a la cadena mediante uniones cabeza-cola o formando uniones cabeza-cabeza y cola-cola tal como se muestra a continuación:



Es de notar que la mayoría de las polimerizaciones incorporan al monómero con una orientación cabeza-cola, debido a aspectos estéricos y electrónicos.

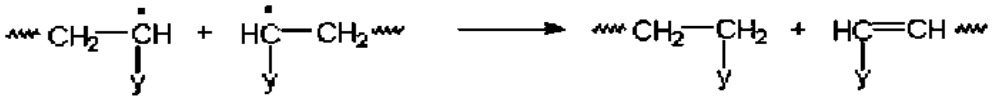
c) Terminación

En esta etapa se termina el crecimiento de la cadena del polímero. Los dos mecanismos más comunes de la terminación implican la reacción bimolecular de las cadenas crecientes del polímero. La combinación involucra el acoplamiento de dos cadenas crecientes para formar una sola molécula de polímero.



Alternativamente otro átomo de hidrogeno puede ser abstraído por otra

cadena creciente generando de esta manera la reacción conocida como desproporción:



Así se forman dos tipos de moléculas, una con un extremo saturado y la otra con un extremo insaturado; en este caso las cadenas tienen moléculas con fragmentos iniciadores solamente en un extremo, mientras la combinación da como resultado moléculas con fragmentos iniciadores en ambos extremos.

En general ocurren ambos tipos de reacciones de terminación pero en diferentes magnitudes, dependiendo del monómero y de las condiciones de polimerización.

Polímeros comerciales

Los polímeros sintéticos pueden clasificarse de una manera general como plásticos, fibras y elastómeros y las características de cada uno se resumen en la Figura 2.12.

En esta sección se realiza una breve descripción de algunos de los materiales más comunes en función de su clasificación. Es importante tener en cuenta que muchos polímeros pueden quedar clasificados en más de una categoría dadas sus características particulares.

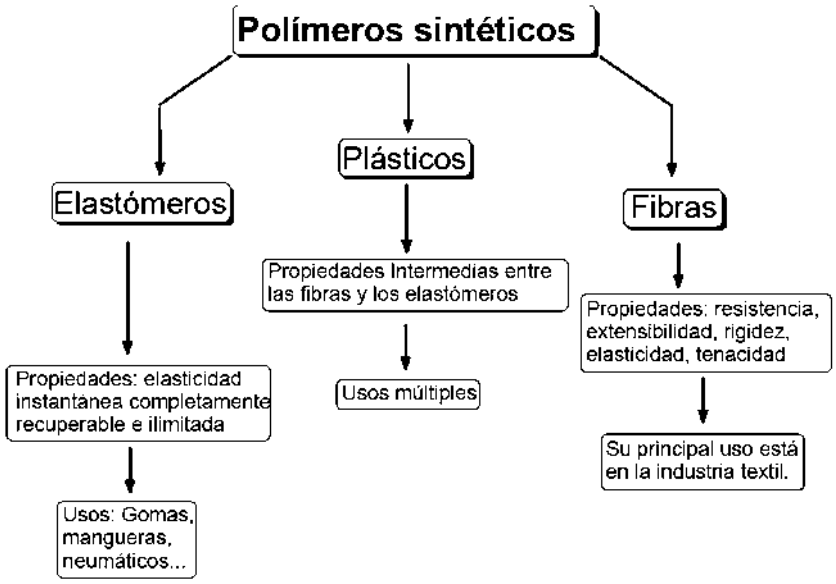


Figura 2.12 Clasificación de los Polímeros Sintéticos

Los plásticos son materiales cuyas propiedades son intermedias entre los elastómeros y las fibras. Estos materiales tienen una infinidad de aplicaciones que se pueden dividir de acuerdo a ellas como:

Plásticos de uso general, también llamados “comodities”, son materiales que se fabrican en grandes cantidades a bajo costo y son empleados en múltiples aplicaciones en la vida diaria como pueden ser recipientes, enseres domésticos, juguetes, etc.

Plásticos de ingeniería, su volumen de producción es menor y su precio más elevado. Se caracterizan por tener propiedades particulares para aplicaciones específicas. Estos plásticos pueden competir con los materiales metálicos o cerámicos a los que aventajan por su menor densidad y facilidad de procesado. Encuentran mucha aplicación en la industria automotriz.

Plásticos avanzados, son materiales que se diseñan con una constitución molecular definida para satisfacer una aplicación concreta. Estos materiales tienen propiedades excepcionales que los califican como polímeros de vanguardia para el futuro. Entre las propiedades más relevantes de ellos destacan la biocompatibilidad y la formación de fases cristal líquido.

Algunos ejemplos de plásticos de aplicación industrial y comercial son los siguientes:

Polietileno (PE)

Este polímero se obtiene a partir del etileno. Las dos variedades comerciales más conocidas de este polímero son el polietileno de baja densidad (LDPE) y el de alta densidad (HDPE). La diferencia en sus propiedades y aplicaciones vienen dadas por el grado de cristalinidad que cada uno puede alcanzar.

El LDPE, que posee una estructura muy ramificada y por ende una baja cristalinidad. Sus principales aplicaciones son la fabricación de bolsas plásticas, tuberías y recubrimiento para cables. Por su parte el polietileno de alta densidad, que posee un mayor cristalinidad debido a su estructura prácticamente lineal, encuentra aplicaciones como tuberías, recipientes, enseres domésticos, aislamiento para cables, juguetes y asientos para uso público, entre otras.

Polipropileno (PP)

El polipropileno se obtiene mediante la polimerización del propileno. Cuando ésta se lleva a cabo por procesos de Ziegler-Natta se obtiene un polímero altamente estereorregular con un contenido de al menos 90% de polímero isotáctico. En términos generales las propiedades del PP son similares a las del HDPE. Se emplea para la elaboración de tubos, fibras para cuerdas, artículos textiles y películas para empaque de alimentos.

Poliestireno (PS)

La polimerización industrial del estireno se lleva a cabo mediante radiales libres con la ayuda de peróxidos. El poliestireno obtenido de esta manera es fundamentalmente atáctico. Existen tres tipos de poliestireno comercial: el poliestireno de alto impacto, empleado por ejemplo, en la fabricación de vasos plásticos desechables, el poliestireno cristal que se emplea en la fabricación de recipientes y el polestireno expandible (anime) que se usa entre otras cosas como material de empaque.

La copolímerización del estireno con butadieno produce un caucho sintético con propiedades análogas al caucho natural.

Polimetilmetacrilato (PMMA)

El PMMA al igual que otros polímeros vinílicos es un material amorfo y su propiedad más destacada es su excelente transparencia lo que hace que una de sus principales aplicaciones sea como sustituto del vidrio.

Policloruro de vinilo (PVC)

Muchos autores consideran el PVC como el plástico más versátil y su producción es solo superada por la del polietileno. Sus usos abarcan desde la construcción de casas hasta prendas de vestir. Se emplea en productos de calandrado, fabricación de tubería, dispositivos de uso médico, etc, etc.

Politetrafluoroetileno (Teflón)

Este es un material tenaz, flexible y de gran resistencia química y térmica, es además un excelente aislante térmico. Su uso se restringe a aplicaciones técnicas tales como sellantes, aislante eléctrico, recubrimientos inertes y valvulería.

Poliamidas y poliésteres

Estos materiales tienen su principal aplicación en la fabricación de fibras, sin embargo, muchos de ellos debido a su versatilidad, pueden ser usados en la fabricación de piezas de plástico tal como podemos ver en los siguientes ejemplos:

El *nylon-6,6* es un material industrial que se usa en la fabricación de rodamientos y engranajes. En general, los plásticos de poliamida se usan para la fabricación de componentes y partes para automóviles y camiones.

El *polietién tereftalato (PET)* se emplea en la fabricación de botellas de refresco y películas para envoltorios.

Plásticos termoestables

Las Resinas fenol formaldehído se preparan por una reacción de condensación entre el fenol y el formaldehído que forman polímeros con un grado de entrecruzamiento que puede ser controlado.

Su principal aplicación es la producción de piezas eléctricas de muy diferente uso. Las Resinas urea formaldehído tienen aplicaciones similares a las anteriores y muchas veces se prefieren a las primeras cuando la presentación del material es un factor a considerar.

Las resinas epoxi, los poliuretanos y los poliésteres insaturados, también son materiales plásticos con infinidad de aplicaciones.

2.8 Instrumentación Virtual

Para poder llevar a cabo el procesamiento de las señales, se suelen utilizar micro-controladores con interfaces de comunicación hacia la PC, dichos micro-controladores son los encargados de digitalizar las señales, procesar la información de acuerdo el protocolo utilizado, almacenar algunos datos en sus memorias internas e intervienen en los casos en que se deba realizar alguna acción de control.

Sistemas de adquisición de datos

La adquisición de datos se inicia con el fenómeno físico o la propiedad física del objeto que se desea medir. Esta propiedad física o fenómeno puede ser el cambio de temperatura en una habitación o equipo de reacción, la intensidad del

cambio en el cambio de una fuente luminosa, la presión dentro de una cámara, la fuerza aplicada a un objeto, o cualquier cambio que se desee monitorear. Un eficaz sistema de adquisición de datos puede medir todos estos fenómenos o propiedades.

Generalmente los datos o variables que se han de captar son de carácter analógicos, mientras que sus tratamientos, almacenamiento y análisis son mucho más eficaces cuando se hacen digitalmente. Esto implica la insaturación de una serie de módulos electrónicos que permitan llevar acabo la transformación de los datos del campo analógico al campo digital, sin que por ello se pierdan los aspectos fundamentales para el proceso que se desea controlar.

Componentes de un sistema de adquisición de datos

Un sistema de adquisición de datos se compone de diferentes módulos electrónicos que permiten llevar a cabo la transformación de los fenómenos antes mencionados. Su estructura general se muestra en la siguiente figura.

Algunos de los elementos que forman el sistema de adquisición de datos se nombran a continuación junto con sus respectivas funciones:

- **Sensor o transductores.** Son los encargados en convertir la variable física a medir (temperatura, presión, humedad, etc.) en señal eléctrica. Esta señal eléctrica suele sr de muy bajo nivel, por lo que generalmente se requiere un acondicionamiento previo para conseguir así los niveles de tensión/corriente adecuados para el resto de los módulos del sistema.
- **Multiplexor.** Este módulo se encarga de seleccionar la señal de entrada que va a ser tratada en cada momento. En el caso en que solo deseáramos tratar con únicamente una señal, este módulo no es indispensable.
- **Amplificador de instrumentación.** La función de este bloque es amplificar la señal de entrada del sistema de adquisición de datos para que su margen dinámico se aproxime lo más posible al margen dinámico del conversor de A/D consiguiendo de esta forma la máxima resolución. En un sistema de adquisición de datos con varios canales de entrada, cada canal tiene un intervalo de entrada distinto, por lo que es necesario que este amplificador sea de ganancia programable.

- **S&H (Muestreo y Retención).** Este módulo es el encargado de tomar la muestra del canal seleccionado y mantenerla durante el tiempo que dura la conversión.

Configuraciones de los sistemas de adquisición de datos

Los sistemas de adquisición de datos se clasifican según el número de canales de entrada que posean, esta clasificación se muestra a continuación:

- **Sistemas monocanales**

  Es la configuración más general de un sistema de adquisición de datos responde al diagrama de bloques de la Figura 2.13 La señal procedente de la fuente de información (cuya obtención se realiza por medio de los sensores apropiados con sus correspondientes acondicionamientos de señal) es aplicada a la entrada del circuito amplificador de instrumentación, el cual adaptara el nivel analógico de la entrada al margen del convertidor A/D.

  Como se puede observar en la Figura 2.13, el sistema monocanal solo permite la adquisición de una señal de entrada, lo que permite optimizar su configuración para un tipo concreto de entrada analógica.

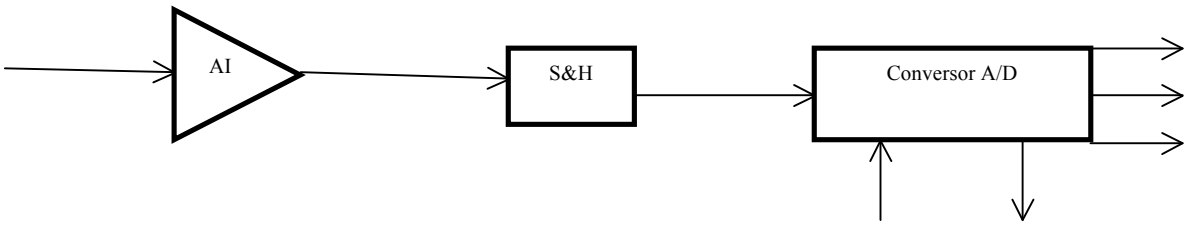


Figura 2.13 Sistema Mono canal

- **Sistemas de adquisición multicanal.**

Cuando se plantea la necesidad de realizar la conversión A/D de diversas señales, los sistemas de adquisición pueden tener diferentes aspectos en sus configuraciones. La configuración a utilizar depende de los siguientes aspectos:

- Las características de las señales de entrada (frecuencia, periodo, rangos, etc.)
- La información que se desea obtener de las señales.
- La velocidad de conversión que se desea tener.
- El costo que tendrá el sistema.

En un sistema de adquisición de datos multicanal pueden existir distintas configuraciones que van en función de cómo se realice la distribución de los módulos del sistema. Esta distribución depende de las necesidades de cada aplicación, como se muestra enseguida:

- *Sistema de adquisición multicanal con muestreo secuencial de canales*

  Es la configuración que menos componentes requiere y por lo tanto la más económica de todos los sistemas multicanal. Su estructura se muestra en la Figura 2.14. El funcionamiento del circuito es sencillo: primero se selecciona el canal de entrada al multiplexor y se fija la ganancia del amplificador de instrumentación, el circuito S/H pasa a modo sample (selección) hasta que se adquiere una muestra de la señal, momento en el que pasa a modo hold (retención), dando así la instrucción al convertidor A/D para que inicie la conversión. Una vez transcurrido el tiempo de la conversión el convertidor A/D lo indica mediante una señal de intensidad la cual es la señal de fin de la conversión.

  Esta configuración permite que durante el tiempo de conversión de un canal, se puede estar seleccionando en el multiplexor, simultáneamente, el siguiente canal a muestrear. Así el tiempo tope establecido del multiplexor no influirá en la velocidad de adquisición final del sistema, siempre y cuando dicho tiempo sea menor que el tiempo de conversión del convertidor A/D.

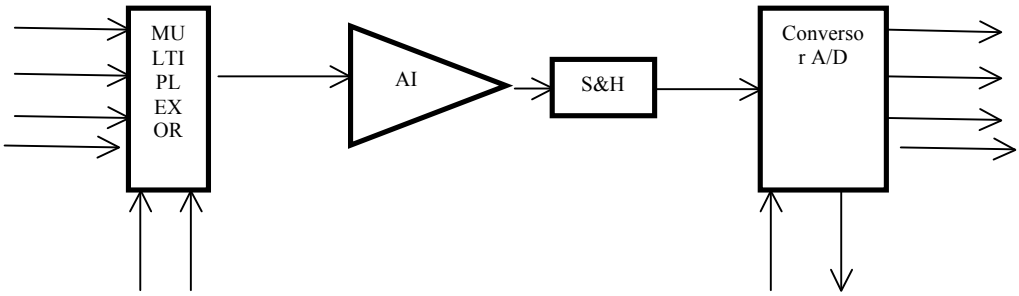


Figura 2.14 Sistema de adquisición multicanal con muestreo secuencial de canales

- *Sistema de adquisición multicanal con muestreo simultaneó de canales.*

  Esta configuración presenta la ventaja de que todos los circuitos de S/H de entrada conmutan simultáneamente a modo hold, manteniendo el valor de la muestra de cada señal de entrada hasta que el convertidor A/D puede realizar la conversión, cosa que no es posible en el modelo de muestreo secuencial. Su estructura se muestra en la Figura 2.15

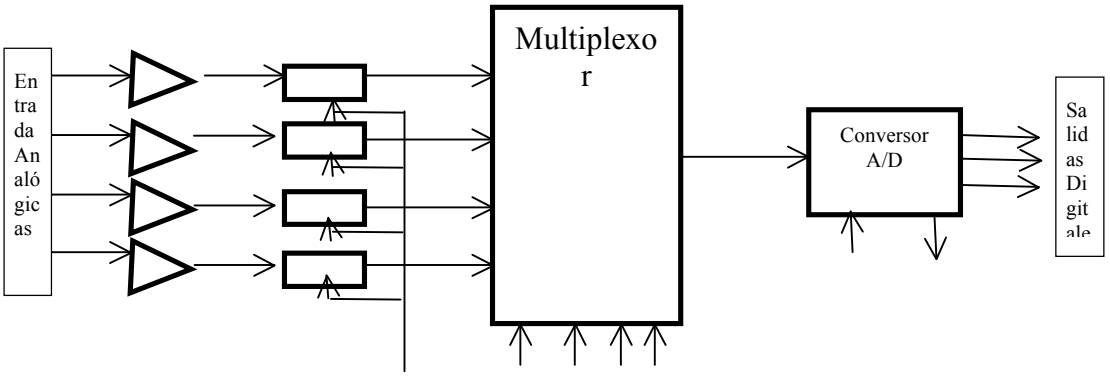


Figura 2.15 Sistema de adquisición multicanal con muestreo simultaneó de canales

- *Sistema de adquisición multicanal paralelo.*

  En este caso se puede decir que cada canal constituye un sistema de adquisición independiente con todos los elementos necesarios para realizar una conversión A/D completa, con la salvedad de que al utilizar, generalmente, un solo canal digital de salida es necesario incluir un multiplexor digital como lo muestra el siguiente diagrama. Se puede observar que el sistema ofrece una gran flexibilidad, en cada canal debido a que puede ser adaptado de forma independiente, según las necesidades requeridas por la señal a adquirir (ganancia del amplificador, velocidad de adquisición, etc.) otra ventaja adicional es que la velocidad del sistema se optimiza

optimiza notablemente, esto mediante la conversión simultanea de varias señales. Como desventaja de este sistema es su costo el cual es elevado.

Parámetros característicos de los sistemas de adquisición de datos.

Los parámetros que caracterizan un sistema de adquisición de datos son básicamente tres:

- **Número de canales.** Esto depende del número de señales que se desean adquirir datos
- **Exactitud de la conversión.** Esta impuesta por los circuitos utilizados, es decir, multiplexores, amplificadores, S/H, convertidores A/D, esencialmente. Así, a cada uno de estos módulos se requiere los mínimos.
- **Velocidad de muestreo.** Este parámetro especifica la velocidad a la que el sistema puede adquirir y almacenar muestras de las entradas. En general debemos identificar la velocidad de muestreo con el número de muestras por unidad de tiempo que pueden obtenerse de un canal. Los cuatro factores a tener en cuenta son:
  - Tiempo de establecimiento del multiplexor.
  - Tiempo de establecimiento del amplificador
  - Tiempo de adquisición de S/H
  - Tiempo del convertidor A/D

2.9 LabView

El entorno LabVIEW y la instrumentación virtual, LabVIEW (Laboratory Virtual Instrument Engineering Workbench), de National Instruments, es un sistema de programación gráfico diseñado para el desarrollo de distintas aplicaciones como el análisis de datos, la adquisición de datos y el control de instrumentos.

Al ser LabVIEW un lenguaje de programación gráfico y basado en un sistema de ventanas, muchas veces es más sencillo de utilizar que otros lenguajes más típicos.

Este tipo de lenguaje se desarrolló a partir de la aparición de la *instrumentación virtual*, es decir, con el uso de los ordenadores para realizar medidas (temperatura, presión, caudal, etc), aprovechando las características de éstos últimos (potencia de cálculo, productividad, capacidad de visualización gráfica y capacidad de conexión con otros dispositivos), para optimizar los resultados

En definitiva, se puede concluir diciendo que con un ordenador personal, un hardware adecuado (placas de adquisición de datos), unos “drivers” y un software como LabVIEW, se pueden obtener datos muy provechosos y mejores que si se utilizan instrumentos tradicionales tales como osciloscopios, generadores de señal, analizadores de espectros, analizadores vectoriales, etc.

Ventajas de usar Labview

- La primera ventaja de usar LabVIEW es que es compatible con herramientas de desarrollo similares y puede trabajar a la vez con programas de otra área de aplicación, como Matlab o Excel. Además se puede utilizar en muchos sistemas operativos, incluyendo Windows y UNIX, siendo el código transportable de uno a otro.
- Otra de las ventajas más importantes que tiene este lenguaje de programación es que permite una fácil integración con hardware, específicamente con tarjetas de medición, adquisición y procesamiento de datos (incluyendo adquisición de imágenes).
- Es muy simple de manejar, debido a que está basado en un nuevo sistema de programación gráfica, llamado *lenguaje G*.
- Es un programa enfocado hacia la instrumentación virtual, por lo que cuenta con numerosas herramientas de presentación, en gráficas, botones, indicadores y controles, los cuales son muy esquemáticos y versátiles. Estos serían complicados de realizar en bases como C++ donde el tiempo para lograr el mismo efecto sería muchas veces mayor.
-Es un programa que contiene librerías especializadas para manejos de DAQ (tarjetas de adquisición de datos), Redes, Comunicaciones, Análisis Estadístico, Comunicación con Bases de Datos (útil para una automatización de una empresa a nivel total).

- Como se programa creando subrutinas en módulos de bloques, se pueden usar otros bloques creados anteriormente como aplicaciones por otras personas.

A continuación se representa en la Tabla 2.2 algunas de las ventajas y desventajas de usar LabView:

Tabla 2.2 Ventajas y Desventajas de usar LabView.

Instrumento Tradicional	Instrumento Virtual
Definido por el fabricante	Definido por el usuario
Funcionalidad específica, con conectividad limitada.	Funcionalidad ilimitada, orientado a aplicaciones, conectividad amplia.
Hadware es la clave.	Sofware es la clave
Alto costo/función	Bajo costo/función, variedad de funciones, reusable.
Arquitectura “cerrada”	Arquitectura “abierta”
Lenta incorporación de nuevas tecnologías.	Rápida incorporación de nuevas tecnologías, gracias a la plataforma PC.
Bajas economías de escala, alto costo de mantenimiento	Altas economías de escala, bajos costos de mantenimiento.

Aplicaciones de LabVIEW

Labview tiene su mayor aplicación en sistemas de medición, como monitoreo de procesos y para aplicaciones de control. Además, LabVIEW se utiliza bastante en el procesamiento digital de señales, en el procesamiento en tiempo real de aplicaciones biomédicas, manipulación de imágenes y audio, automatización, diseño de filtros digitales, generación de señales, entre otras, etc.

Capítulo 3

Metodología

En este capítulo se presenta la metodología a seguir para las simulaciones del sistema de reacción a controlar (parte teórica), de la misma manera también se presenta la metodología usada en la experimentación de obtención de Polimetil metacrilato, así como la aplicación de la red neuronal evolutiva al control de temperatura de un reactor batch a nivel laboratorio.

3.1 Parte Teórica

En esta sección se mostrara la metodología para obtener los resultados teóricos.

3.1.1 Modelo matemático de un reactor batch

La reacción exotérmica de la polimerización por radicales libres de metil metacrilato que se lleva a cabo en un reactor de 4 L de acero inoxidable, de tipo batch.

Considerando el siguiente caso de estudio para el control de temperatura en un reactor batch (Figura 3.1).

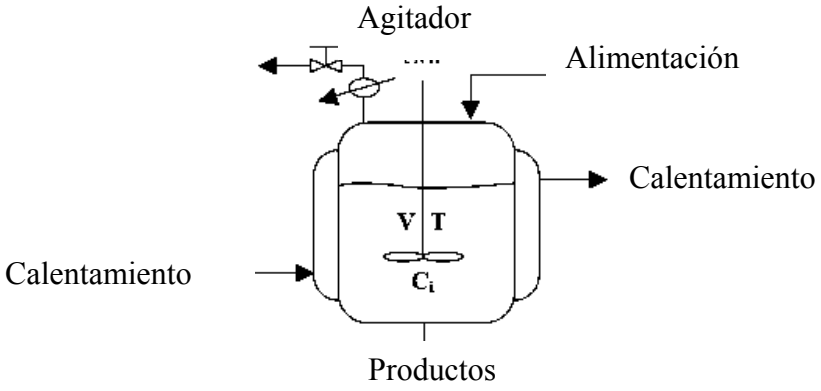


Figura 3.1 Reactor batch con Calentamiento/Enfriamiento.

Mecanismo para la reacción por radicales libres:

Iniciacion

$$I \xrightarrow{k_j} 2\dot{I} \tag{3.1}$$

$$\dot{I} + M \longrightarrow P_1 \tag{3.2}$$

Propagación

$$P_n + M \xrightarrow{k_p} P_{n+1} \tag{3.3}$$

Terminación por acoplacion de reaccion :

$$P_n + P_m \xrightarrow{k_{tc}} D_{n+m} \tag{3.4}$$

Terminación por disprotonacion de reactivos :

$$P_n + P_m \xrightarrow{k_{td}} D_n + D_m \tag{3.5}$$

Transferencia de anillo para reacciones del monomero :

$$P_n + M \xrightarrow{k_{fm}} D_n + P_1 \tag{3.6}$$

Considerando el estándar de cinética de polimerización por radical libre (Ray, 1972; Baillagouy Soong 1985b; Tirrell*y col*, 1987). Se hacen las siguientes consideraciones:

- Estado cuasi-estable aproximado e hipótesis del longitud de anillo limitado
- Todas las reacciones son elementales e irreversibles
- Las velocidades de reacción son independientes de la vida del polímero
- La velocidad de transferencia de anillo para las reacciones del solvente son despreciables comparados con otras reacciones

Con las consideraciones anteriores se toma la velocidad de reacción como:

$$k_d = 1.58 \times 10^{15} \exp\left(\frac{-1.2874 \times 10^5}{RT}\right) \tag{3.7}$$

$$k_{p0} = 7.0 \times 10^6 \exp\left(\frac{-2.6334 \times 10^4}{RT}\right) \tag{3.8}$$

$$k_{fm} = 4.661\times10^{9}\exp\left(\frac{-7.4479\times10^{4}}{RT}\right) \tag{3.9}$$

$$k_{fs} = 1.49\times10^{9}\exp\left(\frac{-6.6197\times10^{4}}{RT}\right) \tag{3.10}$$

$$k_{\theta p} = 3.0233\times10^{13}\exp\left(\frac{-1.1700\times10^{5}}{RT}\right) \tag{3.11}$$

$$k_{\theta t} = 1.4540\times10^{20}C_{i,0}\exp\left(\frac{-1.4584\times10^{5}}{RT}\right) \tag{3.12}$$

$$k_{p} = \frac{k_{p0}}{\left(1+\frac{k_{p0}}{Dk_{\theta p}}\right)} \tag{3.13}$$

$$k_{t} = \frac{k_{t0}}{\left(1+\frac{k_{t0}}{Dk_{\theta t}}\right)} \tag{3.14}$$

$$D = \exp\left[\frac{2.303\left(1-\phi_{p}\right)}{0.168-8.21\times10^{-6}\left(T-387\right)^{2}+0.03\left(T-\phi_{p}\right)}\right] \tag{3.15}$$

$$\phi_{p} = \frac{\mu_{1}MW_{m}}{\rho_{p}} \tag{3.16}$$

$$k_{fm} = Z_{fm}\exp\left(\frac{-E_{fm}}{RT}\right) \tag{3.17}$$

Balances de Masa

$$\frac{dC_{m}}{dt} = -\left(k_{p0}+k_{fm}\right)C_{m}\xi_{0} \tag{3.18}$$

$$\frac{dC_{i}}{dt} = -k_{d}C_{i} \tag{3.19}$$

Balance de Energía

$$\frac{dT}{dt} = \frac{(-\Delta H_r)R_p}{Cp\rho_{mix}} - \frac{UA(T - T_J)}{CpV\rho_{mix}} \tag{3.20}$$

$$\frac{dT_J}{dt} = \frac{(T_{JSP} - T_J)}{\tau_J} + \frac{UA(T - T_J)}{Cp_J V_J \rho_J} \tag{3.21}$$

Ecuaciones adicionales

$$\frac{d\xi_0}{dt} = 2fk_d C_i - k_t \xi_0^2 \tag{3.22}$$

$$\frac{d\xi_1}{dt} = 2fk_d C_i - k_p C_m \xi_0 + (k_{fm} C_m + k_{fs} C_s)(\xi_0 - \xi_1) - k_t \xi_0 \xi_1 \tag{3.23}$$

$$\frac{d\xi_2}{dt} = 2fk_d C_i - (2\xi_1 - \xi_0)k_p C_m + (k_{fm} C_m + k_{fs} C_s)(\xi_0 - \xi_2) - k_t \xi_0 \xi_2 \tag{3.24}$$

$$\frac{d\mu_0}{dt} = (k_{fm} C_m + k_{fs} C_s)\xi_0 + (0.5k_t)\xi_0^2 \tag{3.25}$$

$$\frac{d\mu_1}{dt} = (k_{fm} C_m + k_{fs} C_s + k_t \xi_0)\xi_1 \tag{3.26}$$

$$\frac{d\mu_2}{dt} = (k_{fm} C_m + k_{fs} C_s)\xi_2 + k_t \xi_0 \xi_2 + k_t \xi_1^2 \tag{3.27}$$

$$X_m = \frac{\mu_1}{C_{m,0}} \tag{3.28}$$

$$R_p = k_p C_m r_0 \tag{3.29}$$

$$r_0 = \left[\frac{2fk_d X_i}{k_t}\right]^{0.5} \tag{3.30}$$

$$\rho_{mix} = (\mu_1 + C_m)MW_m + C_s MW_s + C_i MW_i \tag{3.31}$$

$$Cp = \frac{\mu_1 Cp_p + C_m Cp_m + C_s Cp_s}{\mu_1 + C_m + C_s} \tag{3.32}$$

Calculo del coeficiente de transferencia de calor

Para el modelo matemático es necesario el cálculo del coeficiente de transferencia de calor global, el cual puede ser determinado mediante correlaciones de la siguiente manera.

$$\frac{1}{U} = \frac{1}{\alpha_k} + \frac{d}{\lambda}\frac{A_k}{A_m} + \frac{1}{\alpha_r}\frac{A_k}{A_r} \tag{3.33}$$

donde:

U = coeficiente global de transferencia de calor

A_k= Superficie de transferencia de calor en el lado de la mezcla reaccionante
α_k= Coeficiente de transferencia de calor en el lado de la mezcla reaccionante
λ = Conductividad térmica a través de la pared
d = espesor de la pared
A_r, α_r = Superficie y coeficiente de transferencia de calor del medio de calentamiento.

Parámetros y constantes del modelo

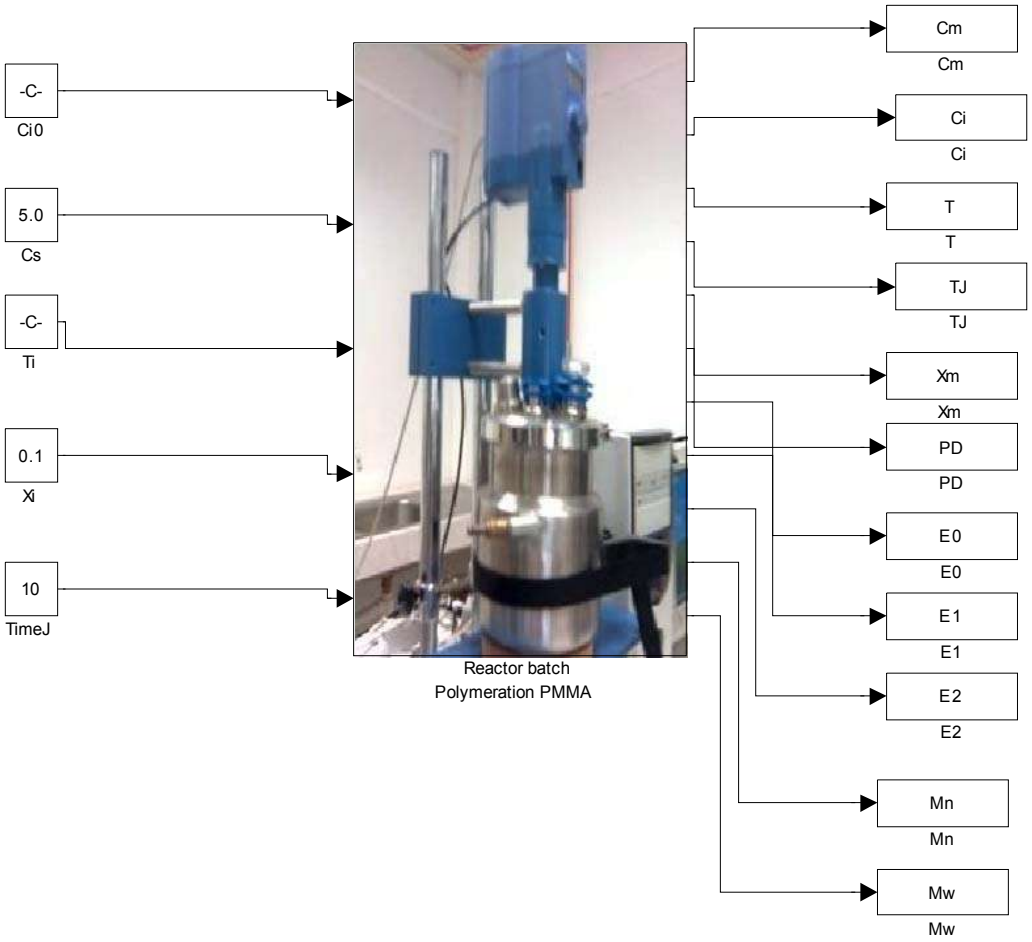
El modelo matemático planteado esta en función de parámetros cinéticos y contantes para su solución, usando como referencia los parámetros y constantes usados en el artículo de Ekpo y Mujtaba (2008), se obtiene la Tabla 3.1.

Tabla 3.1 Constantes para el modelo de polimerización.

$R=8.314$ kJ/kmolK	$Cp_m=1.648$ kJ/kgK	$V_J=1.5$ L
$f= 0.53$	$Cp_p=1.47$ kJ/kgK	$-\Delta H_r=0.000578$ kJ/kmol
$C_{s,0}=5.0$ kmol/m^3	$Cp_s=1.70$ kJ/kgK	$\rho_J=997$ kg/m3
$C_{m,0}=3.76$ kmol/m^3	$Cp_J=4.18$ kJ/kgK	$\rho_m=915.1$ kg/m3
$M_{Ws}=92.14$ kmol/m^3	$A=0.0774$ m^2	$\rho_p=1200$ kg/m3
$M_{Wm}=100.12$ kmol/m^3	$U=2.55$ kJ/sm2K	$\tau_j = 10$
$M_{Wi}=164.21$ kmol/m^3	$V=0.7$ L	

3.1.2 Solución del modelo mediante Matlab

La solución se hace mediante la aplicación de Simulink y su *toolbox* de herramientas, para cada una de las ecuaciones se considera un subsistema, para evitar la aglomeración de líneas de unión del modelo (Figura 3.2 y Figura 3.3)



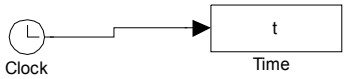


Figura 3.2 Solución modelo Matemático (Parte I)

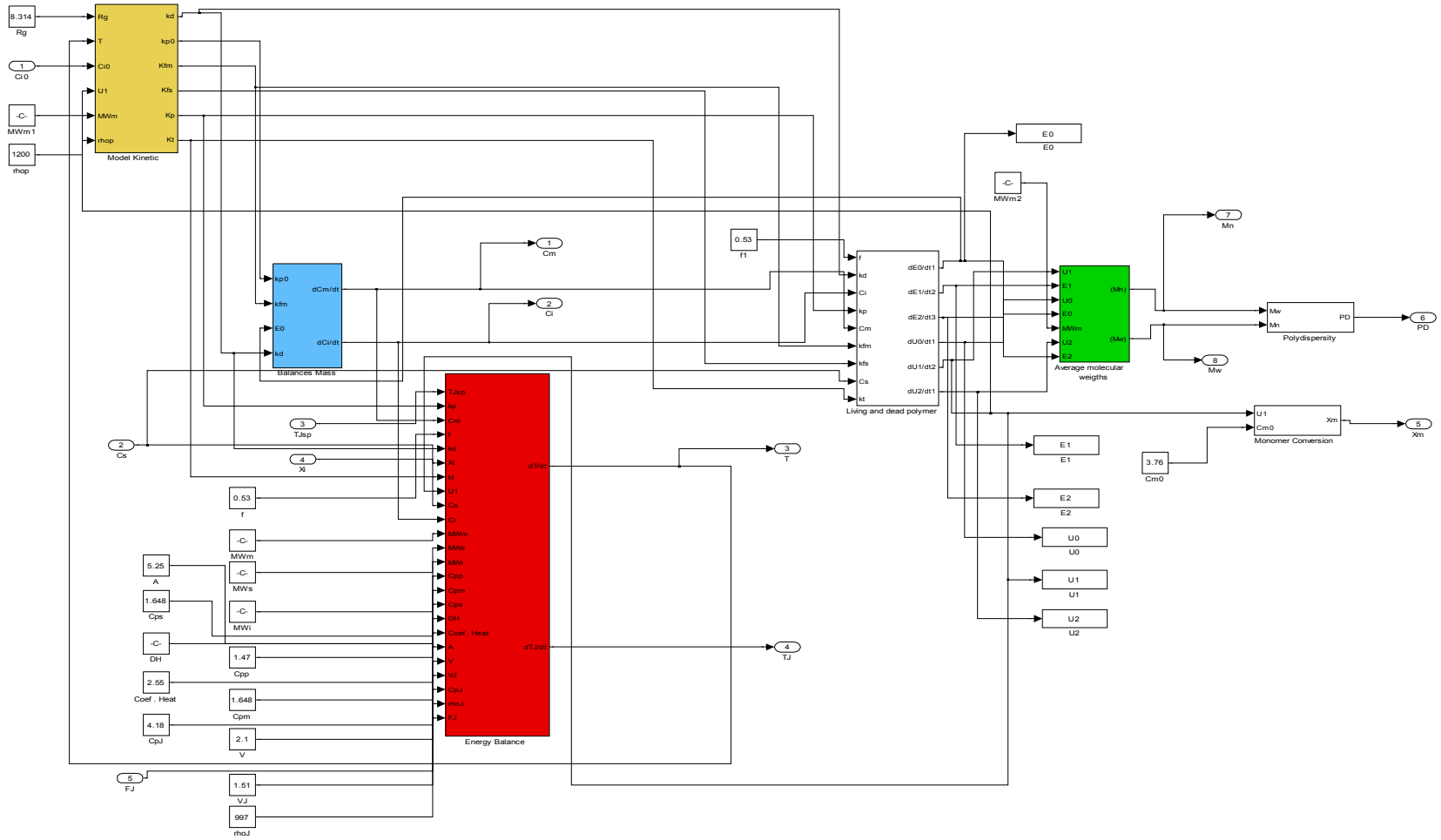


Figura 3.3 Solución del modelo Matemático (Parte II)

Se plantea la solución del modelo cinético de la reacción en Simulink dando como resultado el diagrama de bloques de la Figura 3.4

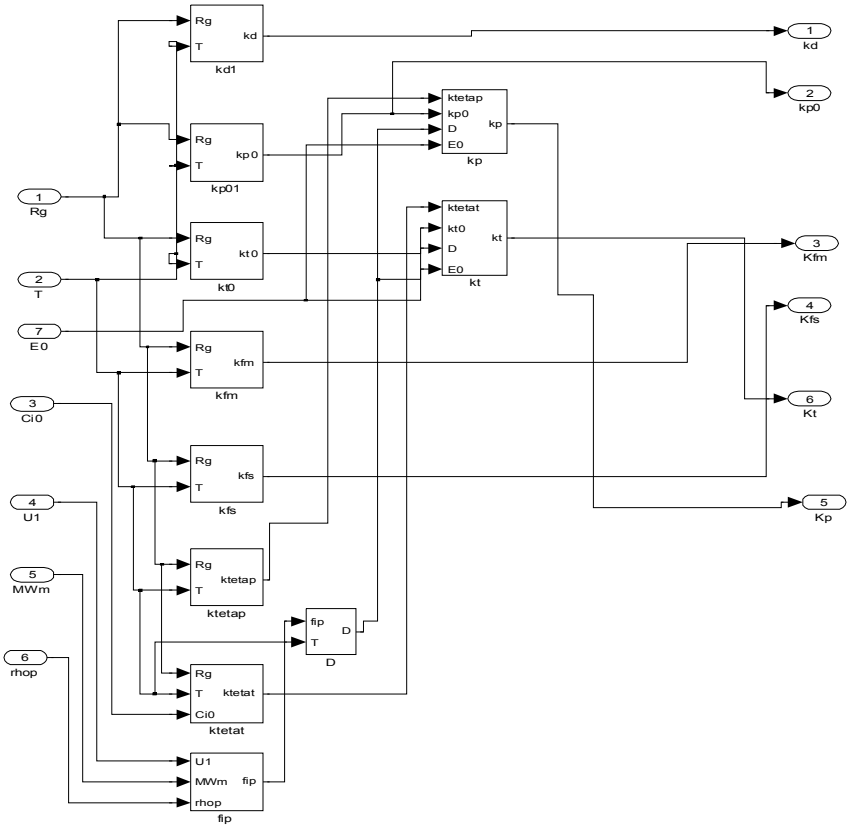


Figura 3.4 Modelo Cinético

El balance de masa se resuleve generando subconjuntos de operaciones en el simulador, estas son referenciadas a cada una de las propiedades que se desea monitoriar dentro del modelo, ver Figura 3.5.

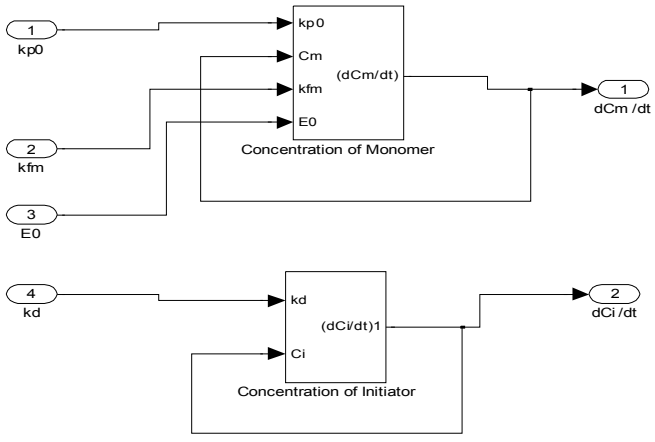


Figura 3.5 Balance de Masa.

Dentro del análisis del modelo y simulación del mismo el balance de energía tiene una cognotación de mayor importancia debido a que la variable a controlar es la temperatura, para esto se plantea la solución del mismo en subsistemas para cada una de las ecuaciones diferenciales, ver Figura 3.6.

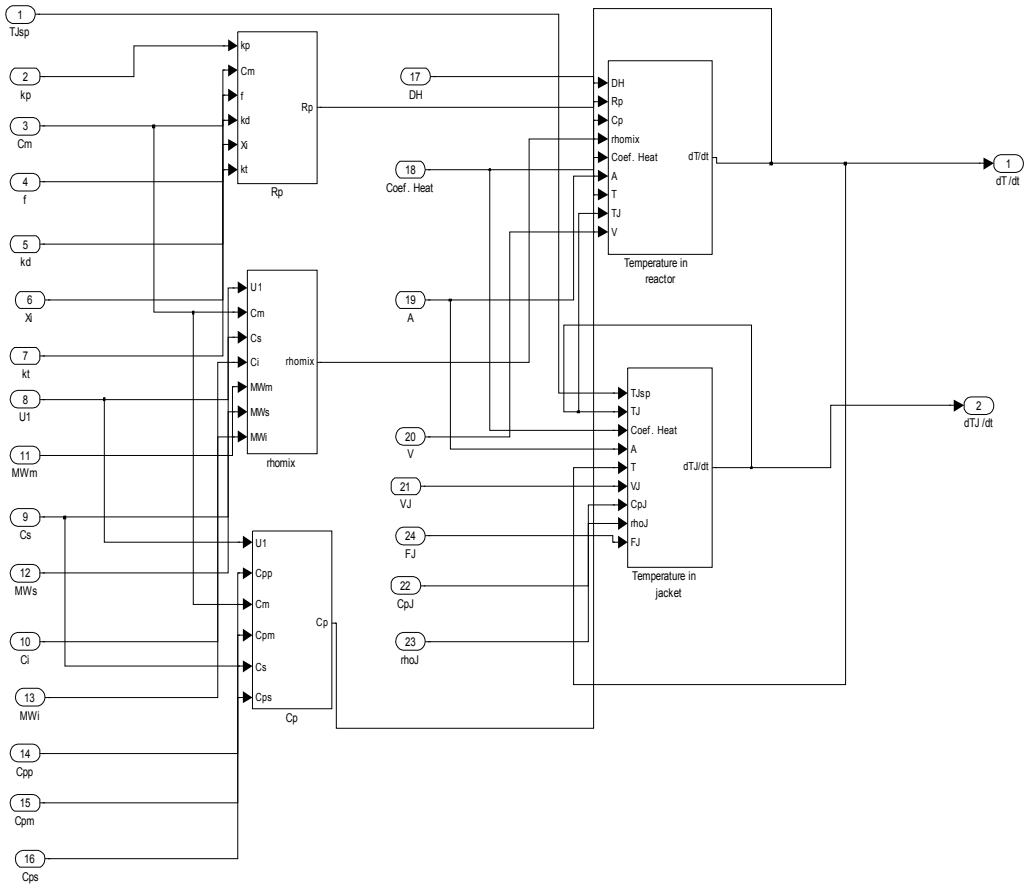


Figura 3.6 Balance de Energía

La Figura 3.7 representa la forma de solución del balance realizado a la conversión del iniciador.

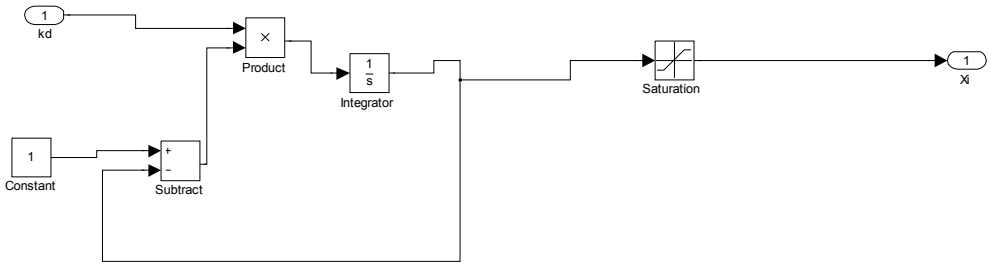


Figura 3.7 Conversión del Iniciador

Las variables de vida y muerte del polímero son variables que son parte fundamental del modelo matemático para esto se plantea la solución (Figura 3.8)

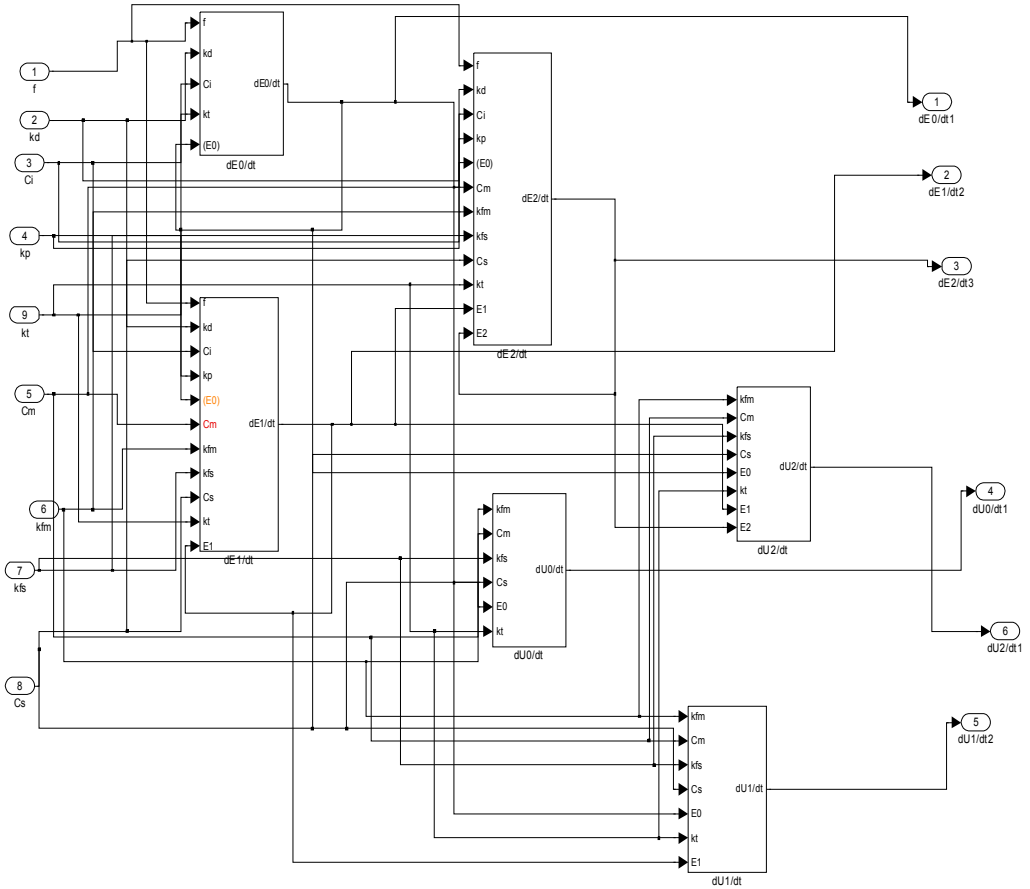


Figura 3.8 Vida y Muerte del Polímero

La masa y peso del polímero se monitorea mediante una relación del vida y muerte del polímero, ver Figura 3.9.

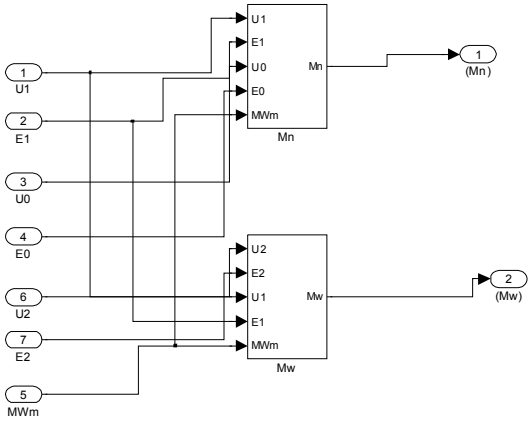


Figura 3.9 Masa y Peso molecular promedio del Polímero

3.1.3 Planteamiento del fenotipo y genotipo del control neuroevolutivo

La ecuación del progenitor A se plantea de la siguiente manera:

$$P_A = \left(\nabla \cdot \sigma_i w_i + y_t\right) e_i \tag{3.34}$$

Dónde:

y_t= serie temporal

σ_i= función de activación

w_i = peso sináptico

e_i= número de eventos

$$\nabla \cdot \sigma_i w_i = \frac{\partial(\sigma_i w_i)}{\partial x} + \frac{\partial(\sigma_i w_i)}{\partial y} + \frac{\partial(\sigma_i w_i)}{\partial z} \tag{3.35}$$

$$\nabla \cdot \sigma_i w_i = \sigma_i \frac{\partial w_i}{\partial x} + w_i \frac{\partial \sigma_i}{\partial x} + \sigma_i \frac{\partial w_i}{\partial y} + w_i \frac{\partial \sigma_i}{\partial y} + \sigma_i \frac{\partial w_i}{\partial z} + w_i \frac{\partial \sigma_i}{\partial z} \tag{3.36}$$

Se consideran las expansiones para los pesos sinápticos y funciones de excitación:

$$\nabla \cdot \sigma_i w_i = \sigma_i \left(\frac{\partial w_i}{\partial x} + \frac{\partial w_i}{\partial y} + \frac{\partial w_i}{\partial z} \right) + w_i \left(\frac{\partial \sigma_i}{\partial x} + \frac{\partial \sigma_i}{\partial y} + \frac{\partial \sigma_i}{\partial z} \right) \tag{3.37}$$

$$\nabla \cdot \sigma_i w_i = \sigma_i \left(\sum_{i=1}^{n} \frac{\partial w_i}{\partial x} + \sum_{i=1}^{n} \frac{\partial w_i}{\partial y} + \sum_{i=1}^{n} \frac{\partial w_i}{\partial z} \right) + w_i \left(\sum_{i=1}^{n} \frac{\partial \sigma_i}{\partial x} + \sum_{i=1}^{n} \frac{\partial \sigma_i}{\partial y} + \sum_{i=1}^{n} \frac{\partial \sigma_i}{\partial z} \right) \tag{3.38}$$

Expandiendo para tres dimensiones ficticias:

$$\nabla \cdot \sigma_i w_i = \left[\sum_{i=1}^{n} \sigma_i \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial w_i}{\partial x} + \frac{\partial w_j}{\partial x} + \frac{\partial w_k}{\partial x} \right) \right] + \sum_{j=1}^{n} \sigma_j \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial w_i}{\partial x} + \frac{\partial w_j}{\partial x} + \frac{\partial w_k}{\partial x} \right) \right] \right]$$
$$+ \sum_{k=1}^{n} \sigma_k \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial w_i}{\partial x} + \frac{\partial w_j}{\partial x} + \frac{\partial w_k}{\partial x} \right) \right] + \sum_{i=1}^{n} w_i \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial \sigma_i}{\partial x} + \frac{\partial \sigma_j}{\partial x} + \frac{\partial \sigma_k}{\partial x} \right) \right]$$
$$+ \sum_{j=1}^{n} w_j \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial \sigma_i}{\partial x} + \frac{\partial \sigma_j}{\partial x} + \frac{\partial \sigma_k}{\partial x} \right) \right] + \sum_{k=1}^{n} w_k \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial \sigma_i}{\partial x} + \frac{\partial \sigma_j}{\partial x} + \frac{\partial \sigma_k}{\partial x} \right) \right]$$
$$+ \left[\sum_{i=1}^{n} \sigma_i \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial w_i}{\partial y} + \frac{\partial w_j}{\partial y} + \frac{\partial w_k}{\partial y} \right) \right] + \sum_{j=1}^{n} \sigma_j \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial w_i}{\partial y} + \frac{\partial w_j}{\partial y} + \frac{\partial w_k}{\partial y} \right) \right] \right]$$
$$+ \sum_{k=1}^{n} \sigma_k \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial w_i}{\partial y} + \frac{\partial w_j}{\partial y} + \frac{\partial w_k}{\partial y} \right) \right] + \sum_{i=1}^{n} w_i \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial \sigma_i}{\partial y} + \frac{\partial \sigma_j}{\partial y} + \frac{\partial \sigma_k}{\partial y} \right) \right]$$
$$+ \sum_{j=1}^{n} w_j \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial \sigma_i}{\partial y} + \frac{\partial \sigma_j}{\partial y} + \frac{\partial \sigma_k}{\partial y} \right) \right] + \sum_{k=1}^{n} w_k \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial \sigma_i}{\partial y} + \frac{\partial \sigma_j}{\partial y} + \frac{\partial \sigma_k}{\partial y} \right) \right]$$
$$+ \left[\sum_{i=1}^{n} \sigma_i \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial w_i}{\partial z} + \frac{\partial w_j}{\partial z} + \frac{\partial w_k}{\partial z} \right) \right] + \sum_{j=1}^{n} \sigma_j \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial w_i}{\partial z} + \frac{\partial w_j}{\partial z} + \frac{\partial w_k}{\partial z} \right) \right] \right]$$
$$+ \sum_{k=1}^{n} \sigma_k \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial w_i}{\partial z} + \frac{\partial w_j}{\partial z} + \frac{\partial w_k}{\partial z} \right) \right] + \sum_{i=1}^{n} w_i \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial \sigma_i}{\partial z} + \frac{\partial \sigma_j}{\partial z} + \frac{\partial \sigma_k}{\partial z} \right) \right]$$
$$+ \sum_{j=1}^{n} w_j \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial \sigma_i}{\partial z} + \frac{\partial \sigma_j}{\partial z} + \frac{\partial \sigma_k}{\partial z} \right) \right] + \sum_{k=1}^{n} w_k \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{\partial \sigma_i}{\partial z} + \frac{\partial \sigma_j}{\partial z} + \frac{\partial \sigma_k}{\partial z} \right) \right]$$

Considerando una sola dirección de la propagación de la información:

$$P_A = \left[\begin{array}{l} \left[\sum_{i=1}^{n} \sigma_i \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{dw_i}{dx} + \frac{dw_j}{dx} + \frac{dw_k}{dx} \right) \right] + \sum_{j=1}^{n} \sigma_j \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{dw_i}{dx} + \frac{dw_j}{dx} + \frac{dw_k}{dx} \right) \right] \right] \\ \quad + \sum_{k=1}^{n} \sigma_k \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{dw_i}{dx} + \frac{dw_j}{dx} + \frac{dw_k}{dx} \right) \right] + \sum_{i=1}^{n} w_i \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{d\sigma_i}{dx} + \frac{d\sigma_j}{dx} + \frac{d\sigma_k}{dx} \right) \right] \\ \quad + \sum_{j=1}^{n} w_j \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{d\sigma_i}{dx} + \frac{d\sigma_j}{dx} + \frac{d\sigma_k}{dx} \right) \right] + \sum_{k=1}^{n} w_k \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{d\sigma_i}{dx} + \frac{d\sigma_j}{dx} + \frac{d\sigma_k}{dx} \right) \right] + y_t \end{array} \right] e_i$$

Serie temporal y_t en una sola dimensión:

$$y_t = \nabla \beta_i \Delta \sigma_i \tag{3.40}$$

Dónde:

$$\Delta \sigma_i = \sigma_f - \sigma_I \tag{3.41}$$

σ_f = activación final

σ_I = activación inicial

Para una sola dirección en la serie temporal:

$$y_t = \left\{ \sum_{i,j,k}^{n} \beta_i \beta_j \beta_k \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left[\left(\frac{d\sigma_f}{dx} - \frac{d\sigma_I}{dx} \right)_i + \left(\frac{d\sigma_f}{dx} - \frac{d\sigma_I}{dx} \right)_j + \left(\frac{d\sigma_f}{dx} - \frac{d\sigma_I}{dx} \right)_k \right] \right] \right\} +$$

$$\left\{ \sum_{i,j,k}^{n} \sigma_{f_i} \sigma_{f_j} \sigma_{f_k} \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{d\beta_i}{dx} + \frac{d\beta_j}{dx} + \frac{d\beta_k}{dx} \right)_i \right] \right\} - \left\{ \sum_{i,j,k}^{n} \sigma_{I_i} \sigma_{I_j} \sigma_{I_k} \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{d\beta_i}{dx} + \frac{d\beta_j}{dx} + \frac{d\beta_k}{dx} \right)_i \right] \right\}$$

Función para cada uno de los eventos:

$$e_i = f_i \left(\nabla w_i \sigma_i - \theta_i \right) \tag{3.43}$$

Dónde:

e_i= número de eventos

f_i = error cuadrático

θ_i = error

$$e_i = \sum_{i,j,k}^{n} f_i f_j f_k \left\{ \begin{array}{l} \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} w_i w_j w_k \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{d\sigma_i}{dx} + \frac{d\sigma_j}{dx} + \frac{d\sigma_k}{dx} \right) \right] + \\ \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \sigma_i \sigma_j \sigma_k \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \left(\frac{dw_i}{dx} + \frac{dw_j}{dx} + \frac{dw_k}{dx} \right) \right] - \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \theta_i \theta_j \theta_k \right] \end{array} \right\}$$

Se obtiene el modelo de un progenitor A para el control neuroevolutivo de la siguiente manera:

$$P_A = \left[\begin{array}{l} \left[\sum_{i=1}^{n}\sigma_i\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right)\right]+\sum_{j=1}^{n}\sigma_j\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right)\right]\right] \\ +\sum_{k=1}^{n}\sigma_k\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right)\right]+\sum_{i=1}^{n}w_i\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)\right]+ \\ +\sum_{j=1}^{n}w_j\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)\right]+\sum_{k=1}^{n}w_k\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)\right] \end{array}\right] +$$

$$\left\{\sum_{i,j,k}^{n}\beta_i\beta_j\beta_k\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left[\left(\frac{d\sigma_f}{dx}-\frac{d\sigma_I}{dx}\right)_i+\left(\frac{d\sigma_f}{dx}-\frac{d\sigma_I}{dx}\right)_j+\left(\frac{d\sigma_f}{dx}-\frac{d\sigma_I}{dx}\right)_k\right]\right]\right\}+$$

$$\left\{\sum_{i,j,k}^{n}\sigma_{f_i}\sigma_{f_j}\sigma_{f_k}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\beta_i}{dx}+\frac{d\beta_j}{dx}+\frac{d\beta_k}{dx}\right)_i\right]\right\}-$$

$$\left\{\sum_{i,j,k}^{n}\sigma_{I_i}\sigma_{I_j}\sigma_{I_k}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\beta_i}{dx}+\frac{d\beta_j}{dx}+\frac{d\beta_k}{dx}\right)_i\right]\right\}\times$$

$$\sum_{i,j,k}^{n}f_i f_j f_k\left\{\begin{array}{l}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}w_i w_j w_k\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)\right]+ \\ \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sigma_i\sigma_j\sigma_k\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right)\right]-\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\theta_i\theta_j\theta_k\right]\end{array}\right\}$$

Para un sistema neuronal evolutivo es necesario el partir por dos progenitores (Ec. 3. 33) de forma similar se plantea la ecuación para el progenitor B

$$P_B = \left[\begin{array}{l} \left[\sum_{i=1}^{n}\sigma_i\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right)\right]+\sum_{j=1}^{n}\sigma_j\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right)\right]\right] \\ +\sum_{k=1}^{n}\sigma_k\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right)\right]+\sum_{i=1}^{n}w_i\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)\right]+ \\ +\sum_{j=1}^{n}w_j\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)\right]+\sum_{k=1}^{n}w_k\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)\right] \end{array}\right]+$$

$$\left\{\sum_{i,j,k}^{n}\beta_i\beta_j\beta_k\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left[\left(\frac{d\sigma_f}{dx}-\frac{d\sigma_I}{dx}\right)_i+\left(\frac{d\sigma_f}{dx}-\frac{d\sigma_I}{dx}\right)_j+\left(\frac{d\sigma_f}{dx}-\frac{d\sigma_I}{dx}\right)_k\right]\right]\right\}+$$

$$\left\{\sum_{i,j,k}^{n}\sigma_{f_i}\sigma_{f_j}\sigma_{f_k}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\beta_i}{dx}+\frac{d\beta_j}{dx}+\frac{d\beta_k}{dx}\right)_i\right]\right\}-$$

$$\left\{\sum_{i,j,k}^{n}\sigma_{I_i}\sigma_{I_j}\sigma_{I_k}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\beta_i}{dx}+\frac{d\beta_j}{dx}+\frac{d\beta_k}{dx}\right)_i\right]\right\}\times$$

$$\sum_{i,j,k}^{n}f_i f_j f_k\left\{\begin{array}{l}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}w_i w_j w_k\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)\right]+ \\ \left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sigma_i\sigma_j\sigma_k\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right)\right]-\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\theta_i\theta_j\theta_k\right]\end{array}\right\}$$

Dónde:

$$\sigma_i = \frac{1}{1-\exp(-w)} \tag{3.47}$$

La Ecuación 3.47 representa la función sigmoidal de la propagación de información.

$$P_{A_{n+1}} = \sum_{n=1}^{n+1}\left(\sum_{i,j,k}^{n+1}\sigma_i\sigma_j\sigma_k\left(\sum_{i,j,k}^{n+1}\left\{\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right\}\right)\right)+\left\{\sum_{i,j,k}^{n+1}\left[\frac{\left(\sum_{j=1}^{i-1}w_{i.j}x_{i,j}\right)\left(\sum_{i=1}^{j-1}w_{i.j}x_{i,j}\right)\left(\sum_{i=1}^{k-1}w_{i.j}x_{i,j}\right)}{\sum_{i,j,k}^{n+1}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)}\right]\right\}+$$

$$\left\{\sum_{i,j,k}^{n}\beta_i\beta_j\beta_k\left[\sum_{i=1}^{n+1}\sum_{j=1}^{n+1}\sum_{k=1}^{n+1}\left[\left(\frac{d\sigma_f}{dx}-\frac{d\sigma_l}{dx}\right)_i+\left(\frac{d\sigma_f}{dx}-\frac{d\sigma_l}{dx}\right)_j+\left(\frac{d\sigma_f}{dx}-\frac{d\sigma_l}{dx}\right)_k\right]\right]\right\}+$$

$$\left\{\sum_{i,j,k}^{n+1}\sigma_{f_i}\sigma_{f_j}\sigma_{f_k}\left[\sum_{i=1}^{n+1}\sum_{j=1}^{n+1}\sum_{k=1}^{n+1}\left(\frac{d\beta_i}{dx}+\frac{d\beta_j}{dx}+\frac{d\beta_k}{dx}\right)_i\right]\right\}-\left\{\sum_{i,j,k}^{n+1}\sigma_{l_i}\sigma_{l_j}\sigma_{l_k}\left[\sum_{i=1}^{n+1}\sum_{j=1}^{n+1}\sum_{k=1}^{n+1}\left(\frac{d\beta_i}{dx}+\frac{d\beta_j}{dx}+\frac{d\beta_k}{dx}\right)_i\right]\right\}$$

$$\left\{\sum_{i,j,k}^{n+1}\left(\frac{1}{1-\exp(-w_i)}\right)\left(\frac{1}{1-\exp(-w_j)}\right)\left(\frac{1}{1-\exp(-w_k)}\right)\left\{\sum_{i,j,k}^{n+1}\left[\frac{\left(\sum_{j=1}^{i-1}w_{i.j}x_{i,j}\right)\left(\sum_{i=1}^{j-1}w_{i.j}x_{i,j}\right)\left(\sum_{i=1}^{k-1}w_{i.j}x_{i,j}\right)}{\sum_{i,j,k}^{n+1}\left(\frac{d\sigma_i}{dx}+\frac{d\sigma_j}{dx}+\frac{d\sigma_k}{dx}\right)}\right]\right\}\right\}+$$

$$\sum_{n=1}^{n+1}\left(\sum_{i,j,k}^{n+1}\sigma_i\sigma_j\sigma_k\left(\sum_{i,j,k}^{n+1}\left\{\frac{dw_i}{dx}+\frac{dw_j}{dx}+\frac{dw_k}{dx}\right\}\right)\right)-\sum_{i,j,k}^{n+1}\theta_i\theta_j\theta_k$$

Donde para analizar los valores de los pesos sinápticos:

$$\left(w_{i,j}\right)_{test} = \frac{\sum_{t=1}^{T}\xi_{i,j}^{t}}{\sqrt{\sum_{t=1}^{T}\left(\xi_{i,j}^{t}-\overline{\xi_{i,j}}\right)^2}} \tag{3.49}$$

Dónde:

$$\xi_{i,j}^{t} = w_{i,j} + \Delta w_{i,j}^{t}(w) \tag{3.50}$$

para este caso el término $\xi_{i,j}^{t}$ denota el promedio sobre el punto t=1,…,A

3.1.4 Estructura retroalimentada de una red neuronal

Se plantean diferentes fenotipos y genotipos que proporcionen respuestas de acuerdo a la dinámica del proceso. Se puede plantear el siguiente fenotipo (Figura 3.10):

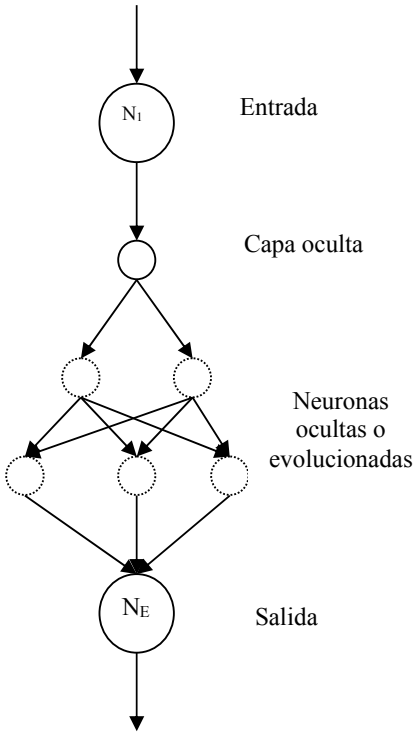


Figura 3.10 Fenotipo propuesto para el control neuroevolutivo

Así mismo plantea una estrategia de evolución. Esta técnica basada en el principio de evolución de Darwin, fue originalmente desarrollada por Rechenberg en 1973, y llevada a su forma actual por Schwefel en 1975.

En la estrategia evolutiva, cada individuo está formado por un vector que contiene tantas variables objeto, $w_{i,j,k}$, como parámetros tiene el vector I, y sendas variables de estrategia, σ_i. De esta forma un individuo tiene la siguiente estructura:

$$I = \lfloor w_{i+1}, w_{i+2}, ..., w_{i+n}; w_{j+1}, w_{j+2}, ..., w_{j+n}, w_{k+1}, w_{k+2}, ..., w_{k+n} \rfloor \tag{3.51}$$

La evolución es realizada sobre una población conformada por N individuos. El algoritmo se inicia generando una población de n individuos donde las variables w_i son generadas de forma aleatoria o a partir de un modelo matemático del proceso a controlar; las variables w_j se inician en tres unidades siguiendo las secuencias dadas por las neuronas antecedentes; y las variables w_k son generadas a partir de la diferencia de la secuencia de entrada con la secuencia hacia delante.

El proceso de reproducción se realiza escogiendo dos padres de forma aleatoria entre la población, para luego proceder a realizar un proceso de combinación entre ellos; existen diversas formas para realizar este proceso, pero se escogió la reproducción sexual panmínica intermedia para maximizar la capacidad de búsqueda del algoritmo. En este tipo de reproducción sexual, cada una de las componentes del nuevo individuo, c_i, se calculan como:

$$P_{A_{n+i}} = P_{A_{S,i}} + \eta_i \nabla \times \left(P_{A_{T,i}} - P_{A_{S,i}} \right) \tag{3.52}$$

Donde S y T representan cada uno de los padres seleccionados y η_i representa un número aleatorio uniforme entre cero y la unidad.

Las estrategias evolutivas poseen la propiedad de auto-adaptación, lo cual implica que no sólo evolucionan las variables del problema, sino también los parámetros mismos del algoritmo.

Dentro del estudio de las redes neuronales evolutivas se puede establecer un método de obtención del número de neuronas evolucionadas mediante la regla heurística de estados de evolución, la regla de estados de evolución relaciona el número de épocas y el número de salidas de la red neuronal de acuerdo a lo siguiente N^n; donde N representa el número de salidas y n el número de épocas de evolución.

3.1.5 Control Neuronal

En la Figura 3.11 se presenta el planteamiento del control neuronal simple que es la ante sala para el control neuroevolutivo.

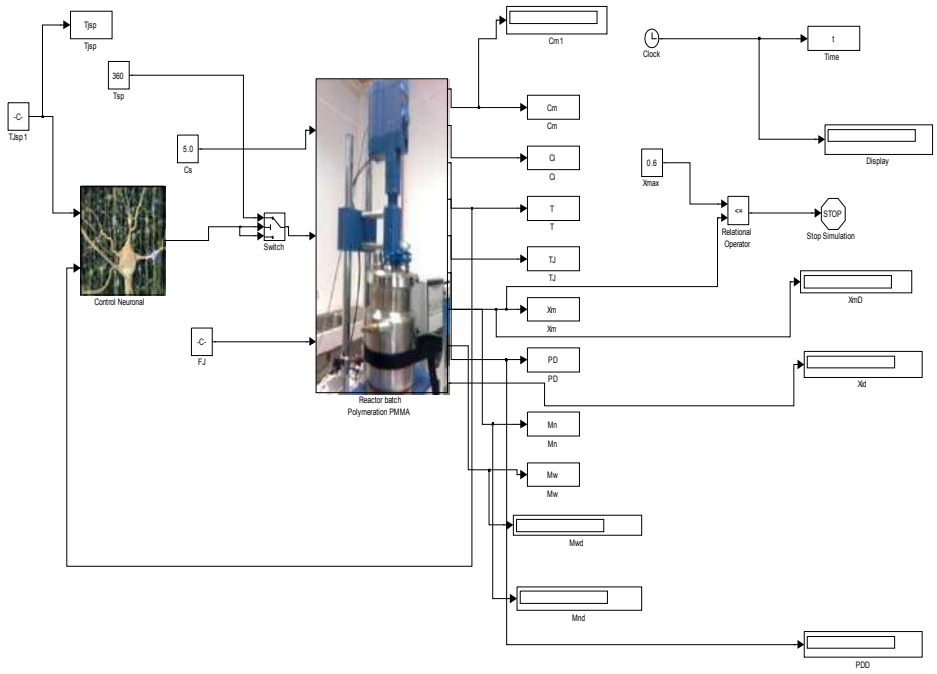


Figura 3.11 Control Neuronal simple

La estructura interna del control neuronal simple se basa en una estructura NARMA-L2 como un preceptor multicapa de la siguiente manera (Figura 3.12).

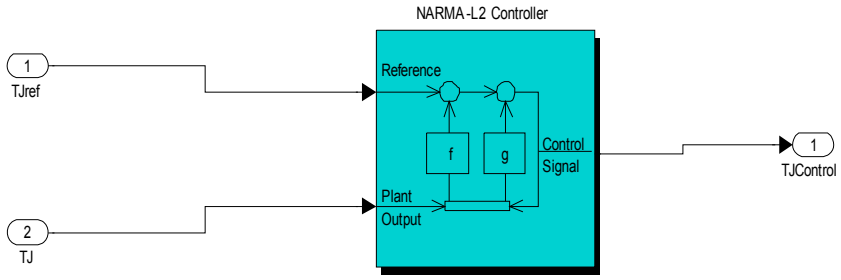


Figura 3.12 Estructura interna del control neuronal

La estructura interna de dicha red neuronal la tiene definida MATLAB de la siguiente forma ver Figura 3.13.

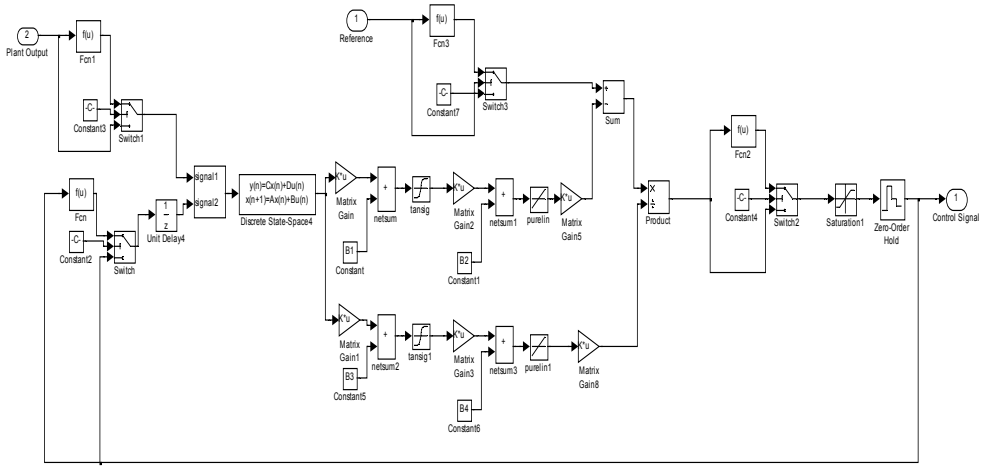


Figura 3.13 Estructura del control NARMA-L2

Entrenamiento de Hebb.

Se trata de uno de los modelos clásicos de entrenamiento de redes neuronales. Donald Hebb en 1949 postuló un mecanismo de aprendizaje para la neurona biológica, cuya idea básica consiste en que cuando un axón presináptico causa la activación de cierta neurona post sináptica, la eficacia de la sinapsis que las relaciona se refuerza.

De una manera general, se denomina entrenamiento Hebbiano a aquellas formas de entrenamiento que involucran una modificación en los pesos proporcional al producto de una entrada j por la salida i de la neurona.

$$\Delta w_{ij} = \varepsilon y_i x_i \tag{3.53}$$

Donde ε es el ritmo de aprendizaje, que suele ser una cantidad entre 0 y 1. En donde la representación matemática es la siguiente considerando una función de transferencia lineal.

$$\Delta w_{ij}{}^{\mu} = t_i^{\mu} x_j^{\mu} \tag{3.54}$$

$$w_{ij}^{nuevo} = w_{ij}^{viejo} + \Delta w_{ij}^{\mu} \tag{3.55}$$

Si los pesos de partida son nulos, el valor final de W para las p asociaciones será:

$$W\mathrm{x}^{\mu} = \left(t^1 x^{1T} + \ldots + t^p x^{pT}\right) x^{\mu} \tag{3.56}$$

Eliminando la condición de ortogonalidad

$$W\mathrm{x}^{\mu} = t^{\mu} + \sum_{\nu \neq \mu} t^{\nu} \left(x^{\nu T} x^{\mu} \right) = t^{\mu} + \delta \tag{3.57}$$

La expresión anterior es denominada expansión señal ruido, pues proporciona la salida deseada más un término adicional, que interpretamos como el ruido superpuesto a la señal.

El entrenamiento de un preceptor (modelo de red neuronal retropropagación), es un algoritmo de entrenamiento de los denominados por corrección de errores. Los algoritmos de este tipo ajustan los pesos en proporción a la diferencia existente entre la salida actual de la red y la salida deseada con el objetivo de minimizar el error actual de la red.

Sea un conjunto de p patrones $\mathbf{x}^{\mu}$, μ=1,…,p, con sus salidas deseadas $\mathbf{t}^{\mu}$. Tanto las entradas como las salidas solamente pueden tomar los valores de -1 o 1. Si se tiene una arquitectura de preceptron simple, con pesos iniciales aleatorios, y se requiere que se clasifiquen correctamente todos los patrones del conjunto de entrenamientos (lo cual es solamente posible si son separables linealmente). Se actúa de un modo ante la presencia del patrón μ-ésimo, si la respuesta que proporciona el preceptor es correcta, no actualiza los pesos; si es incorrecta se modifican según la regla de Hebb.

$$\Delta w_{ij}^{\mu}(t) = \begin{cases} 2\varepsilon . t_i^{\mu} x_j^{\mu}, & si \quad y_i^{\mu} \neq t_i^{\mu} \\ 0 \qquad , & si \quad y_i^{\mu} = t_i^{\mu} \end{cases} \tag{3.58}$$

$$\Delta w_{ij}^{\mu}(t) = \varepsilon . \left(t_i^{\mu} - y_i^{\mu} \right) x_j^{\mu} \tag{3.59}$$

Habrá que decir que la forma anterior es para el cálculo de los gradientes de los pesos en punto actual de la salida o entrada de la red neuronal y como ya se había mencionado con anterioridad este entrenamiento se basa en la estimación de las salida por lo tanto es necesario el conocimiento de la salida, esto se logra introduciendo la siguiente ecuación para el entrenamiento para un conjunto de iteraciones t.

$$w_{ij}(t+1) = w_{ij}(t) + \sum_{\mu=1}^{p} \Delta w_{ij}^{\mu}(t) \tag{3.60}$$

Rosemblatt demostró que si la función a representar es linealmente separable, este algoritmo siempre converge en un tiempo finito y con independencia de los pesos de partida. Por otra parte si la función es no linealmente separable, el proceso de entrenamiento oscilara.

Si se incrementan capas intermedias (ocultas) a un perceptron multicapa o MLP. Esta arquitectura suele entrenarse mediante el algoritmo denominado retropropagación de errores o bien haciendo uso de alguna de sus variantes o derivados, motivo por lo que en muchas ocasiones el conjunto arquitectura MLP + aprendizaje Backprpagation suele denominarse red de retropropagación.

Una solución al problema de entrenar los nodos de las capas ocultas pertenecientes a arquitecturas multicapa la proporciona el algoritmo de retropropagación de errores (Rumelhart, 1986ª, 1986b, Hecht-Nielsen, 1991).

Dado un patrón de entradas $\mathbf{x}^{\mu}$, (μ=1,…, p), en donde la operación global de esta arquitectura se expresa del siguiente modo.

$$z_k^{\mu} = \sum w_{kj}^{'} y_j^{\mu} - \theta_k^{'} = \sum_j w_{kj}^{'} f\left(\sum_i w_{ji} x_i^{\mu} - \theta_j \right) - \theta_k^{'} \tag{3.61}$$

Las funciones de transferencia de las neuronas son del tipo sigmoideo, con h el potencial post-sináptico o local. Como en el caso de la Adalina, la función de costo de la que es parte el error cuadrático medio.

$$E\left(w_{ji}, \theta_j, w_{kj}^{'}, \theta_k^{'} \right) = \left(\frac{1}{2} \right) \sum_{\mu} \sum_{k} \left[t_k^{\mu} - f\left(\sum_j w_{kj}^{'} y_j^{\mu} - \theta_k^{'} \right) \right]^2 \tag{3.62}$$

La minimización se lleva a cabo mediante descenso por el gradiente, pero en esta ocasión habrá un gradiente respecto de los pesos de la capa de salida y otro con respecto de los de la oculta.

$$\delta w_{kj}^{'} = -\varepsilon \frac{\partial E}{\partial w_{kj}^{'}} \tag{3.63}$$

$$\delta w_{ji} = -\varepsilon \frac{\partial E}{\partial w_{ji}} \tag{3.64}$$

Las expresiones de actualización de los pesos se obtienen sólo con derivar, teniendo en cuenta las dependencias funcionales y aplicando adecuadamente la regla de la cadena.

$$\delta w_{kj}^{'} = \varepsilon \sum_{\mu} \Delta_k^{'\mu} y_j^{\mu} \tag{3.65}$$

$$\delta w_{ji} = \varepsilon \sum_{\mu} \Delta_j^{\mu} x_i^{\mu} \tag{3.66}$$

$$\Delta_k^{'\mu} = \left[t_k^{\mu} - f\left(v_k^{'\mu} \right) \right] \frac{\partial f\left(v_k^{'\mu} \right)}{\partial v_k^{'\mu}} \tag{3.67}$$

$$\Delta_j^{\mu} = \left(\sum_k \Delta_k^{'\mu} w_{kj}^{'} \right) \frac{\partial f\left(\nu_j^{\mu}\right)}{\partial \nu_j^{\mu}} \tag{3.68}$$

La actualización de los umbrales se realiza haciendo uso las ecuaciones anteriores, considerando que el umbral es un caso particular de peso sináptico, cuya entrada es una constante igual a -1.

En las ecuaciones anteriores está implícito el concepto de propagación hacia atrás de los errores que da nombre al algoritmo. En primer lugar se calcula la expresión Δ'^{μ}_k que se denomina señal de error, por ser proporcional al error de la salida actual de la red, con el calculamos la actualización $\delta w'_{kj}$ de los pesos de la capa de salida. A continuación se propaga hacia atrás los errores Δ'^{μ}_k a través de las sinapsis, proporcionando así las señales de error Δ^{μ}_j, correspondientes a las sinapsis de la capa oculta; con éstas se calcula la actualización δw_{kj} de las sinapsis ocultas. El algoritmo puede extenderse fácilmente a arquitecturas con más de una capa oculta siguiendo el mismo esquema.

En resumen el procedimiento para entrenar mediante retropropagación una arquitectura multicapa se muestra en la Figura 3.14:

1) Establecer aleatoriamente los pesos y umbrales iniciales (t:=0)
2) Para cada patrón μ del conjunto de aprendizaje.
   2.1) Llevar a cabo una fase de ejecución para obtener la respuesta de la red ante el patrón μ-ésimo.
   2.2) Calcular las señales de error asociadas Δ'^{μ}_k y Δ^{μ}_j según (3.58-3.66)
   2.3) Calcular el incremento parcial de los pesos y umbrales debidos a cada patrón (μ).
3) Calcular el incremento total (para todos los patrones) actual de los pesos $\delta w'_{kj}$ y δw_{kj} según (3.56-3.59). Hacer lo mismo para los umbrales.
4) Actualizar pesos y umbrales.
5) Calcular el error actual (b) t:=t+1, y volver a 2) si todavía no es satisfactorio.

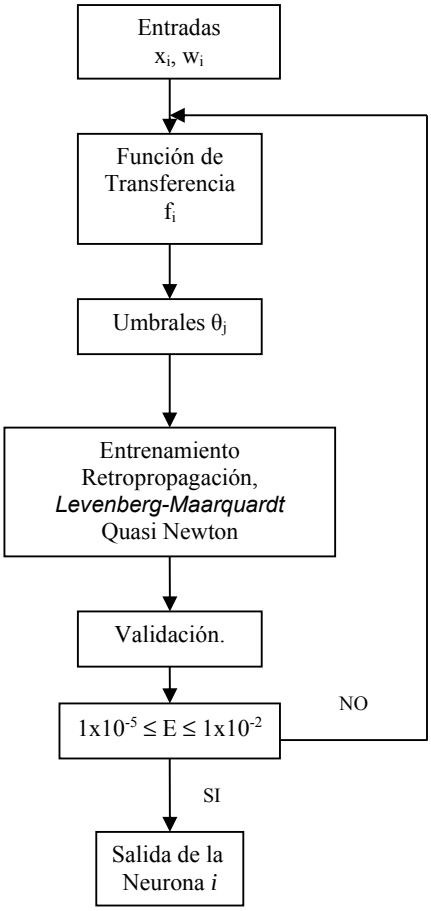


Figura 3.14 Diagrama de flujo de una neurona simple

Para resolver algunos de los inconvenientes de la retropropagación se plantea continuamente correcciones o variantes. Buena parte de estas modificaciones tratan de resolver el problema de su lenta convergencia, mientras que otras se centran en conseguir una mejor generalización. Una aceleración del entrenamiento lo da la siguiente expresión:

$$\delta w_{kj}^{'}(t+1) = -\varepsilon \left.\frac{\partial E}{\partial w_{kj}^{'}}\right|_t + \alpha.\delta w_{kj}^{'}(t-1) \tag{3.69}$$

$$\delta w_{ji}(t+1) = -\varepsilon \left.\frac{\partial E}{\partial w_{ji}}\right|_t + \alpha.\delta w_{ji}(t-1) \tag{3.70}$$

Existen otras técnicas que permiten acelerar el entrenamiento, consistentes en procesar las entradas, asignar un ritmo de aprendizaje diferente a cada peso, ritmos adaptivos, etc. (Sanz, 2005).

El entrenamiento de la retropropagación estándar está basado en la técnica de descenso por el gradiente, es decir, calculando las derivadas de la superficie de error respecto de cada peso, se obtiene la dirección de máxima pendiente de la superficie la cual se sigue para actualizar los pesos; sin embargo, nadie garantiza que sea ese precisamente el camino más rápido hacia el mínimo. Los denominados métodos de segundo orden se basan en realizar el descenso utilizando también la información por el ritmo de cambio de la pendiente.

$$H = \frac{\partial^2 E(W)}{\partial w_{ij} \partial_{kj}} \tag{3.71}$$

Los algoritmos de gradientes conjugados, gradientes conjugados escalados y *Levenberg-Maarquardt* son ejemplos de ellos.

Este último está en función de la matriz Hessiana, el algoritmo de este tipo de entrenamiento es el siguiente:

$$x_{k+1} = x_k - \left[J^T J + \mu I \right]^{-1} J^T e \tag{3.72}$$

Otra técnica que tiene como punto de partida el entrenamiento de tipo retro-propagación, es el entrenamiento Quasi-Newton *(BFGS, Broyden, Fletcher, Golfart, Shanno por sus siglas en inglés*) que consiste en una técnica de minimización del error cuadrático, este método es más robusto con respecto al entrenamiento de la red pero tiene una desventaja con respecto al otros métodos de optimización del entrenamiento y reside principalmente en obtener valores locales que pueden no describir de forma adecuada la dinámica del proceso, muchos autores plantean entrenamientos combinados para evitar estos problemas pero su programación no suele ser fácil para alguien que no tiene experiencia en los fenómenos ocurridos con las redes neuronales, las ecuaciones que describen el proceso de mejora del entrenamiento son las siguientes:

$$H_{k+1} = H_k + \frac{q_k q_k^T}{q_k^k s_k} - \frac{H_k^T s_k^T s_k H_k}{s_k^T H_k s_k} \tag{3.73}$$

Donde *H* representa a la matriz Hessiana para la neurona en la dirección *k*, *s* representa el vector cuadrático tomado desde el punto de referencia de un vector de parámetros y *q* es la suma de los errores residuales al cuadrado propagado en las diferentes direcciones de la red neuronal.

$$s_k = \Theta^{(r+1)} - \Theta^r \tag{3.74}$$

$$q_k = \nabla \cdot \sum_{t,i,j}^{n} \left(\Theta^{(r+1)}\right) - \nabla \cdot \sum_{i,j}^{n} \left(\Theta^{r}\right) \tag{3.75}$$

Las Ecuaciones (3.74-3.75) tienen como parámetro fundamental el vector de pesos o simplemente vector de parámetros *Θ*, el vector de parámetros tiene importancia dentro del entrenamiento por el método BFGS debido principalmente a que el parámetro debe coincidir con un punto de partida en el óptimo global con esto en el entrenamiento se puede asegurar que la red tiene una rápida convergencia evitando valores locales.

Estas dos técnicas planteadas con anterioridad son procedimientos más robustos que la retropropagación, que pueden acelerar en uno o dos órdenes de magnitud la convergencia, aunque como contrapartida son mucho más complejas de implementar y precisan más recursos de cálculo.

Se debe dejar en claro que desgraciadamente ninguno de los algoritmos o recetas descritos puede considerarse superior en general: un buen rendimiento en una puede proporcionar un rendimiento pobre para otra.

No obstante Matlab proporciona una gama importante de la implementación de dichos entrenamientos en donde el fundamento básico es el entrenamiento de Hebb, como el mostrado en la Figura 3.15, en donde se presenta un entrenamiento en base al modelo matemático del proceso analizado.

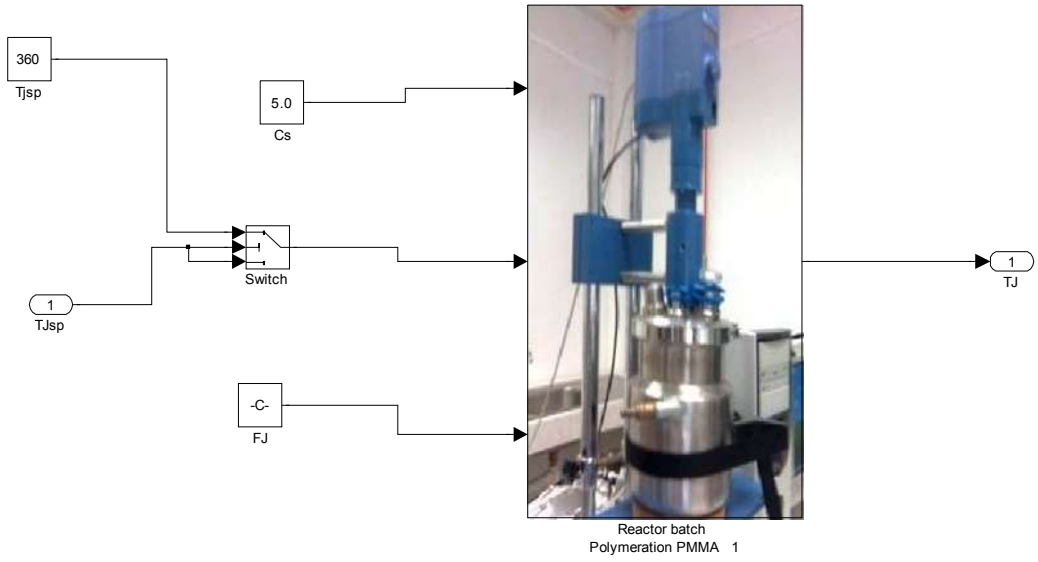


Figura 3.15 Modelo de Entrenamiento de la red Neuronal

3.1.6 Control PID

Una de las razones de la búsqueda de nuevos métodos de control de procesos es el sustituir el control de tipo tradicional como el PID (Proporcional Integral Derivativo), por tipos de control que no requiera sintonización mediante reglas heurísticas como es el caso del control mencionado. Para fines de comparación se realiza el control PID para el sistema de estudio, del cual se obtiene su ley control así como las constantes de cada una de las partes del controlador.

$$\frac{dx}{dt} = K\left(\left(x_{sp} - x\right) + \frac{1}{T_i}\int_0^t \left(x_{sp} - x\right)dt + T_d \frac{d\left(x_{sp} - x\right)}{dt} \right) \tag{3.76}$$

Donde K es la constante proporcional, T_i es el tiempo integral y T_d es el tiempo derivativo, además x es la variable a controlar y x_{sp} es la variable en el set point. Partiendo de esto la ley de control para el PID queda de la siguiente manera

$$T_J = T + \frac{K\rho_r Cp_r V_r}{UA}\left(\left(T_{rsp} - T\right) + \frac{1}{T_i}\int_0^t \left(T_{rsp} - T\right)dt + T_d \frac{d\left(T_{rsp} - T\right)}{dt} \right) - \frac{Q_r}{UA} \tag{3.77}$$

Para el sistema de estudio el control PID se programa en Simulink de Matlab Figura 3.16, de la misma forma que para el control con redes neuronales este control se incorpora al modelo matemático (Figura 3.17) planteado para así observar su comportamiento de acuerdo al perfil y para cuando se presentan perturbaciones con respecto al tiempo.

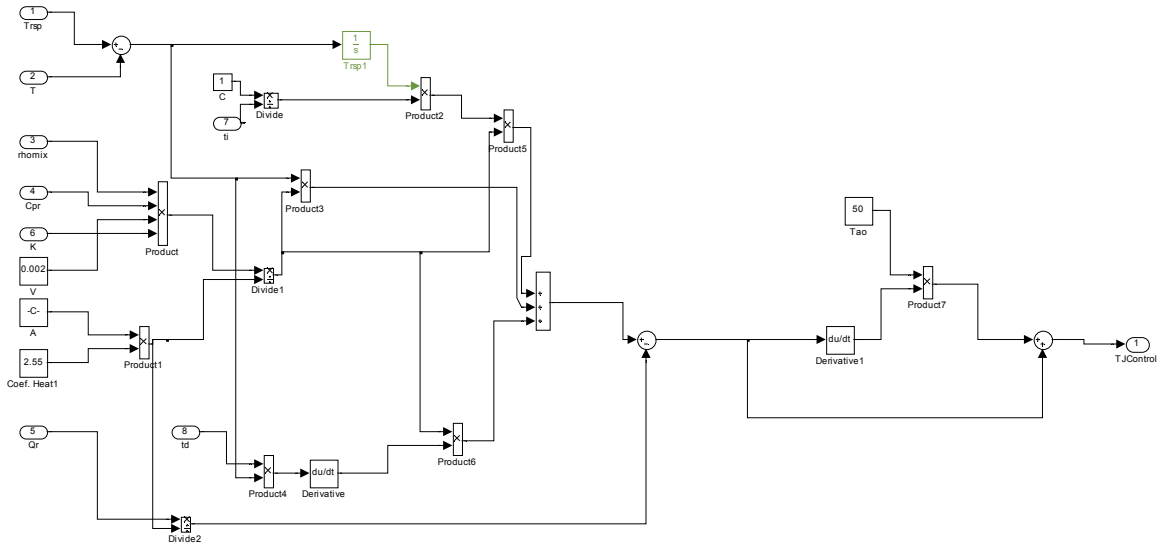


Figura 3.16 Control PID programado en Simulink

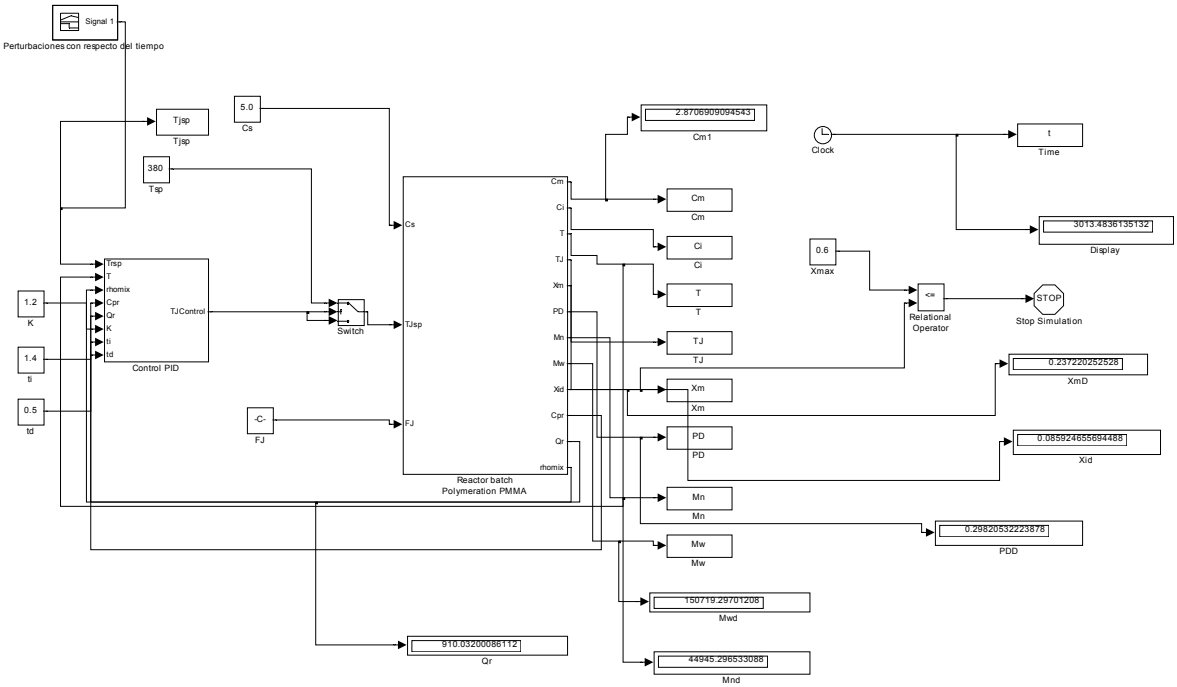


Figura 3.17 Control PID

3.1.7 Control GMC

El control PID puede ser una alternativa buena para sistemas de tipo continuo pero para sistemas de tipo batch este tipo de control no puede dar buenos resultados para esto existe un control de tipo *Generic Model Control (GMC por sus siglas en*

inglés), el cual puede ser una alternativa viable para la aplicación en sistemas intermitentes o por lotes como se plantea en este trabajo, pero al igual que el control PID el control GMC requiere de una sintonización del mismo esto debido a la ley de control que posee, la ley de control del GMC tiene como característica dos constantes de control las cuales pueden tener analogía con las constantes del control PID, es decir, el control GMC tiene sensibilidad en la obtención de dichas constantes de control. El control GMC posee una ley de control de la siguiente manera.

$$\frac{dx}{dt} = k_1\left(x_{sp} - x\right) + k_2 \int_0^t \left(x_{sp} - x\right)dt$$

(3.78)

Para el modelo del PMMA se establece la siguiente ley de control

$$T_J = T + \frac{\rho_r Cp_r V_r}{UA}\left(k_1\left(T_{rsp} - T\right) + k_2 \int_0^t \left(T_{rsp} - T\right)dt \right) - \frac{Q_r}{UA}$$

(3.79)

De la misma manera que para el control PID el control GMC (Figura 3.18) se programa en Simulink para determinar su funcionamiento para el modelo matemático de la obtención del PMMA Figura 3.19.

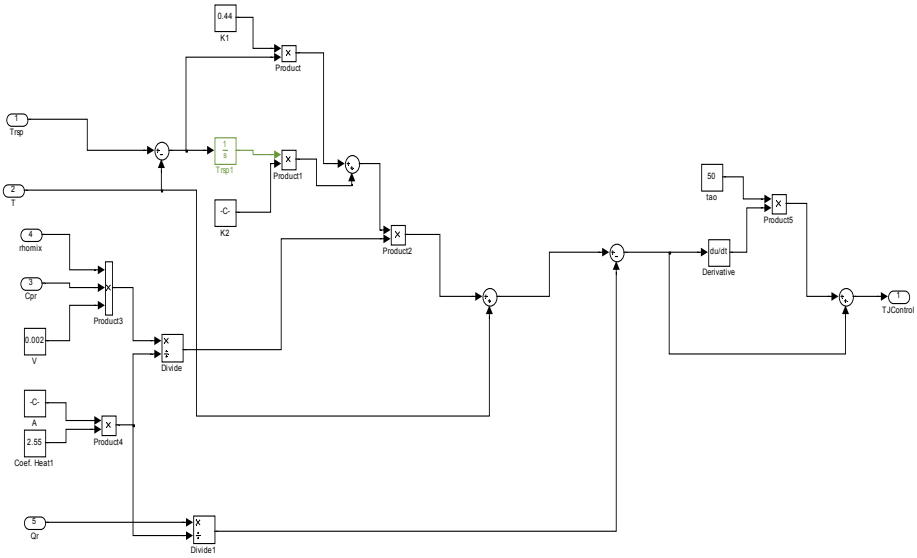


Figura 3.18. Control GMC programado en Simulink

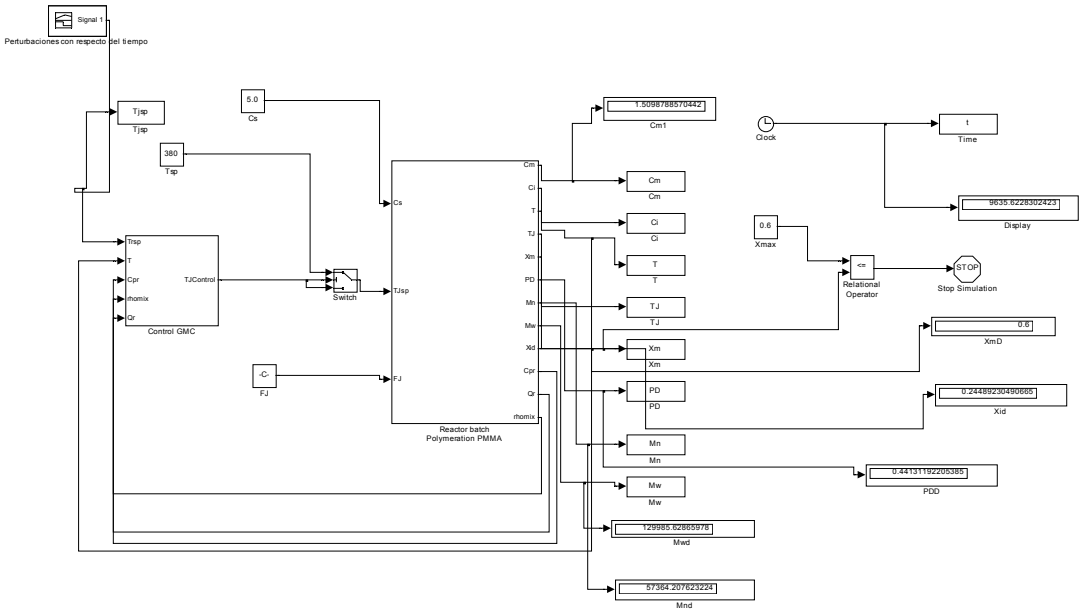


Figura 3.19 Control GMC en un reactor de polimerización

3.1.8 Control Neuronal Evolutivo

Para el control neuronal evolutivo se establece una arquitectura como base la cual consta de una neurona de entrada, una neurona en la capa oculta con diferente número de salidas (2-4) y una neurona de salida. Diferentes referencias proponen como algoritmo de evolución a la función de excitación de la red neuronal únicamente con la variación en la pendiente de la función de excitación, generalmente la función de excitación más usada es la función tangencial de tipo sigmoidea la cual presenta dos umbrales en la pendiente (1, -1), donde la ecuación se representa como una razón de un exponencial ecuación 3.80, en donde la pendiente se representa por φ y los valores de los pesos sipnaticos se representar por w_i, Figura 3.20

$$f(w) = \frac{\exp(-\varphi \cdot w_i) + \varphi \exp(\varphi \cdot w_i)}{\exp(-\varphi \cdot w_i) - \exp(\varphi \cdot w_i)} \tag{3.80}$$

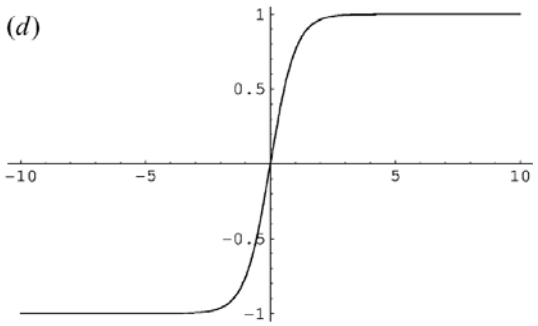


Figura 3.20 Función Tangencial de Tipo Sigmoidea

Helmult y col. (2002) Proponen el uso de diferentes tipos de funciones de activación como algoritmo de evolución dentro de las cuales se encuentran la función tangencial y logarítmica entre otras.

Matthias Ihme y col (2008). Del MIT demuestran que la optimización de una red neuronal mediante funciones de activación da como resultado una red con 117 conexiones y el mismo número de neuronas.

J. Parker y col. (2002) del MIT. Demuestran la capacidad de adaptación de una función de excitación de tipo sesgo radial (*RBF Radial Basis Function, por sus siglas en inglés*) el cual genera una red neuronal óptima pero no la mejor para un tipo de proceso de medición de temperatura.

La modificación de la pendiente implicaría que únicamente la trayectoria de los pesos sinápticos puede seguir una evolución para cada neurona en las capas ocultas, esto debido a que los umbrales de la función excitación permanecen de forma constante Figura 3.21.

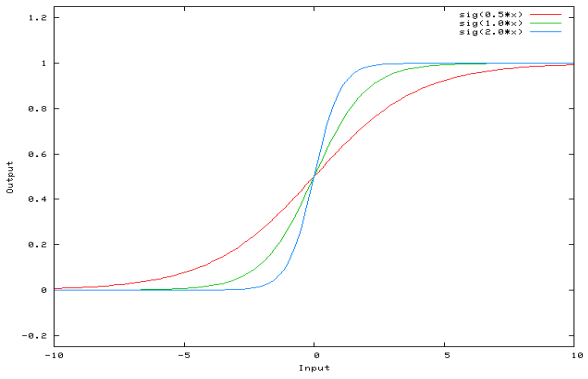


Figura 3.21 Diferentes pendientes en la función excitación tangencial

Además del uso de la función tangencial otros autores proponen el uso de funciones de excitación de tipo logarítmica sigmoidea (Figura 3.22), donde dicha función es un valor inverso exponencial con dependencia de la pendiente (φ) y de los pesos sinápticos (w_i), ecuación 3.81.

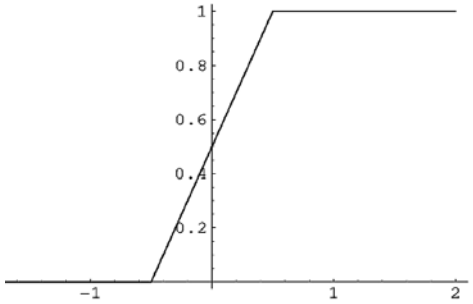


Figura 3.22 Función Logarítmica Sigmoidea

$$f(w) = \frac{1}{1+\exp(-\varphi \cdot w_i)} \tag{3.81}$$

La función de base radial es una función que calcula la distancia euclidiana de un vector de entrada *w* respecto de un centro *c* y tiene como función la Ec. 3.81

$$f(w) = \sum_{i=1}^{n} w_i \Phi\left(\left\| w - w_{ci} \right\|\right) \tag{3.82}$$

Donde w_i son los vectores de peso de la red neuronal w_{ci} es el vector de peso con respecto a un centro y Φ es la distancia euclidiana la cual depende de método usado para establecer su mecanismo de cálculo de distancia desde el centro al vector de pesos de entrada (Ec. 3.83-3.88).

Función Gaussiana

$$\Phi(w) = \exp(\varphi \cdot w_i)^2 \tag{3.83}$$

Función Multi - cuadrática

$$\Phi(w) = \sqrt{1+(\varphi \cdot w_i)^2} \tag{3.84}$$

Función Multi- cuadrática inversa

$$\Phi(w) = \frac{1}{\sqrt{1+(\varphi \cdot w_i)^2}} \tag{3.85}$$

Función Spline poli armónico

$$\Phi(w) = w_i^k \qquad k = 1,3,5,.... \tag{3.86}$$

$$\Phi(w) = w_i^k \ln(w_i) \qquad k = 2,4,6,.... \tag{3.87}$$

Función Spline placa delgada

$$\Phi(w) = w_i^2 \ln(w_i) \tag{3.88}$$

Para este trabajo se emplea la función de tipo Gaussiana (Figura 3.23) teniendo en cuenta la evolución de la pendiente φ dentro del intervalo establecido.

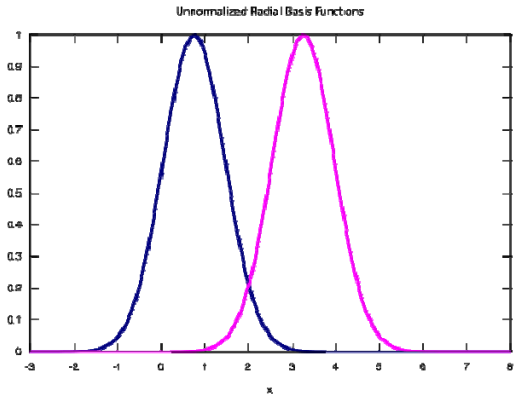


Figura 3.23. Función de tipo Base Radial (RBF)

Al hacer uso de las diferentes funciones de excitación dentro del control se debe proponer una estructura inicial o arquitectura de la red neuronal, para este trabajo se propone una red con una neurona de entrada, una capa oculta que contiene una neurona y una neurona de salida, siendo el número de salidas la variable a manipular conjuntamente con el valor de la pendiente de la función de excitación.

Existen ciertas características de este espacio de búsqueda que convierten a los algoritmos evolutivos en candidatos perfectos a la hora de localizar estructuras de redes neuronales artificiales óptimas. Dichas características son (Miller y col, 1989):

a) El espacio de búsqueda es infinitamente grande, dado que no se debe establecer límite al número de posibles nodos y conexiones
b) El espacio es no diferenciable con respecto a número de nodos y conexiones, pues los cambios en tales valores son discretos y a su vez pueden provocar un efecto de discontinuidad en la salida ofrecida por la red.

c) Es un espacio complejo y en el que aparece ruido; dado que la condición en un cromosoma de una determinada arquitectura impide relacionar directamente a ésta con su comportamiento, hace que surjan efectos de fuerte epistasia y que sea dependiente del método de evaluación utilizado.
d) El espacio de búsqueda es engañoso. Arquitecturas similares pueden dar lugar a resultados muy distintos con respecto al objetivo buscado.
e) El espacio de búsqueda es multimodal. Al contrario del punto anterior, arquitecturas muy diferentes entre sí pueden tener comportamientos muy parecidos.

3.2 Parte Experimental

En la siguiente sección se presenta la metodología seguida para la obtención de resultados del control de temperatura en un reactor batch de laboratorio.

3.2.1 Experimentación

Los reactivos son meti-metacrilamida (MMA, Aldrich 99%) el cual puede ser purificado si es pasada sobre alúmina básica o mediante una agitación vigorosa con NaOH durante una hora hasta alcanzar un pH de 7.0, posteriormente se deja agitar la solución de Metil metacrilato durante aproximadamente una hora con el peróxido de bezoilo (PBO) sin calentamiento, se adiciona el solvente tolueno en una relación de 80/20 %v/v para iniciar la polimerización (Figura 3.24). Se espera que se cumpla el mecanismo de reacción de la Figura 3.25.

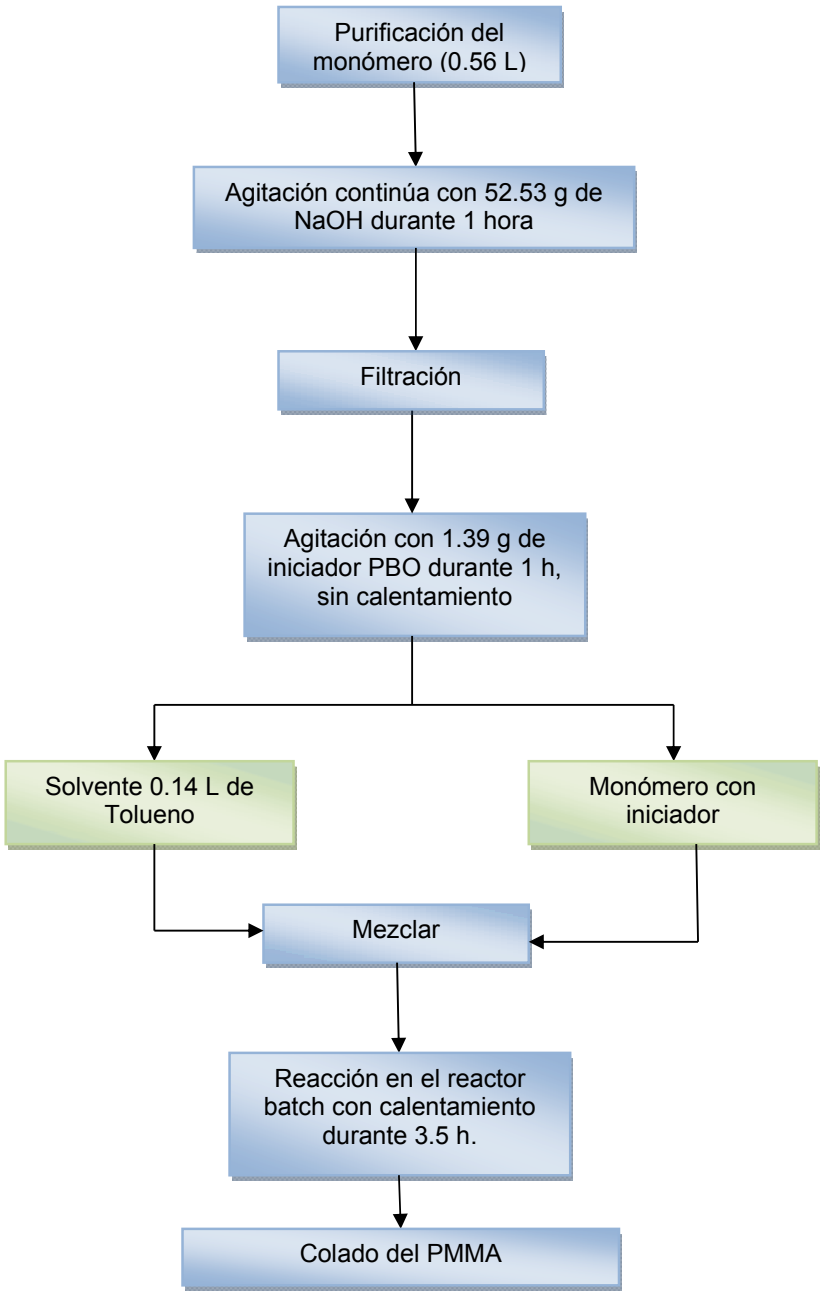


Figura 3.24 Proceso de Preparación del PMMA

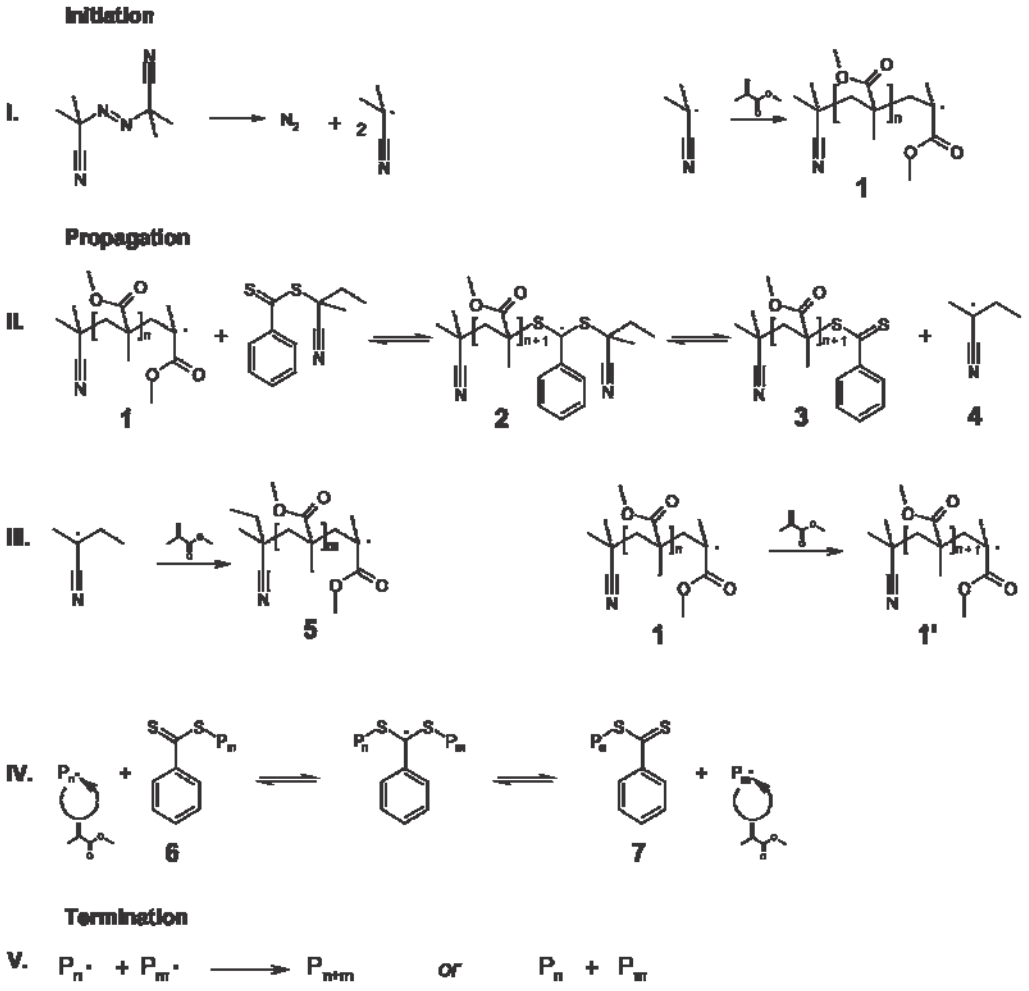


Figura 3.25 Mecanismo propuesto experimental

Descripción del equipo experimental

El equipo está conformado por:

- Reactor batch LR-2.10 marca Eurostar, tiene una capacidad de 4 litros. El reactor cuenta con una varilla agitadora Power control-visc P7 con velocidad de agitación de 8-290 rev/min., su microprocesador cuenta con interfase RS232 con conector serie hembra de 9 pines para la transferencia de datos.
- Baño recirculador enfriador/calentador marca Polyscience, capacidad de 13 litros, con un rango de temperatura de -30 a 200 ºC, sus partes sumergibles son de acero inoxidable 300 para resistir la corrosión. Cuenta con interfase RS232 con conector serie hembra de 9 pines para la transferencia de datos.

- Termómetro de resistencia RTD tipo Pt100 clase B, salida de corriente lineal de 4 a 20 mA y rango de temperatura de -50 a 200 ºC. Marca Kobold.
- Tarjeta de adquisición de datos (DAQ) NI 9219 con chasis de cDAQ 9178 de 8 sockets.
- Amplificador de voltaje tipo operacional LM324 no inversor.

Datos técnicos del reactor batch LR-2.10

El sistema IKA LR-2.10 es un sistema modular miniplanta reactor. Ha sido diseñado para simular y optimizar procesos con reacciones químicas y puede también ser usado para mezclado, dispersión y homogenización en modelado de procesos (Figura 3.26). Los siguientes datos son proporcionados por el fabricante del equipo y se encuentran en el manual de operación:

- El volumen de trabajo es de 500 a 2000 ml.
- Debido a su altamente resistente tapa hermética el medio en el recipiente del reactor puede ser calentado hasta 230 ºC, siendo ésta la temperatura máxima permitida en el recipiente del reactor.
- El aparato está concebido para un funcionamiento en vacío de hasta 25 mbar. No utilizar el aparato en sobrepresión.
- El recipiente del reactor sólo puede calentarse sin presión mediante el uso del revestimiento doble que esté dotado de termostatos o fuentes térmicas similares. En ningún caso puede utilizarse una campana ni una placa calefactora (riesgo de reventones).
- El recipiente del reactor esta hecho de acero inoxidadle 1.4571.
- No utilizar gases, vapores ni disolventes inflamables o explosivos en el reactor.
- El reactor sólo puede utilizarse con el interruptor de seguridad que se incluye en el volumen de suministro, así como con el disco de protección contra astillas.
- No poner en marcha el reactor si está abierta la tapa.
- El sistema reactor debe estar bien ventilado siempre que se trabaje con presión normal, pues de este modo se evitará la acumulación de presión debido a la existencia de gases altamente volátiles o al desarrollo de gradientes de presión impredecibles en la reacción. Condense los gases volátiles en un refrigerador que esté dotado de un tapón cónico esmerilado en la tapa del reactor.

Figura 3.26 Reactor batch marca Eurostar RL-2.10 con varilla agitadora

Baño recirculador enfriador/ calentador.

En la Tabla 3.2 se muestran importantes datos técnicos del baño recirculador enfriador/calentador marca polyscience proporcionados por el fabricante del equipo (Figura 3.27). Las especificaciones de rendimiento son determinadas a una temperatura ambiente de 20 ºC (68 ºF). Los conectores plásticos provistos con la unidad son ideales para aplicaciones de entre -40 ºC y 93 ºC. En aplicaciones sobre 93 ºC se recomienda conectores de acero inoxidable o Teflón.

Figura 3.27 Baño recirculador enfriador/calentador**.**

Tabla 3.2 Datos técnicos del baño recirculador enfriador/calentador.

Estabilidad de temperatura	±0.01 ºC
Controlador/RS232	Sí
Sonda de temperatura externa	Funcional en modelos programables No disponible en modelos digitales
Exactitud del indicador	LCD Gráfico, ºC o ºF, ±0.25 ºC
Calentador	1100 W - 115 V, 2200 W - 240 V
Presión máxima del flujo	30 L/min/5.0 psi (60 Hz) 22 L/min/4.3 psi (50 Hz)
Succión máxima del flujo	22 L/min (60 Hz); 15 L/min (50 Hz)
Protección temperatura elevada	Sí, ajustable por el usuario
Protección nivel líquido bajo	Sí
Velocidad de la bomba	Ajustables por el usuario
Conexiones de entrada y salida de la bomba	Descarga trasera de ¼ de pulgada hembra
Diámetro interno (DI) de los conectores de la entrada y salida de la bomba	½ in (13 mm)
Temperatura ambiente	5 – 30 ºC
Humedad relativa	80 % hasta 30 ºC
Tipo de material en sus partes sumergibles	Acero inoxidable 300

3.2.2 Adquisición de datos

El chasis USB NI CompactDAQ es un equipo con sencillez plug-and-play de USB a las medidas de sensor y eléctricas. Cuenta con una, cuatro y ocho ranuras, estos chasis USB NI CompactDAQ están diseñados para sistemas pequeños y portátiles de medidas mixtas en el laboratorio o en campo.

Diseño Mecanico NI CompactDAQ

- Construcción de metal A380 para durabilidad
- 30 g de shock y 0.3 g de vibración operacional de acuerdo con el IEC-60068-2-27/64
- Temperatura de operación de -20 a 55 °C
- Juegos de montaje en panel, en rack, en riel DIN y en escritorio (Figura 3.28).

Figura 3.28 Chasis de NI CompactDAQ de 4 y 8 ranuras

Múltiples Motores de Temporización para Múltiples Velocidades

- El chasis NI CompactDAQ tiene tres motores de temporización de entrada analógica. Esto permite a los programadores dividir todas sus entradas analógicas en hasta tres grupos diferentes conocidos como "tasks".
- Cada tarea puede ejecutar una velocidad distinta (Figura 3.29). Esto es ideal cuando se combinan medidas de temperatura, lo cual es generalmente lento, con medidas de más alta velocidad como sonido y vibración.
- Las tres tareas operan de manera independiente, pueden ser direccionadas desde ciclos o threads distintos en un programa y pueden ser iniciadas simultáneamente.

- Todos los canales en una sola tarea son sincronizados automáticamente. En caso de que un módulo multiplexado sea combinado en una tarea con un módulo de muestreo simultáneo, el primer canal en el módulo multiplexado es sincronizado y los canales subsecuentes en el módulo multiplexado se escanean en sucesión.
- Todos los canales en una sola tarea, simultánea o multiplexada, son regresados a la velocidad de muestreo solicitada.
- Si lo desea, todos los módulos pueden ser colocados en una sola tarea. Esto sincroniza todos los canales en el mismo reloj.

Figura 3.29 Diferentes tareas a velocidades variables.

Tarjeta NI 9219

La tarjeta 9219 de National Instruments es un módulo universal de la Serie C de 4 canales diseñado para pruebas de usos múltiples en cualquier chasis NI CompactDAQ o CompactRIO. Con el NI 9219 se puede medir varias señales desde sensores como galgas extensiométricas, RTDs, termopares, celdas de carga y otros sensores. Los canales son seleccionados individualmente, esto para poder realizar un tipo de medida diferente en cada uno de los cuatro canales. Los intervalos de medida difieren para cada tipo de medida e incluyen hasta ±60 V para voltaje y ±25 mA para corriente (Figura 3.30).

Debido al diseño del controlador, el NI 9219 no limita la velocidad total de un sistema NI CompactDAQ cuando se usa con módulos de muestreo más rápidos. Con aislamiento entre canales de 250 Vrms, el NI 9219 protege no solo los módulos

alrededor, chasis y sistemas de cómputo conectados sino también los otros canales en el mismo módulo. Además para aumentar la seguridad, el aislamiento entre canales elimina los problemas asociados con lazos a tierra.

Usa conectores de terminal de resorte de 6 posiciones en cada canal para conectividad directa de la señal. Usted puede comprar conectores adicionales para reducir el tiempo de conexión de señal para múltiples unidades de pruebas. Además de los conectores extra, un juego de liberación de tensión está disponible para asegurar los cables de señal.

Figura 3.30 Modulo NI 9219

Termómetro de resistencia

Este instrumento de medición de temperatura consiste en un RTD que se basa en el principio según el cual la resistencia de todos los metales depende de la temperatura. La elección del platino en los RTD de la máxima calidad permite realizar medidas más exactas y estables hasta una temperatura de aproximadamente 500 ºC. Los RTD más económicos utilizan níquel o aleaciones de níquel, pero no son tan estables ni lineales como los que emplean platino.

Para este trabajo se utiliza un RTD de 2 hilos (Figura 3.31-3.32), esto debido en un principio a la poca cantidad de canales de muestreo que se tenia con la tarjeta de adquisicion de datos anterior, aunque la exacitud de la medicion no depende del numero de canales de muestreo si depende del instrumento usado.

Figura 3.31 RTD de dos hilos para temperatura del reactor

Figura 3.32 RTD de dos hilos para temperatura ambiente

Amplificador de Voltaje

Un amplificador operacional es un amplificador diferencial. Desde el punto de vista de una señal, el Amp. Op. tiene tres terminales: dos terminales de entrada y un terminal de salida. La Figura 3.33 muestra el símbolo utilizado para representar el Amp. Op. Los terminales 1 y 2 son los terminales de entrada, y el terminal 3 es el de salida.

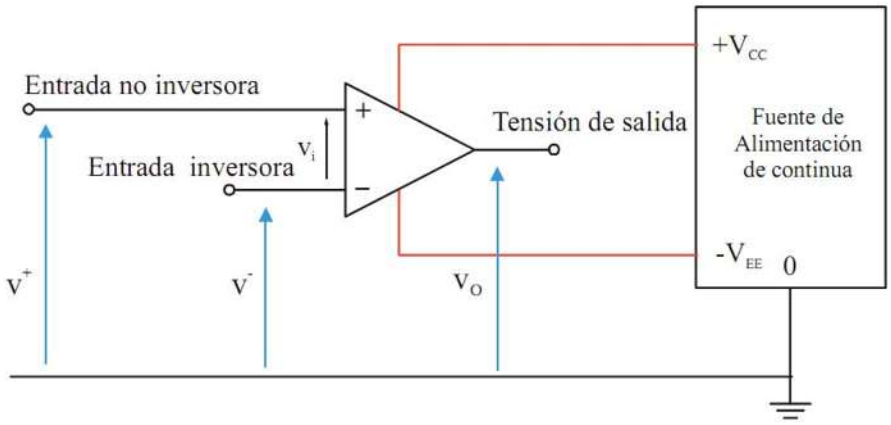


Figura 3.33 Amplificador Operacional

Como el Amp. Op. es un dispositivo activo (está formado por transistores, resistencias y algún condensador), requiere una potencia de continua para funcionar. La mayoría de Amp. Op. de circuito integrado requieren dos fuentes de continua, como se muestra en la Figura 3.34. Los terminales, 4 y 8 del operacional se conectan a un voltaje positivo, + V_{cc}, y a uno negativo, -V_{cc}, respectivamente, siendo habitual que sean iguales en valor absoluto. Las dos fuentes de alimentación de continua presentan una tierra común. Es interesante observar que el punto tierra de referencia en los Amp. Op. es precisamente el terminal común de las dos fuentes de alimentación; esto es, ningún terminal del Amp. Op. se conecta físicamente a tierra.

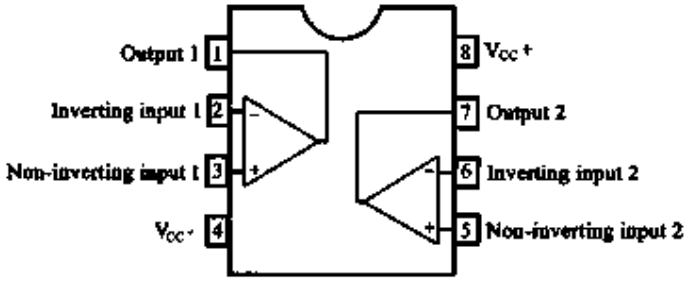


Figura 3.34 Circuito Integrado Amplificador Operacional

Para la aplicación de amplificación de intensidad medida por el RTD en el sistema de trabajo, se muestra en la Figura 3.35.

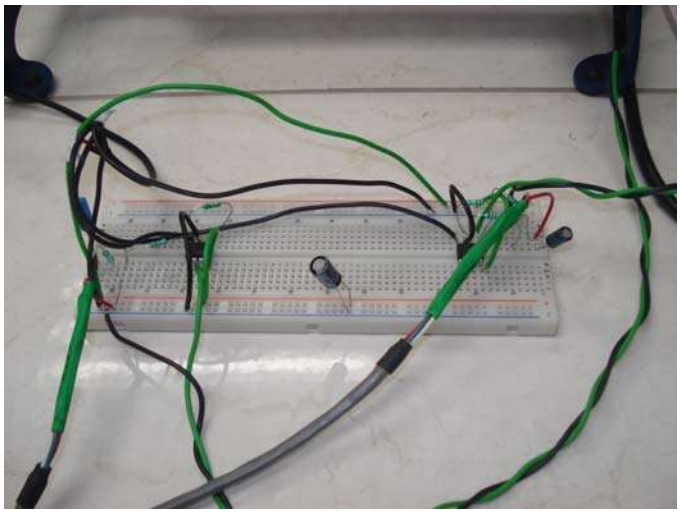


Figura 3.35 Amplificador Operacional físico.

3.2.3 Instrumentación Virtual

El propósito de adquisición de datos en un instrumento virtual es medir un fenómeno eléctrico y físico como voltaje, corriente, temperatura, presión o sonido. La adquisición de datos basada en PC utiliza una combinación de hardware modular, software de aplicación y una PC para realizar medidas. Mientras cada sistema de adquisición de datos se define por sus requerimientos de aplicación, cada sistema comparte una meta en común de adquirir, analizar y presentar información. Los sistemas de adquisición de datos incorporan señales, sensores, actuadores, acondicionamiento de señales, dispositivos de adquisición de datos y software de aplicación.

La mayoría de las tarjetas de adquisición de datos, realizan comunicación de datos a través de los puertos serial, paralelo, USB y otros lo que permite involucrar la instrumentación virtual por medio de una interfaz gráfica en una PC.

El concepto de instrumentación virtual no se limita a una medición de voltaje o corriente sino que también involucra el procesamiento, análisis, almacenamiento, distribución y despliegue de los datos e información relacionados con la medición de una o varias señales especificas; es decir el instrumento virtual no se conforma con la adquisición , sino que involucra la interfaz hombre-máquina, las funciones de análisis y procesamiento de señales, las rutinas de almacenamiento de datos y la comunicación con otros equipos.

Programación Modular

La programación modular se basa en dividir el programa en partes que tengan una personalidad propia, es decir, en dividir el programa en LabVIEW en varios subprogramas que ahorren tiempo y esfuerzo a la hora de realizar y ejecutar el programa.

Los subprogramas tienen las mismas propiedades que un programa y se utilizan mediante la creación de iconos y conectores que facilitan la lectura y la interpretación del instrumento virtual global. En el proyecto se utilizará una mezcla de ambas programaciones, tanto modular como estructurada. Esta técnica de programación modular estructurada se establece para el control de temperatura del reactor, mediante la siguiente secuencia:

- Protocolo de comunicación entre LabView y el agitador
- Protocolo de comunicación entre LabView con el baño calentador/enfriador.
- Adquisición de datos de voltaje de RTD dentro del reactor.
- Control basado en redes neuronales evolutivas.

Protocolo del agitador.

Para la comunicación entre el agitador y LabView se utiliza del menú de funciones la función VISA (Figura 3.36) la Virtual Instrument Software Architecture (VISA) es un estándar para configurar, programar y depurar sistemas de instrumentación que comprenden interfaces GPIB, VXI, PXI, serial (RS232/485), Ethernet/LXI, USB y/o IEEE 1394. Algunas versiones incluyen LXI autodiscovery con información mejorada sobre el dispositivo LXI en Measurement & Automation Explorer (MAX), configuraciones de VISA Conflict Manager en MAX, soporte adicional para Mandriva Linux 2009, soporte para open SUSE 11.0 y arquitectura insertable VISA de múltiples proveedores para la versión de 64 bits de Windows Vista.

NI-VISA es la implementación de National Instruments del estándar de E/S VISA. Proporciona la interfaz de programación entre el hardware y los entornos de desarrollo de aplicaciones como NI LabVIEW, LabWindows™/CVI y Measurement Studio para Microsoft Visual Studio.

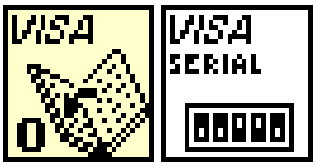


Figura 3.36 Protocolo VISA

El protocolo VISA se conecta de forma secuencial a la interfaz del agitador mediante el siguiente módulo (Figura 3.37)

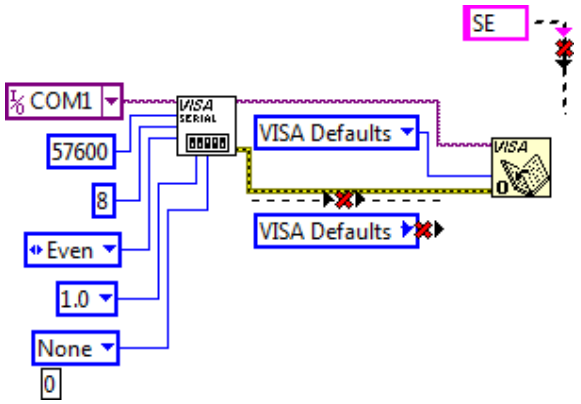


Figura 3.37 Modulo de comunicación Agitador-LabView

Para establecer la comunicación y el control del agitador desde la interfaz virtual se crea un instrumento virtual para el control de las rpm, el instrumento se establece mediante un medidor operacional (Figura 3.38), el cual debe tener comunicación con la interfaz virtual de LabView (Figura 3.39).

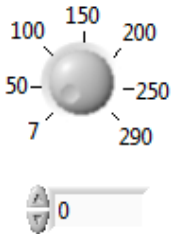


Figura 3.38 Instrumento Virtual.

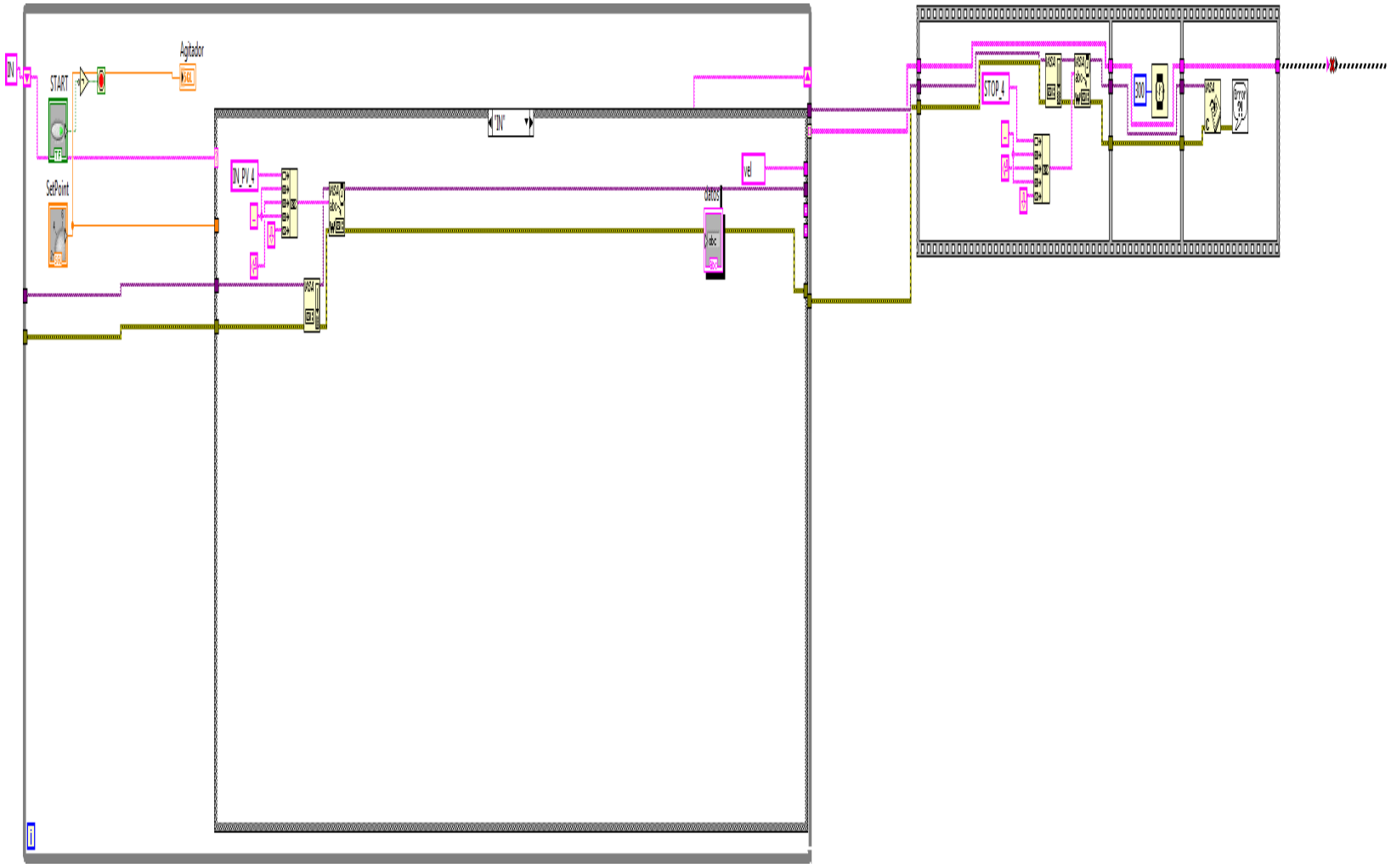


Figura 3.39 Interfaz Virtual en LabView

Protocolo Baño calentador/enfriador.

Para establecer la comunicación del baño calentador/enfriador que tiene un protocolo RS232, se implementa el mismo módulo virtual de comunicación virtual como lo es VISA, esto mediante la siguiente secuencia de operaciones (Figura 3.40), de igual forma que para el agitador se implementa un instrumento virtual para el control de la temperatura del baño calentador/enfriador (Figura 3.41), y la secuencia modular de instrumento virtual se muestra en la Figura 3.42.

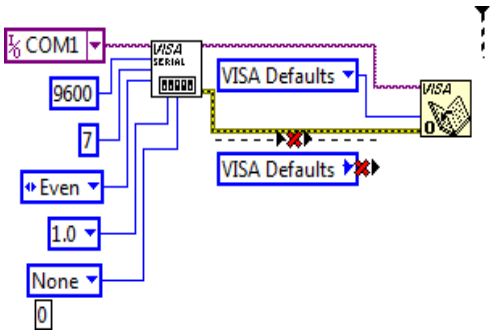


Figura 3.40 Protocolo de comunicación LabView – Baño calentador/enfriador

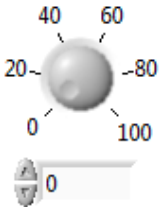


Figura 3.41 Instrumento Virtual control de Temperatura

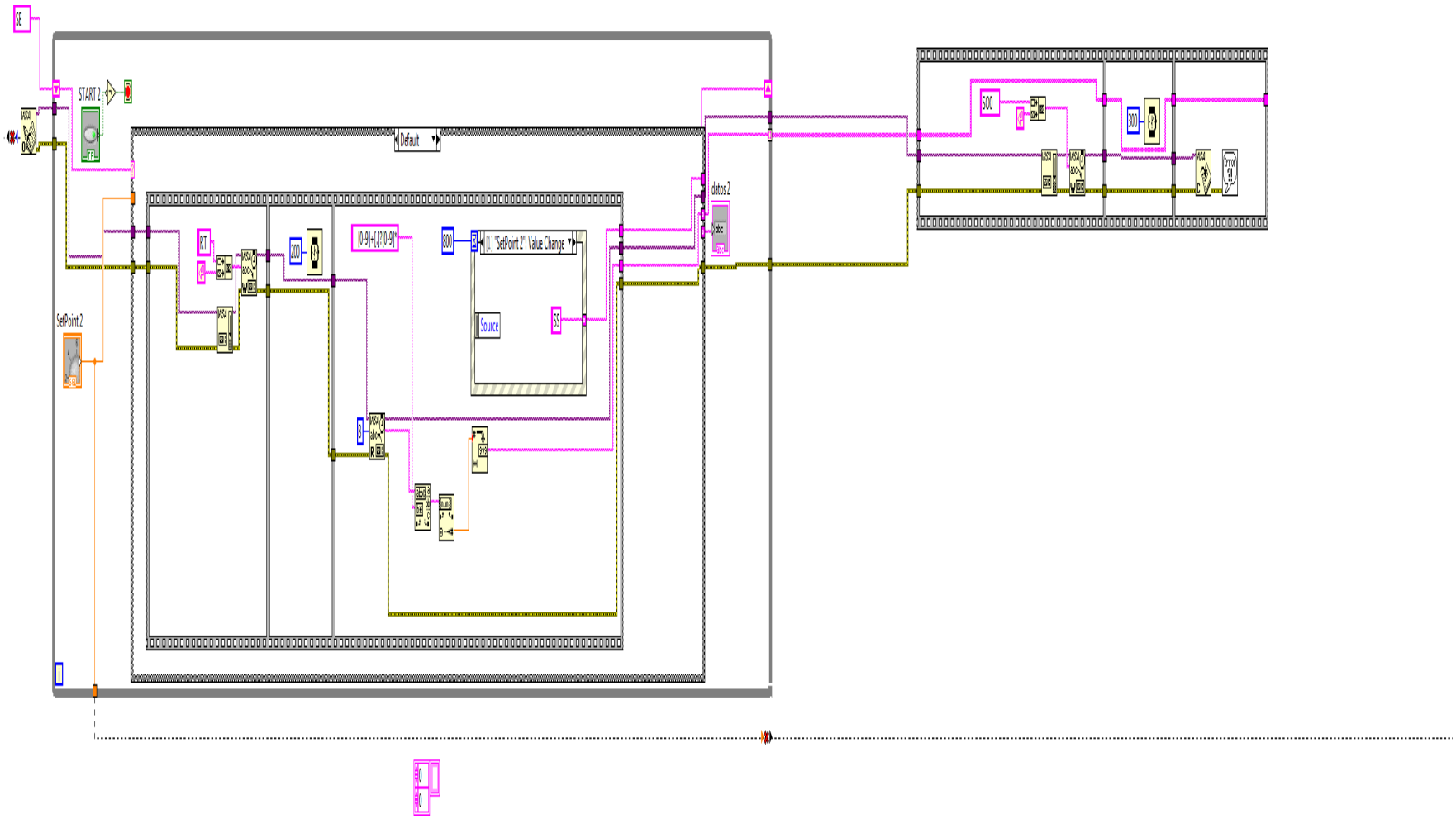


Figura 3.42 Interfaz Virtual Baño Calentador/Enfriador

Adquisición de datos del RTD

La adquisicion de datos del voltaje se realiza mediante la implementacion de la tarjeta NI 9219 explicada con anterioridad, el intrumento virtual para la adquisicion de datos usado fue el DAQ Assistant (Figura 3.43), el asistente de adquisición de datos de LabView puede ser programado para establecer el número de canales a usar de la tarjeta, así como la velocidad de muestreo (Figura 3.44)

Figura 3.43 Asistente de Adquisición de datos

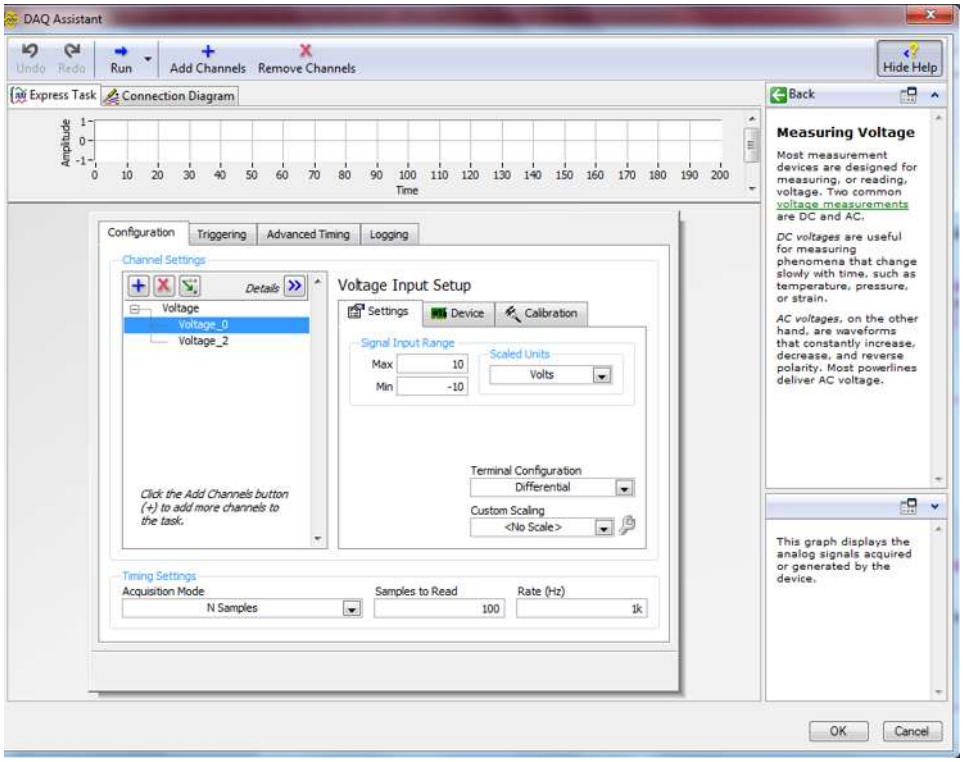


Figura 3.44 Configuración del Asistente de adquisición de datos

Una vez configura la tarjeta para la adquisición de datos desde la interfaz virtual se establece los módulos secuenciales para el tratamiento de la señal adquirida por la tarjeta (Figura 3.45) y así poder establecer el método de control.

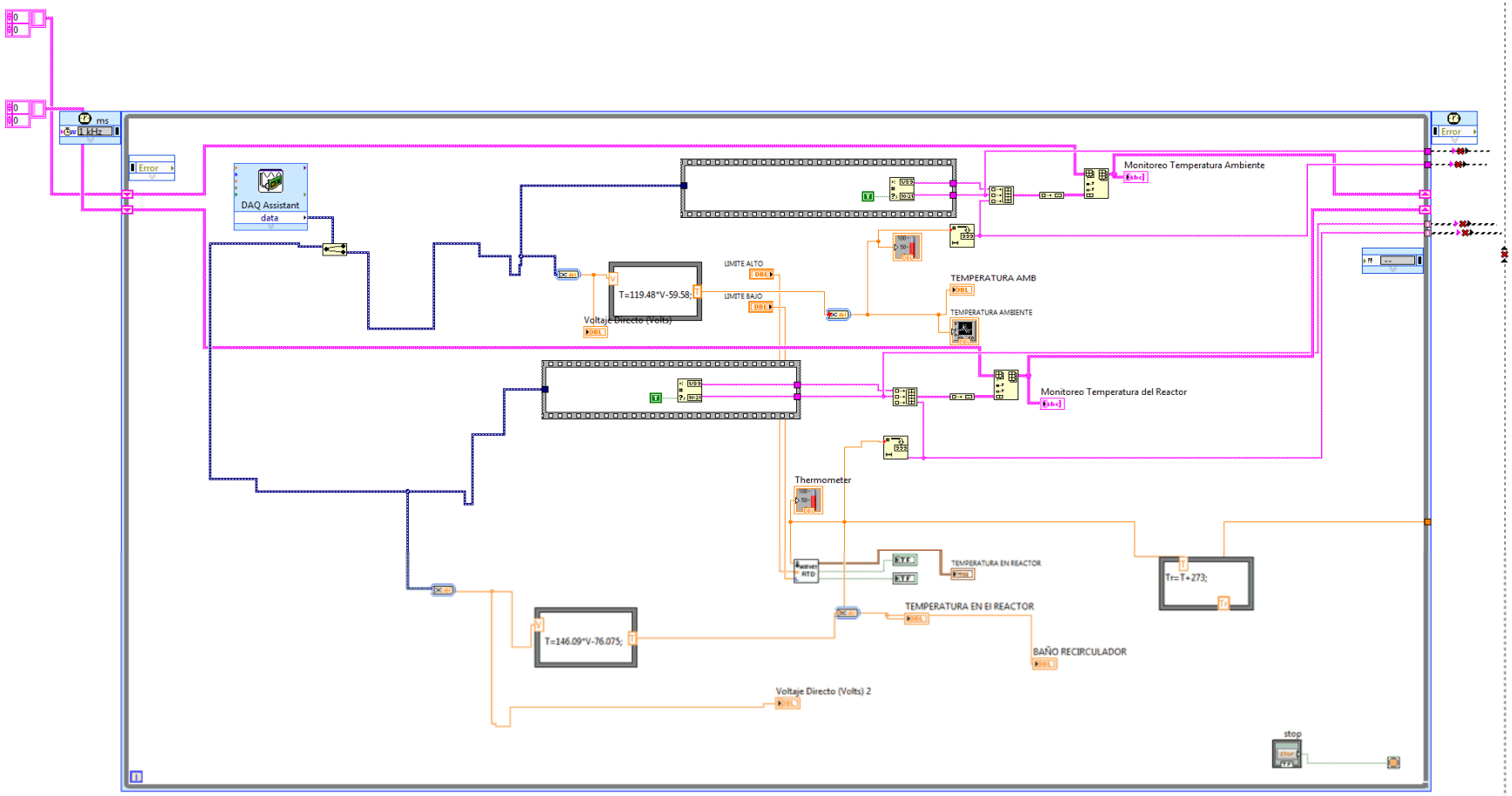


Figura 3.45 Interfaz Virtual de Adquisición de datos

Para realizar la conversión del voltaje medido por el RTD a mediciones de temperatura, se establece un método de calibración del RTD a partir de diferentes temperaturas, esto teniendo en cuenta que el incremento de la temperatura es directamente proporcional al incremento de voltaje por lo cual la ecuación de calibración-conversión tendrá una tendencia lineal con respecto al voltaje medido, esto se realiza tanto para el RTD de la temperatura del reactor (Ec. 3.89) como para el RTD de la temperatura ambiente (Ec. 3.90).

$$T_R = 146.09V_R - 76.075 \tag{3.89}$$

$$T_{amb} = 119.48V_{amb} - 59.58 \tag{3.90}$$

Donde T_R denota la temperatura del reactor a partir del voltaje medido dentro del reactor V_R, de igual forma para T_{amb} y el V_{amb}, de esta manera la interfaz de adquisición de datos se encuentra completada (Figura 3.46-3.47).

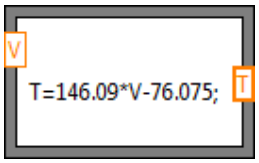


Figura 3.46 Ecuación de Temperatura del Reactor en LabView

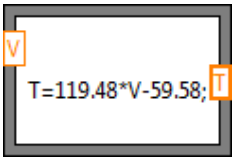


Figura 3.47 Ecuación de Temperatura Ambiente en LabView

3.2.4 Control con redes Neuronales Evolutivas en LabView

La plataforma de instrumentos virtuales de LabView no cuenta con un instrumento especificó para el diseño de un controlador basado en redes neuronales artificiales, cabe indicar que existen algunas librerías que contienen modelos de redes neuronales (Ponce, 2009) pero estas no son de acceso libre en LabView, por esta razón para este trabajo se planteó el diseño de la red neuronal desde sus aspectos básicos de adquisición de datos y entrenamiento en línea. Los parámetros usados para la programación de la red neuronal se establecieron a partir del modelo

matemático de la red neuronal de tipo multicapa (MLP, por sus siglas en inglés), además se realizó una analogía con las secuencias establecidas en Matlab de MathWorks en su toolbox de redes neuronales, considerando los parámetros de evolución ya establecidos en esta metodología, este modelo establece un aprendizaje en línea a partir de datos experimentales obtenidos mediante medición de temperatura, los intervalos de temperatura se pueden establecer como los límites máximos de temperatura y el limite mínimo de temperatura los cuales se logran a partir de un instrumento virtual de temperatura o de medición.

Una de las limitantes que se estableció en el diseño de la red neuronal en LabView fue la arquitectura de la misma y el valor de la pendiente óptimo que demostró mejor respuesta de la red neuronal, a diferencia de Matlab en LabView se parte de una arquitectura ya establecida sin tener la posibilidad de modificación durante el transcurso de la aplicación de la red neuronal, por esta razón se establece mediante el proceso de simulación que la red neuronal con una arquitectura básica de una neurona de entrada, una neurona en la capa oculta y una neurona en la capa de salida es la indicada para el proceso de evolución dentro del proceso, además que la red debe contener una pendiente de una función de excitación definida, una de las variables de evolución es la arquitectura de la red neuronal que tiene como punto principal en número de salidas de la red neuronal para este trabajo y específicamente para el diseño del controlador se establece que el número de salidas óptimo es de una cantidad de cuatro salidas de cada neurona evolucionada con una sola entrada para cada neurona evolucionada.

Los estados de aprendizaje de la red neuronal en LabView no se pueden establecer de forma sencilla pero se realiza una estimación de la cantidad de estados de aprendizaje mediante una simulación de los parámetros medidos en el toolbox de redes neuronales de Matlab. Los parámetros medidos para la simulación y establecimiento de los estados de aprendizaje son la temperatura del reactor, temperatura de la chaqueta y temperatura del ambiente, con esta variables muestreadas en intervalos de 2 s, se puede determinar el número de estados de aprendizaje de la red.

La realización de esta secuencia de programación modular para el control con redes neuronales artificiales evolutivas se observa en la Figura 3.48.

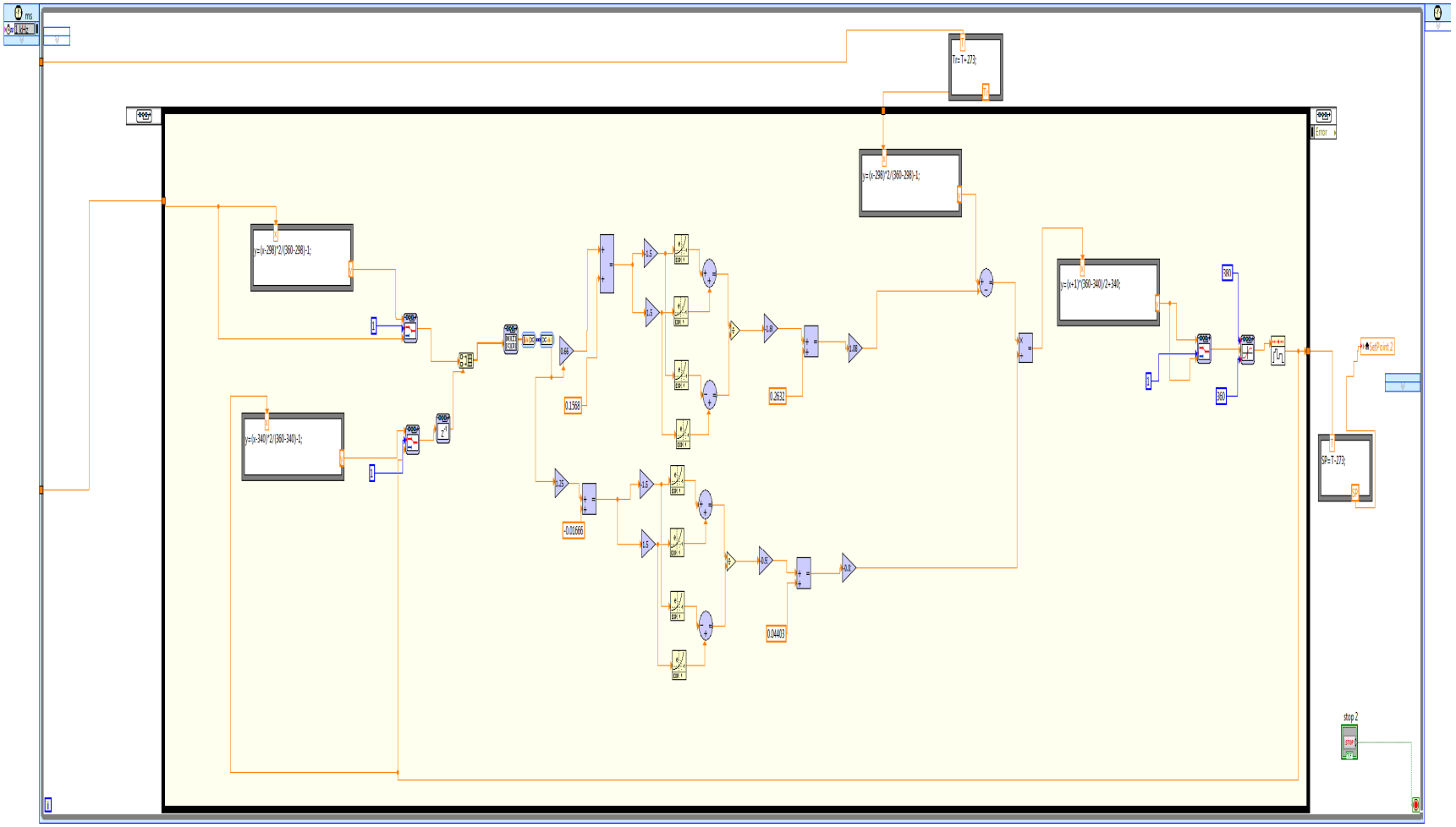


Figura 3.48 Interfaz del Control Neuronal Evolutivo en LabView

Se pueden establecer interfaces de medición para cada sistema de medición contiene, es decir, medición de la temperatura del reactor (Figura 3.49), temperatura del baño calentamiento/enfriamiento (Figura 3.50), medición de la temperatura ambiente (Figura 3.51) y la trayectoria seguida por el agitador (Figura 3.52).

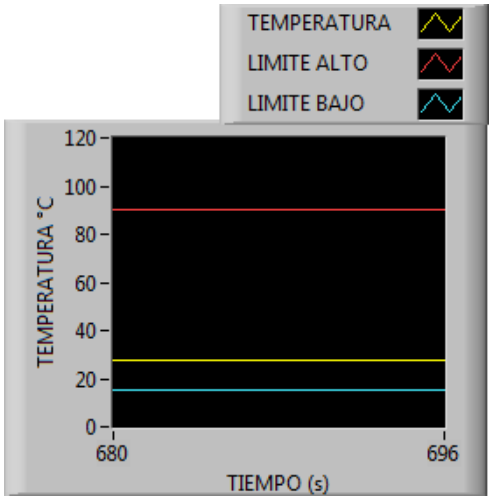


Figura 3.49 Medición de Temperatura del Reactor Interfaz de LabView

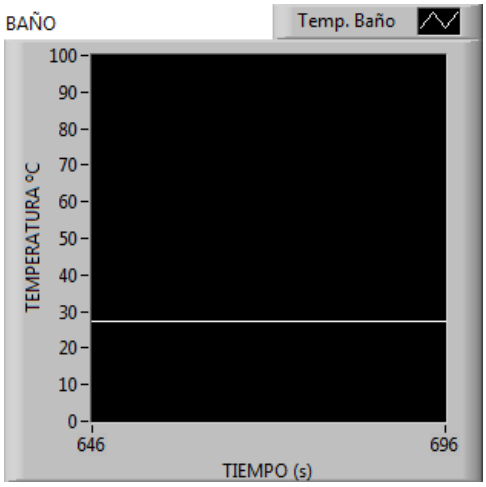


Figura 3.50 Medición de la Temperatura del Baño (C/E) interfaz de LabView

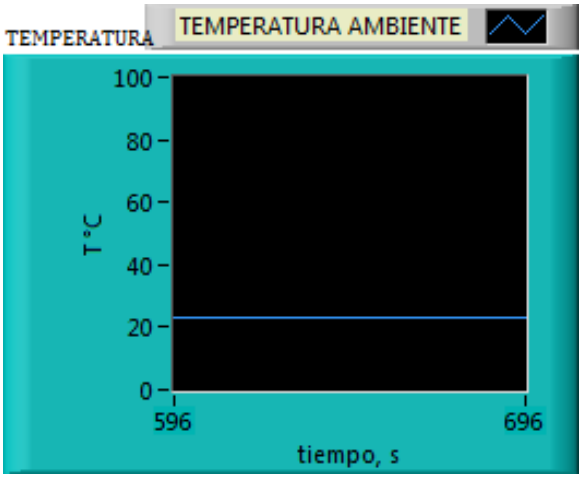


Figura 3.51 Medición de la Temperatura Ambiente interfaz de LabView

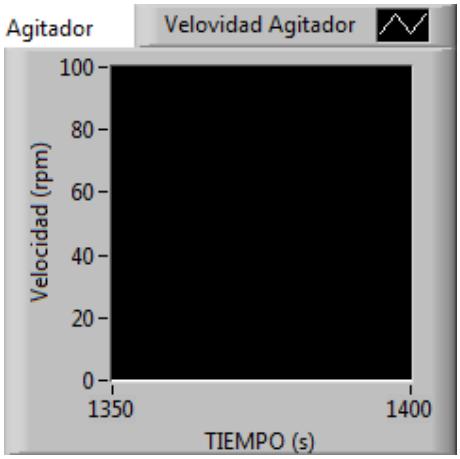


Figura 3.52 Medición de Agitador interfaz de LabView

Al unir las interfaces involcradas para cada una de la mediciones en LabView se puede establecer como PLC que monitorea cada una de las variables de interés con acciones de control (Figura 3.53).

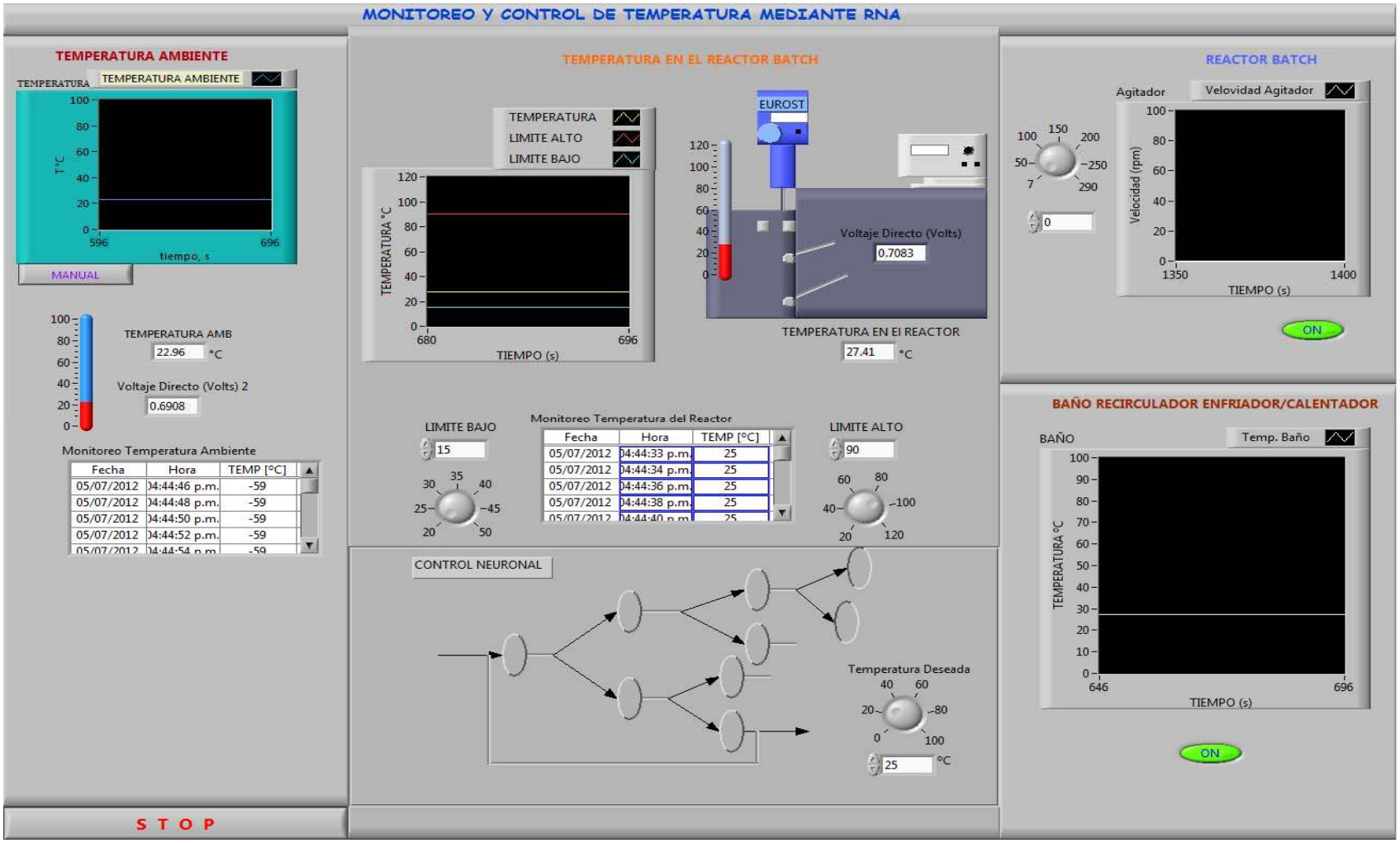


Figura 3.53 PLC estructurado en LabView para el reactor batch.

3.2.5 Perturbaciones al sistema experimental

Una de las maneras de comprobar la robustez del sistema del control es mediante la introducción de perturbaciones, para sistemas de tipo batch la introducción de perturbaciones provoca de las acciones de control sean complicadas de realizar, esto principalmente a la naturaleza del sistema. Para establecer el adecuado funcionamiento del sistema de control se provoca una perturbación al sistema de calentamiento, el cual consiste en retirar un volumen especifico de agua de calentamiento (8 L) para provocar un descenso de la temperatura del sistema de calentamiento induciendo de esta forma una perturbación de forma directa al sistema de reacción, la temperatura del sistema de calentamiento disminuye dentro de un orden de 5 K.

La comprobación del sistema de control consiste principalmente en observar la acción de control aplicada por el sistema programado de forma modular en LabView y que tiene como sistema de control a las redes neuronales evolutivas.

Capítulo 4

RESULTADOS

En esta sección se presentan los resultados obtenidos de las simulaciones de las redes neuronales evolutivas para el modelo matemático del poli-metil metacrilato (Parte teórica), así como los resultados obtenidos de forma experimental del control aplicado al reactor batch basado en rede neuronales evolutivas para el mismo sistema.

4.1 Parte Teórica

Los resultados obtenidos en la sección teórica representan los sistemas de control de tipo convencional como el PID y GMC, además de presentar los resultados de las evoluciones de la red neuronal a partir de la estrategia de evolución seleccionada.

4.1.1 Control a Lazo Abierto

La programación del modelo matemático en Simulink da como resultado un perfil de temperatura a lazo abierto en donde se puede demostrar la necesidad de control al sistema propuesto en este trabajo, se obtiene el perfil de temperatura (Figura 4.1)

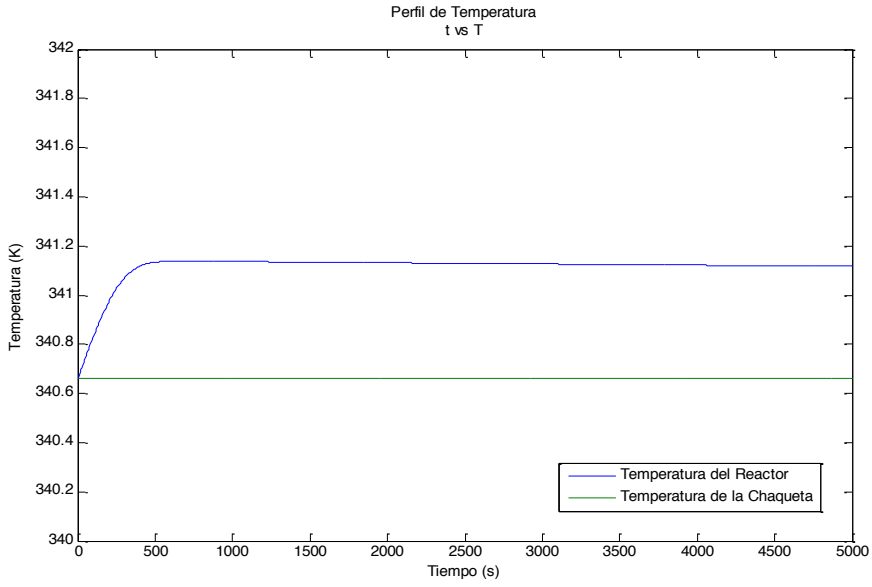


Figura 4.1 Perfil de Temperatura a Lazo Abierto

4.1.2 Control PID

El control de tipo PID (Proporcional, Integral y Derivativo) es un tipo de control altamente convencional que es utilizado en gran parte en el control de procesos químicos, pero este tipo de control presenta una desventaja en procesos de tipo no lineal o tipo batch, este tipo de control no es tan fácilmente ajustable a la dinámica del proceso además que puede presentar sobre tiros grandes con respecto a una respuesta rápida del control, para este caso de estudio del PMMA el control PID muestra el siguiente resultado Figura 4.2, cabe señalar que el control PID fue sintonizado por el método de Ziegler-Nicholson para las constantes para cada parte de controlador.

$$\frac{dx}{dt} = K\left(\left(x_{sp} - x\right) + \frac{1}{T_i}\int_0^t \left(x_{sp} - x\right)dt + T_d \frac{d\left(x_{sp} - x\right)}{dt}\right) \tag{3.76}$$

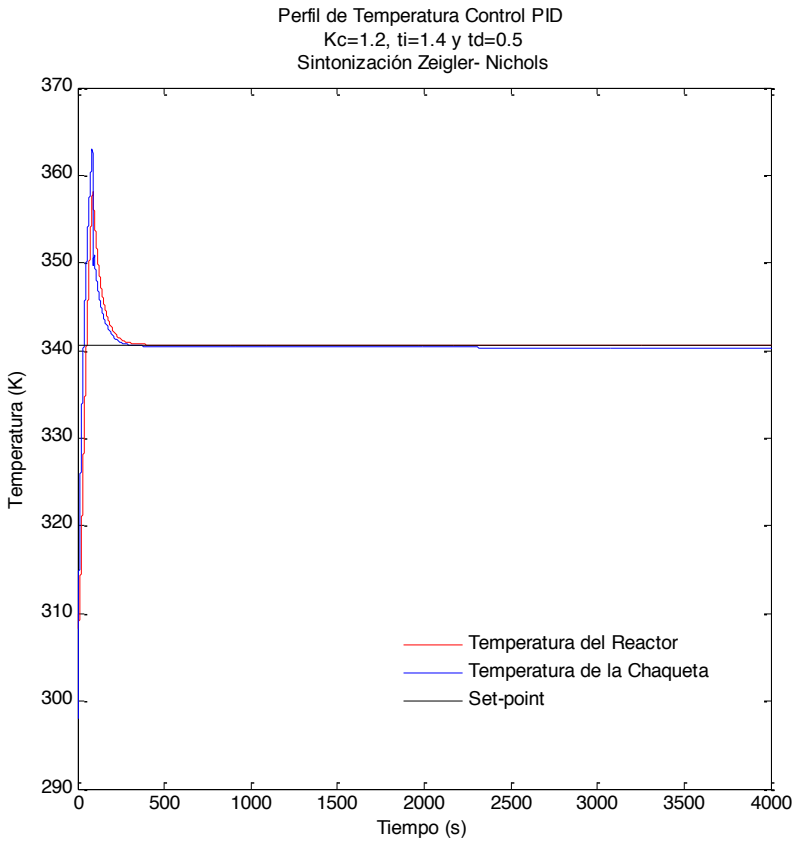


Figura 4.2 Respuesta de control PID

4.1.3 Control GMC

La implementación de este tipo de control a sistemas no lineales o de dinámica poco predecible, con lleva a la reformulación del control convencional tal reformulación se establece mediante el control de tipo no lineal conocido como control GMC (*Generic Model Control por sus siglas en inglés*), el cual para el proceso de polimerización muestra una respuesta aceptable.

Se busca el encontrar el perfil de temperatura mostrado en la Figura 4.3, pues para un mismo sistema de polimerización Ekpo Emmanuel y Mujtaba I.M, en el 2008 publicaron la implementación de un sistema hibrido de control basado en redes neuronales y un control de tipo GMC (*Generic Model Control por sus siglas en inglés*) dando como resultado el siguiente perfil (Ekpo, Mujtaba, 2008).Controlador GMC ley de control (Lee *y col*., 1988).

$$\frac{dx}{dt} = K_1\left(x_{sp} - x\right) + K_2\int_0^t\left(x_{sp} - x\right)dt \qquad (3.78)$$

Se comparan los tiempos de computo del control de referencia al simulado en el presente trabajo Tabla 4.1.

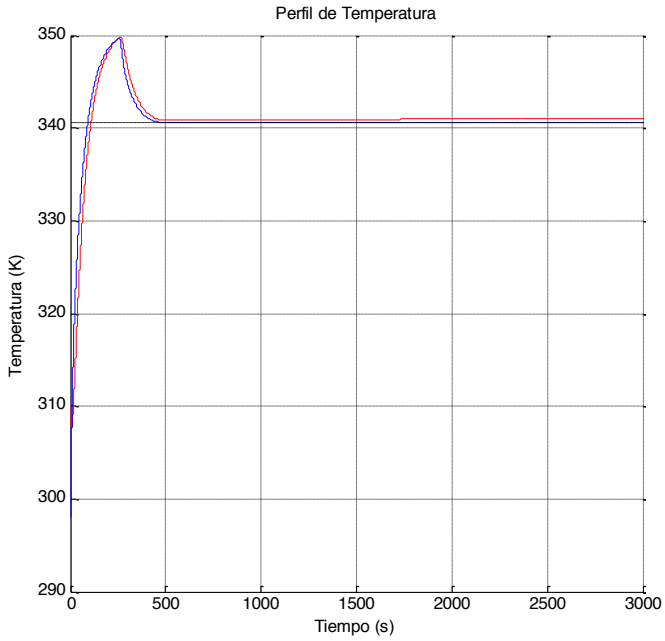


Figura 4.3 Perfil de Temperatura del sistema de Ekpo y Mujtaba Usando Matlab

Tabla 4.1 Comparación con el control de Referencia

Referencia	Temperatura (K)	Conversión (X_m)	Tiempo (s)
Ekpo-Mujtaba (*Neurocomputing 2008)*	340.66	60	2966.2
		90	16815.4
Este trabajo	340.66	60	2856.6
		90	16810.8

4.1.4 Control Neuronal

Se planteo un modelo de control neuronal basado en una red neuronal que consta de 4 neuronas en una sola capa oculta, con una sola salida en cada una de las neuronas presentes, como característica adicional una neurona de entrada y una neurona de salida dando un total de 6 neuronas en el sistema de control (Figura 4.4).

El modelo de entrenamiento es de tipo acelerado, para cual se debe hacer una comparación de los datos obtenidos de la planta así como con la implementación del control (Figura 4.5).

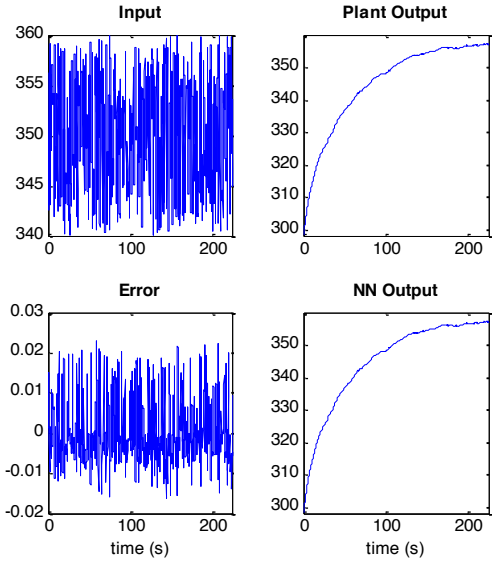


Figura 4.4 Perfiles de Entrenamiento

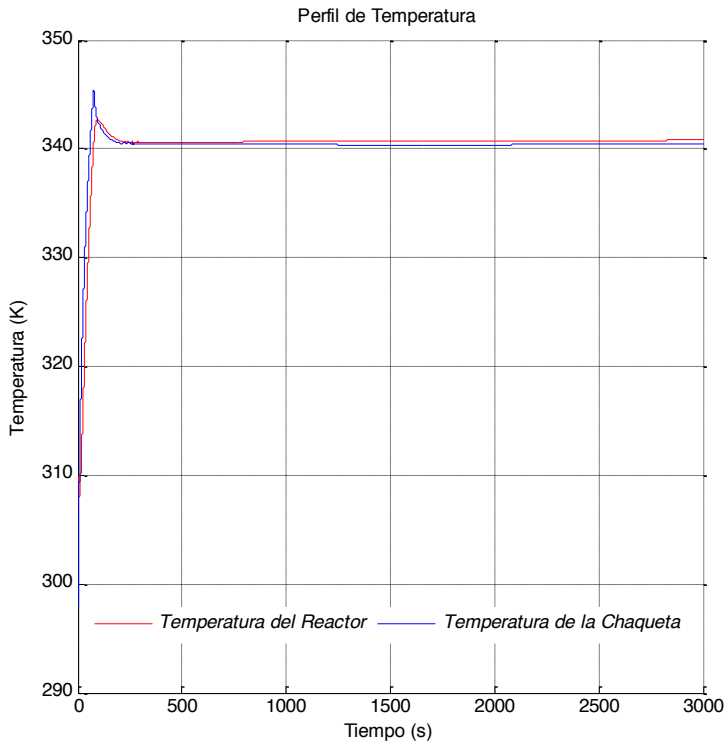


Figura 4.5 Pefil de Temperatura 6 neuronas totales

4.1.5 Control Neuronal Evolutivo

Diferentes referencias proponen como algoritmo de evolución a la función de excitación de la red neuronal únicamente con la variación en la pendiente de la función de excitación, generalmente la función de excitación más usada es la función tangencial de tipo sigmoidea la cual presenta dos umbrales en la pendiente (1, -1), donde la ecuación se representa como una razón de un exponencial (Ec. 3.80), en donde la pendiente se representa por φ y los valores de los pesos sinápticos se representar por w_i, Figura 4.6

$$f(w) = \frac{\exp(-\varphi \cdot w_i) + \varphi \exp(\varphi \cdot w_i)}{\exp(-\varphi \cdot w_i) - \exp(\varphi \cdot w_i)} \tag{3.80}$$

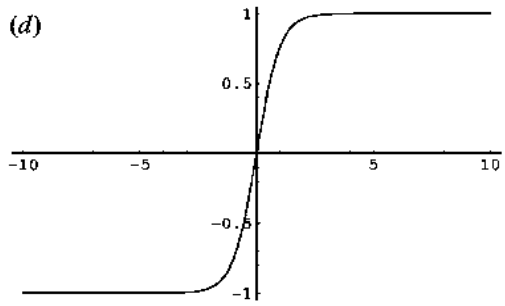


Figura 4.6 Función Tangencial de Tipo Sigmoidea

La modificación de la pendiente implicaría únicamente en la trayectoria que pueda seguir la evolución de la red, esto debido a que los umbrales de la función excitación permanecen de forma constante

Además del uso de la función tangencial otros autores proponen el uso de funciones de excitación de tipo logarítmica sigmoidea (Figura 4.7), donde dicha función es un valor inverso exponencial con dependencia de la pendiente (φ) y de los pesos sinápticos (w_i) (Ec. 3.81) y la función de tipo radial con sesgo (RBF) (Ec. 3.82)

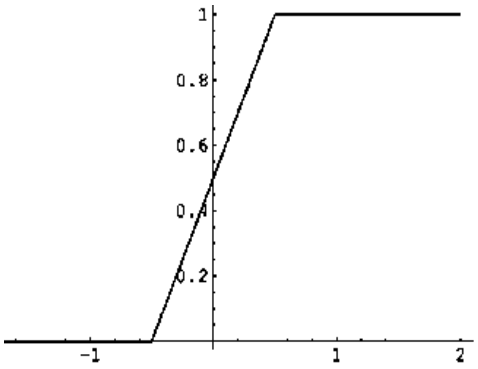


Figura 4.7 Función Logarítmica Sigmoidea

$$f(w) = \frac{1}{1 + \exp(-\varphi \cdot w_i)} \tag{3.81}$$

$$f(w) = \sum_{i=1}^{n} w_i \exp(\varphi \cdot w_i)^2 \left(\| w - w_{ci} \| \right) \tag{3.82}$$

Algoritmo de evolución Función Tangencial

Se plantea el entrenamiento para los diferentes valores de la pendiente de la función de excitación de tipo tangencial para la red neuronal cuya estructura se muestra en la Figura 4.8.

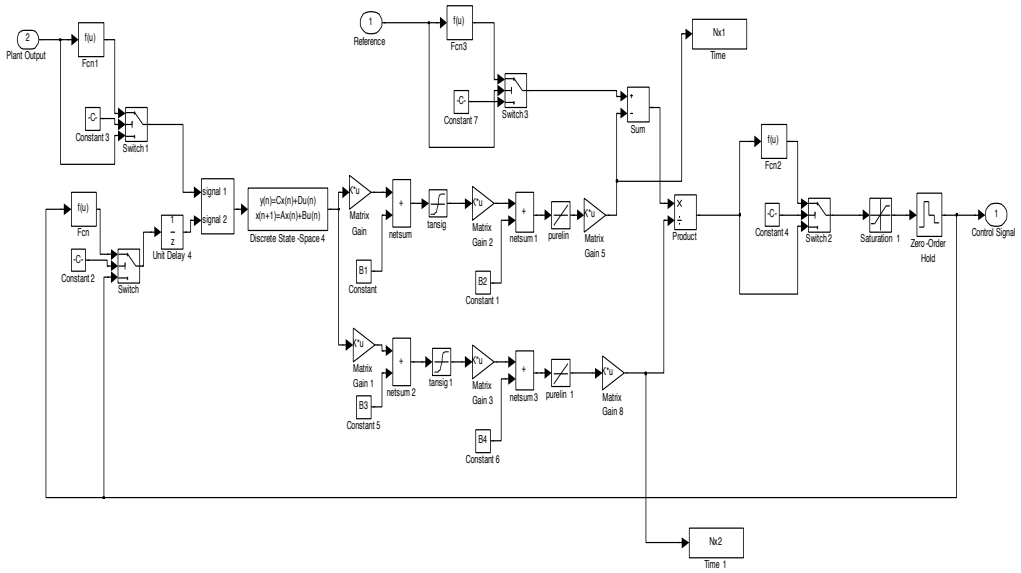


Figura 4.8 Estructura de la red neuronal con una función tangencial sigmoidea

Red neuronal evolutiva función tangencial pendiente 0.25

En las Figuras 4.9 - 4.10 se muestra la respuesta del control neuronal evolutivo con dos salidas, se observa que el control llega al set-point en un tiempo aproximado de 200 segundos, presentando un sobre tiro de temperatura hasta los 348 K, la tendencia de control muestra que existe mejora con respecto a otro tipo de control. De manera similar las Figuras 4.11-4.12 se muestra la respuesta del control neuronal evolutivo con tres salidas, se observa que el control llega a estabilizarse en un tiempo aproximado de 150 segundos siendo más rápido con respecto al control con una red neuronal de dos salidas, además, presentando un sobre tiro de temperatura hasta los 344 K, menor con respecto al control de dos salidas los cual se presenta la mejora en el control debido a la evolución misma del sistema. Las Figuras 4.13-4.14 se muestra la respuesta del control neuronal evolutivo con cuatro salidas, se observa que el control llega al control en un tiempo aproximado de 110 segundos siendo más rápido con respecto al control con una red neuronal de tres salidas, además, presentando un sobre tiro de temperatura hasta los 345 K,

semejante con respecto al control de dos salidas observándose la evolución del sistema.

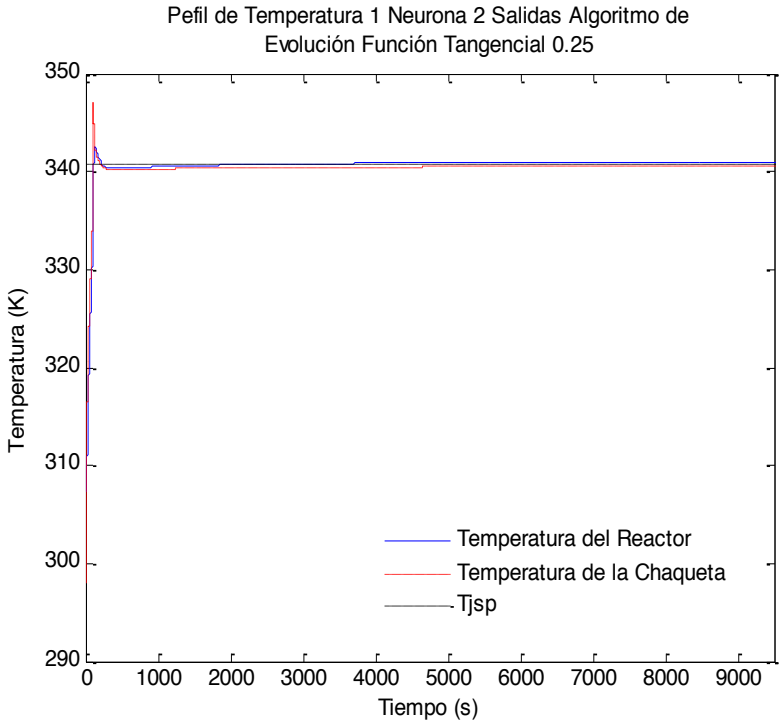


Figura 4.9 Perfil de Temperatura pendiente 0.25 dos salidas

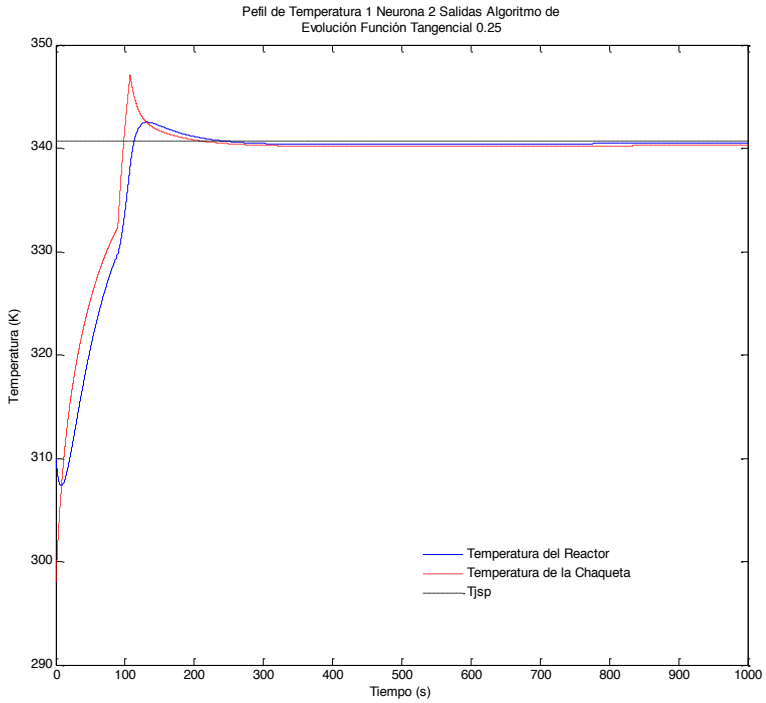


Figura 4.10 Perfil de Temperatura (Ampliación), pendiente 0.25 ,dos salidas

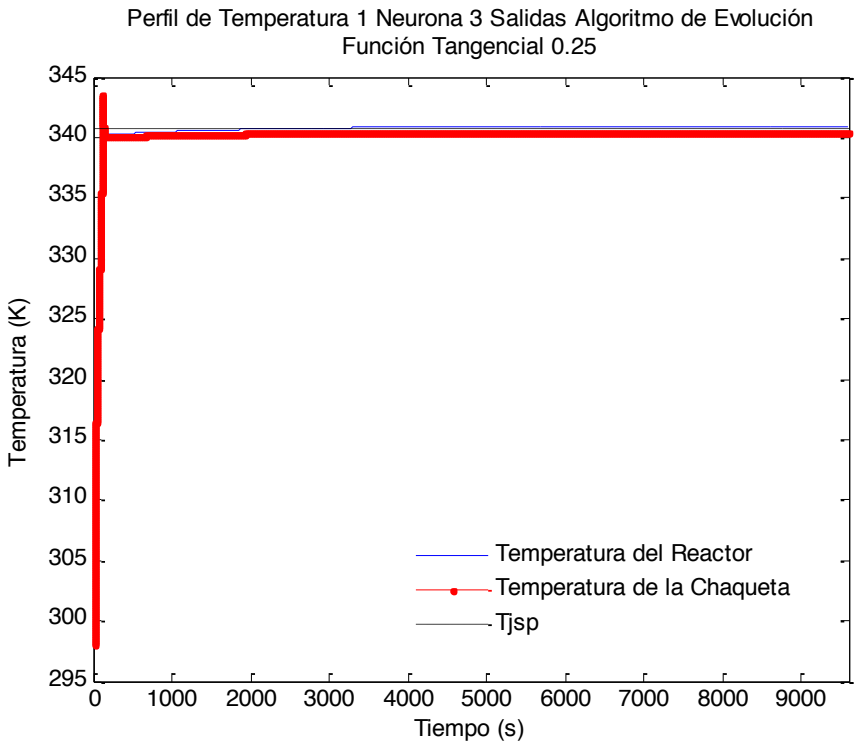


Figura 4.11 Perfil de Temperatura con tres salidas y una pendiente de 0.25

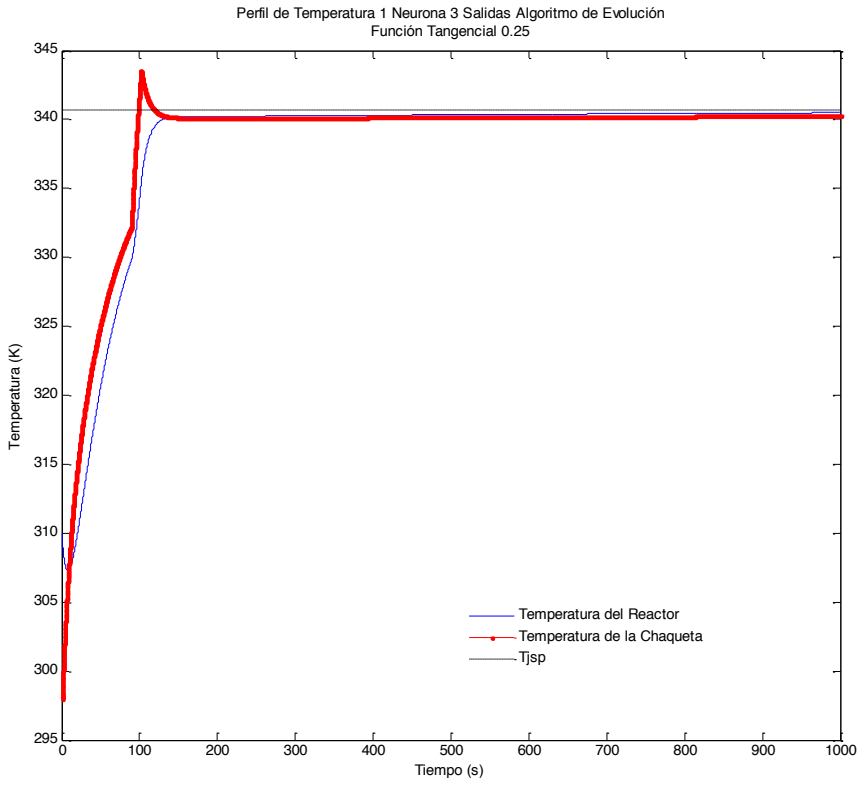


Figura 4.12 Perfil de Temperatura (Ampliación), pendiente 0.25, cuatro salidas.

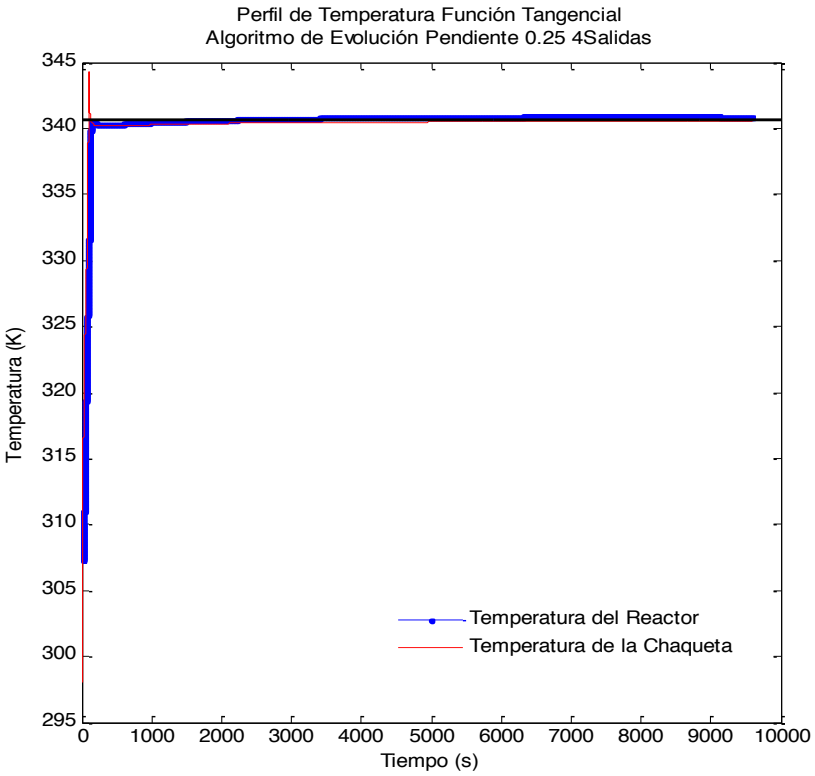


Figura 4.13 Perfil de Temperatura con cuatro salidas y una pendiente de 0.25

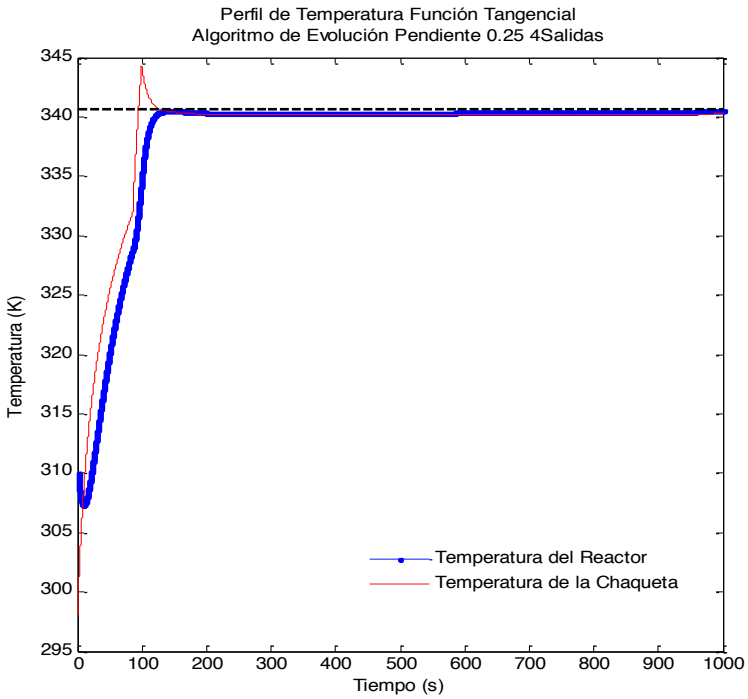


Figura 4.14 Perfil de Temperatura (ampliación) con cuatro salidas y una pendiente de 0.25

Red neuronal evolutiva función tangencial pendiente 0.5

La respuesta de control para una pendiente de 0.5 y modificando el número de salidas de la red neuronal, se muestra en las Figuras 4.15-4.16 se muestra la respuesta del control neuronal evolutivo con dos salidas, el control alcanza el set-point en un tiempo aproximado de 210 segundos, además, presenta un sobre tiro de temperatura hasta los 351 K, es claro que al presentarse inconsistencias en el perfil de temperatura se puede deber al número de estados de evolución que presenta la red, provocando de esta manera que el sistema no pueda presentar un ajuste dinámico. En las Figuras 4.17 no presenta control de la variable, esto se puede sustentar en el número de neuronas presentes en sistema de control, tal es que las neuronas evolucionadas presentan un sobre aprendizaje, además de no provocar un ajuste a la dinámica del proceso. En las Figuras 4.18-4.19 se muestra la respuesta del control neuronal evolutivo con cuatro salidas, se controla en un tiempo aproximado de 100 segundos, además, presenta un sobre tiro de temperatura hasta los 344 K, el cual es menor con referencia al control con solo dos salidas y la misma pendiente

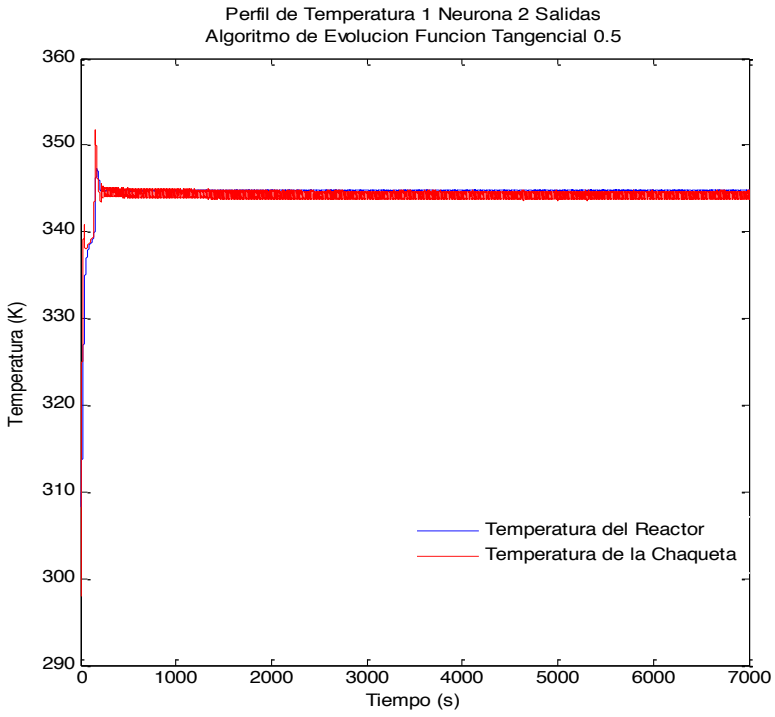


Figura 4.15 Perfil de Temperatura con dos salidas y una pendiente de 0.5

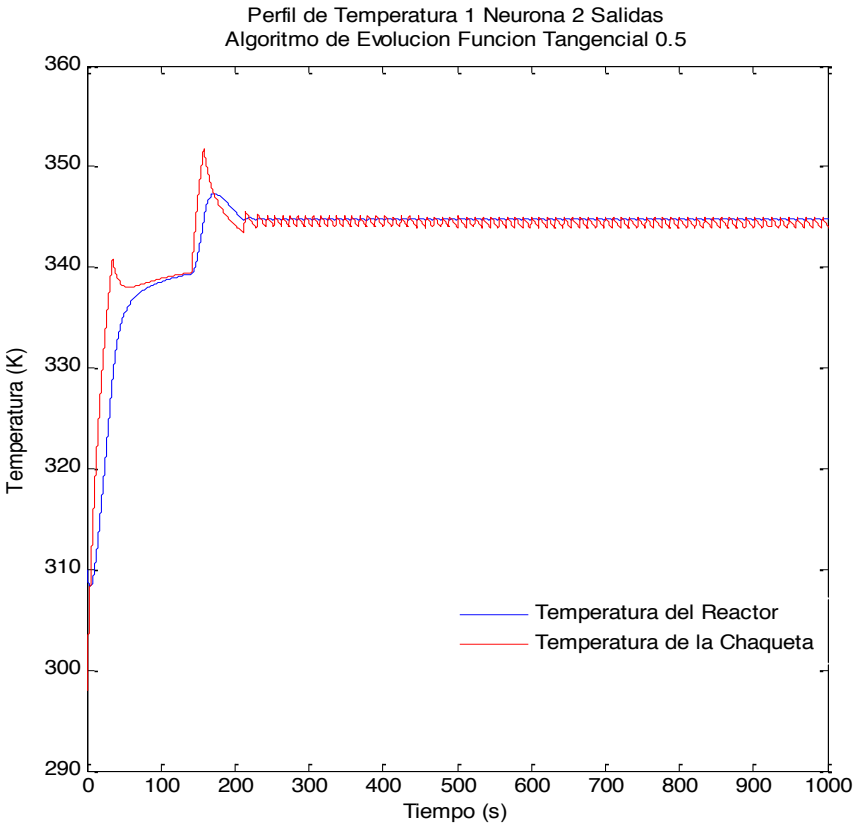


Figura 4.16 Perfil de Temperatura (ampliación) con dos salidas y una pendiente de 0.5

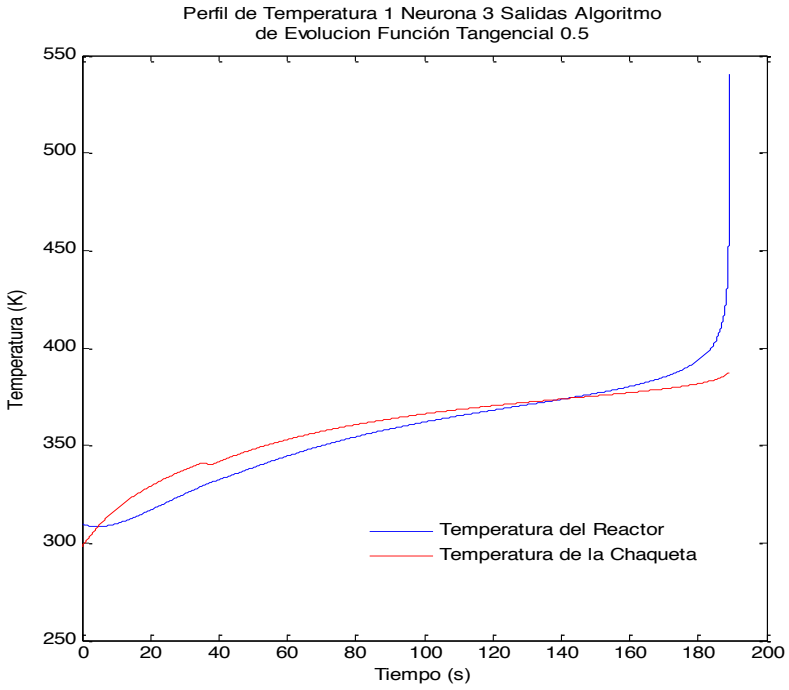


Figura 4.17 Perfil de Temperatura con tres salidas y una pendiente de 0.5

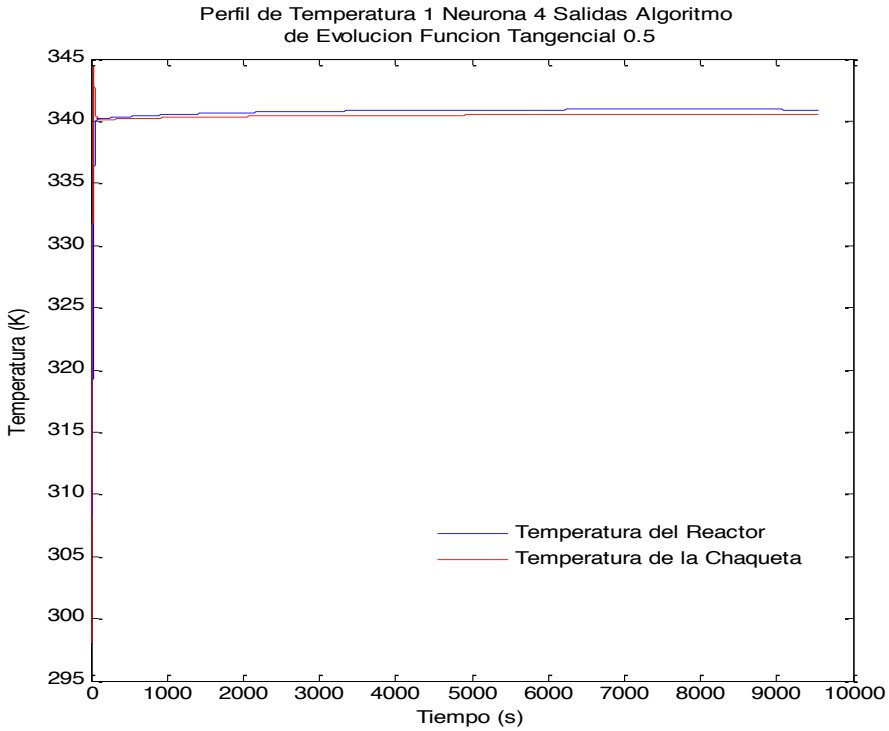


Figura 4.18 Perfil de Temperatura con cuatro salidas y una pendiente de 0.5

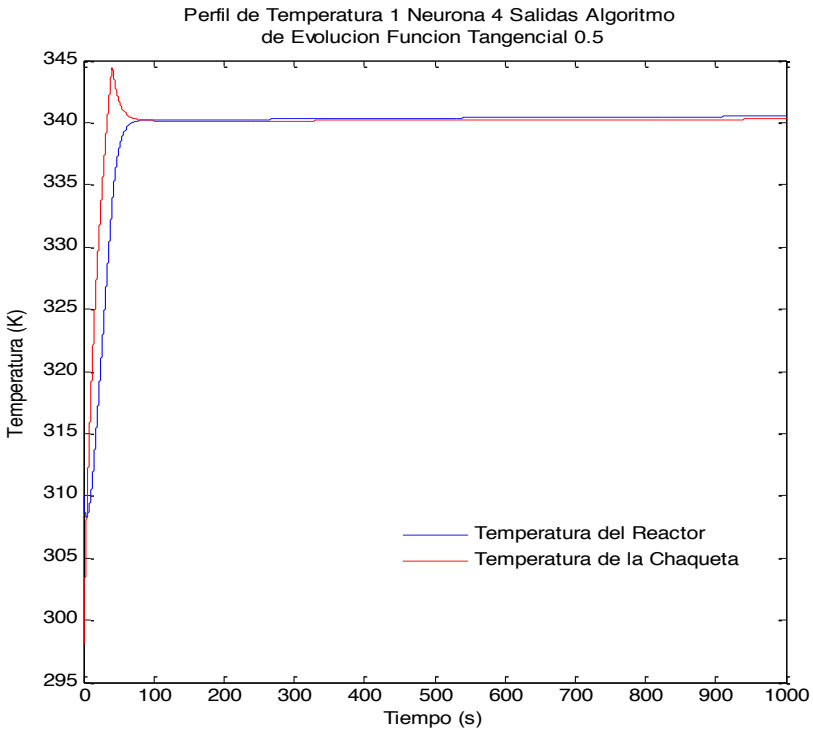


Figura 4.19 Perfil de Temperatura (ampliación) con cuatro salidas y una pendiente de 0.5

Red neuronal evolutiva función tangencial pendiente 1.0

En las Figuras 4.20-4.25 se muestra la respuesta del control neuronal evolutivo con dos, tres y cuatro salidas con una pendiente de 1.0, para cada uno de los casos se demuestra que la respuesta del control neuronal evolutivo es mejor con respecto a las respuestas de los sistemas de control convencional, la única discrepancia que existe entre las respuesta de control es la velocidad de respuesta del mismo, se puede ver en la figuras que conforme aumenta el número de salidas la respuesta de control se ve incrementada, además, que se presentan también la disminución del sobre-tiro que presenta el control neuronal evolutivo.

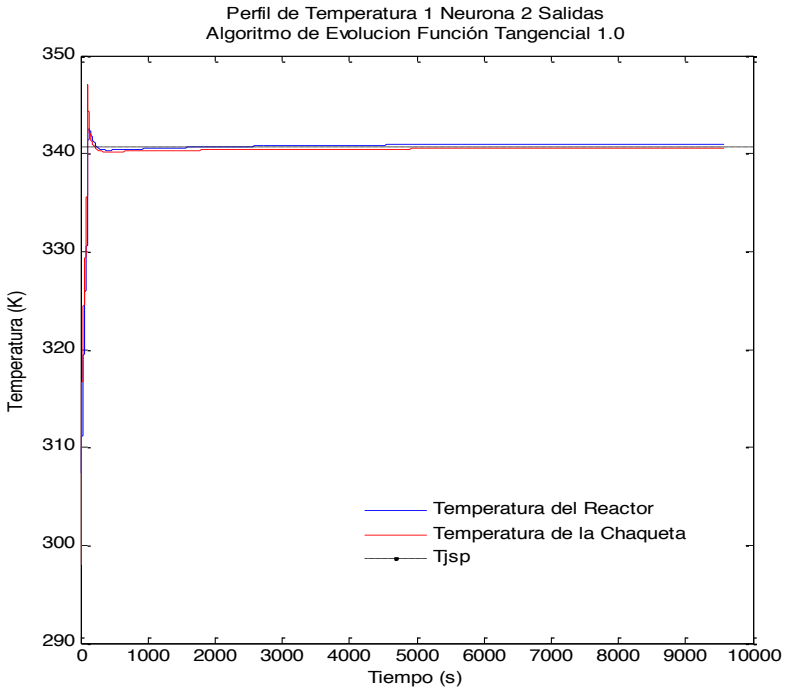


Figura 4.20 Perfil de Temperatura con dos salidas y una pendiente de 1.0

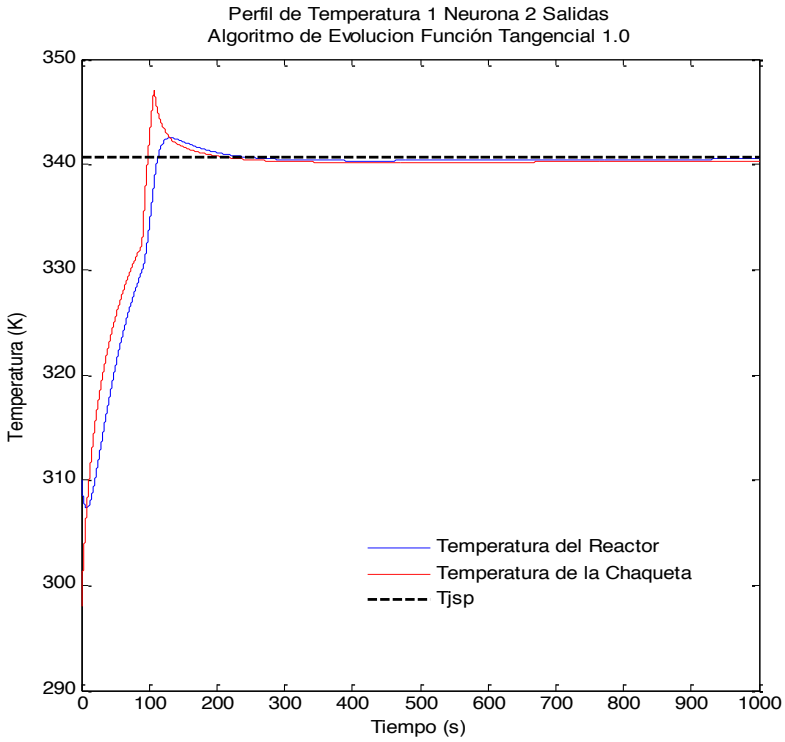


Figura 4.21 Perfil de Temperatura (ampliación) con dos salidas y una pendiente de 1.0

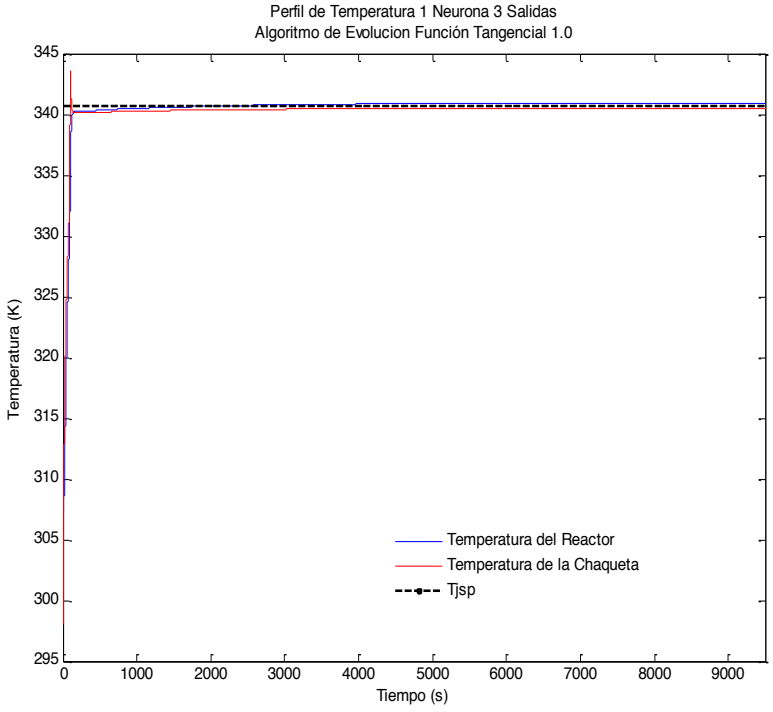


Figura 4.22 Perfil de Temperatura con tres salidas y una pendiente de 1.0

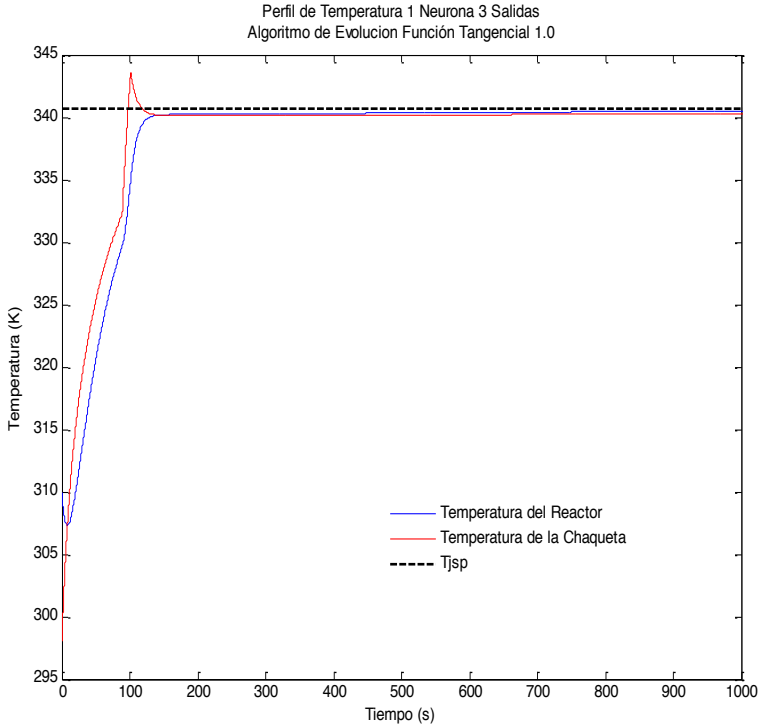


Figura 4.23 Perfil de Temperatura (ampliación) con tres salidas y una pendiente de 1.0

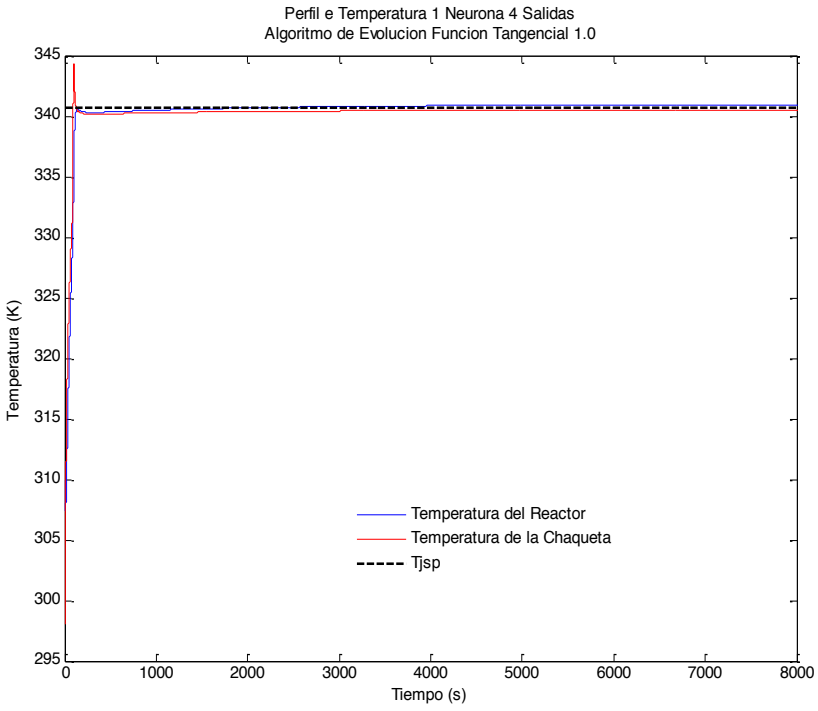


Figura 4.24 Perfil de Temperatura con cuatro salidas y una pendiente de 1.0

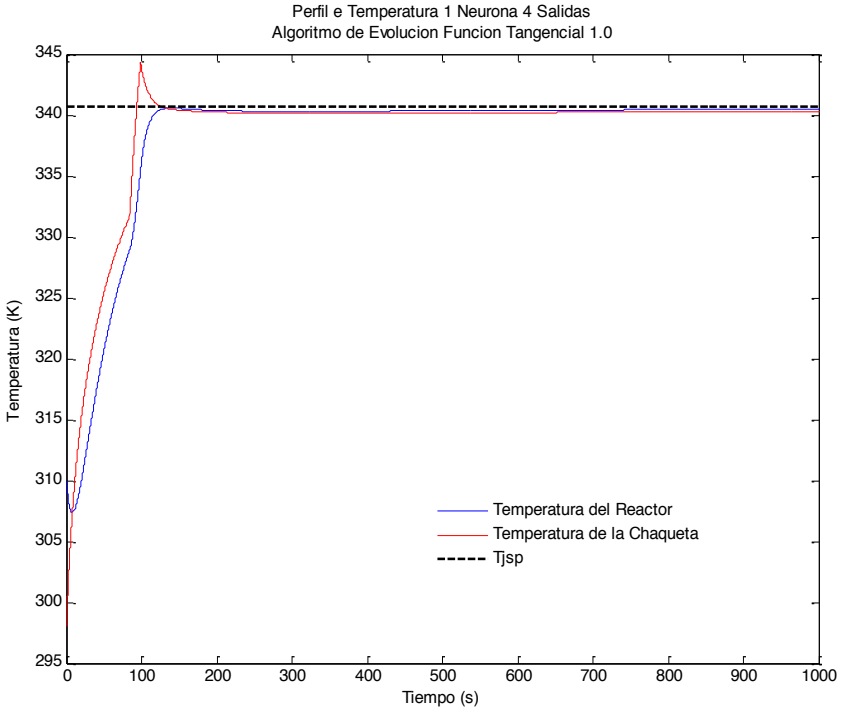


Figura 4.25 Perfil de Temperatura (ampliación) con cuatro salidas y una pendiente de 1.0

Red neuronal evolutiva, función tangencial, pendiente 1.5

En las Figuras 4.26-4.27 se muestra la respuesta del control neuronal evolutivo con dos salidas y una pendiente de 1.5, se observa que el control llega al set-point en un tiempo aproximado de 200 segundos, además, presenta un sobre tiro de temperatura hasta los 348 K. En la Figura 4.28 se muestra la respuesta del control neuronal evolutivo con dos salidas y una pendiente de 1.5, se observa que el control no llega al set-point volviéndose incontrolable el sistema. En las Figuras 4.29-4.30 se muestra la respuesta del control neuronal evolutivo con dos salidas y una pendiente de 1.5, se observa que el control llega al set-point en un tiempo aproximado de 70 segundos, además, presenta un sobre tiro de temperatura hasta los 344 K, lo cual indica quela respuesta del controlador es mejor con respecto a un sistema con dos salidas y la misma pendiente, comprobándose la evolución del sistema neuronal.

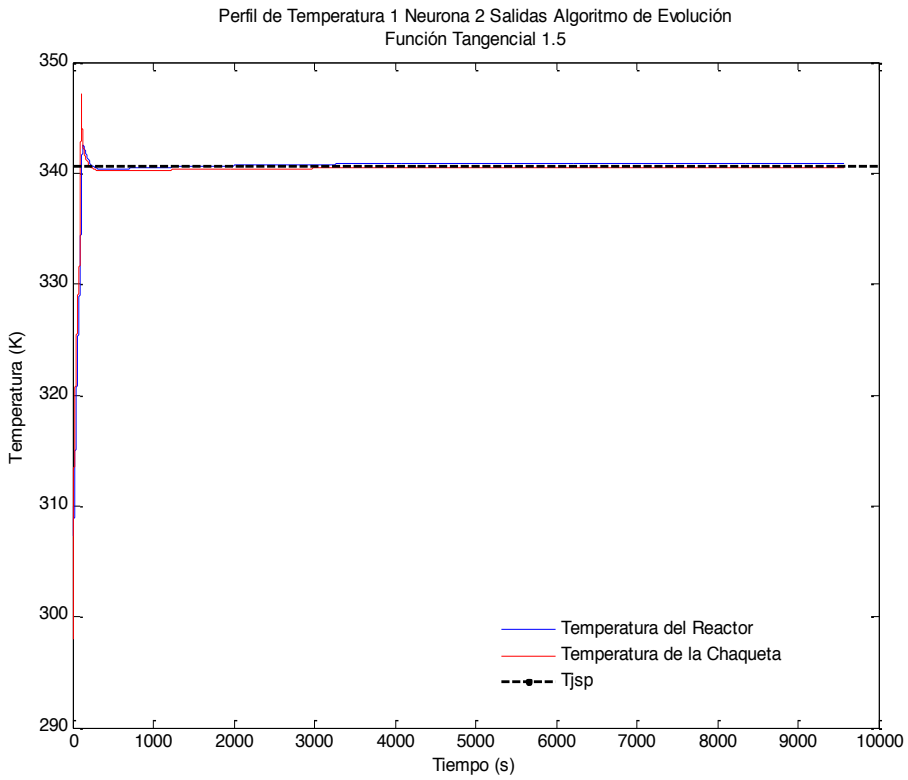


Figura 4.26 Perfil de Temperatura con dos salidas y una pendiente de 1.5

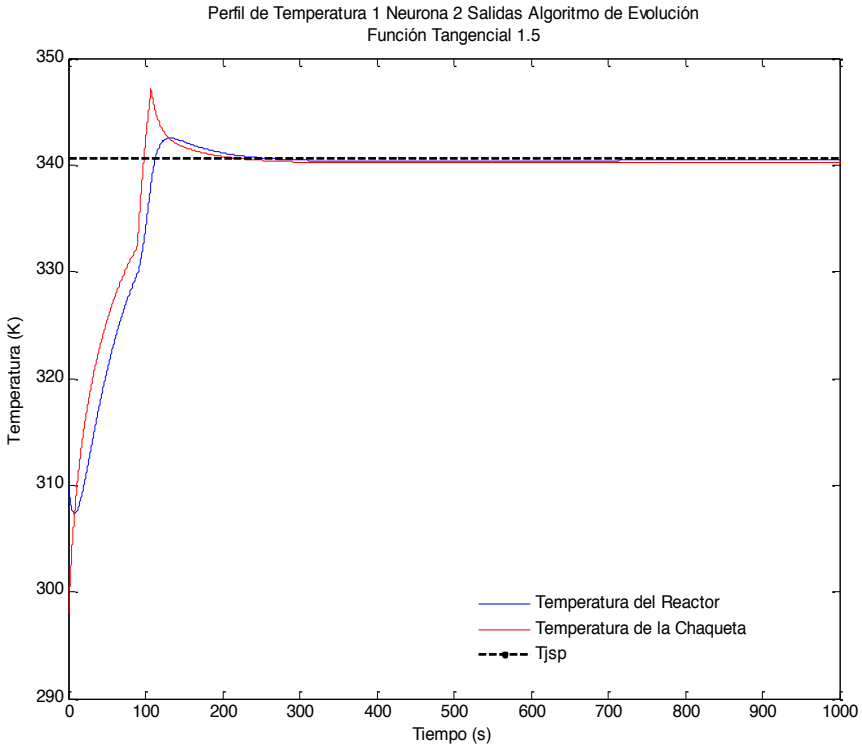


Figura 4.27 Perfil de Temperatura (ampliación) con dos salidas y una pendiente de 1.5

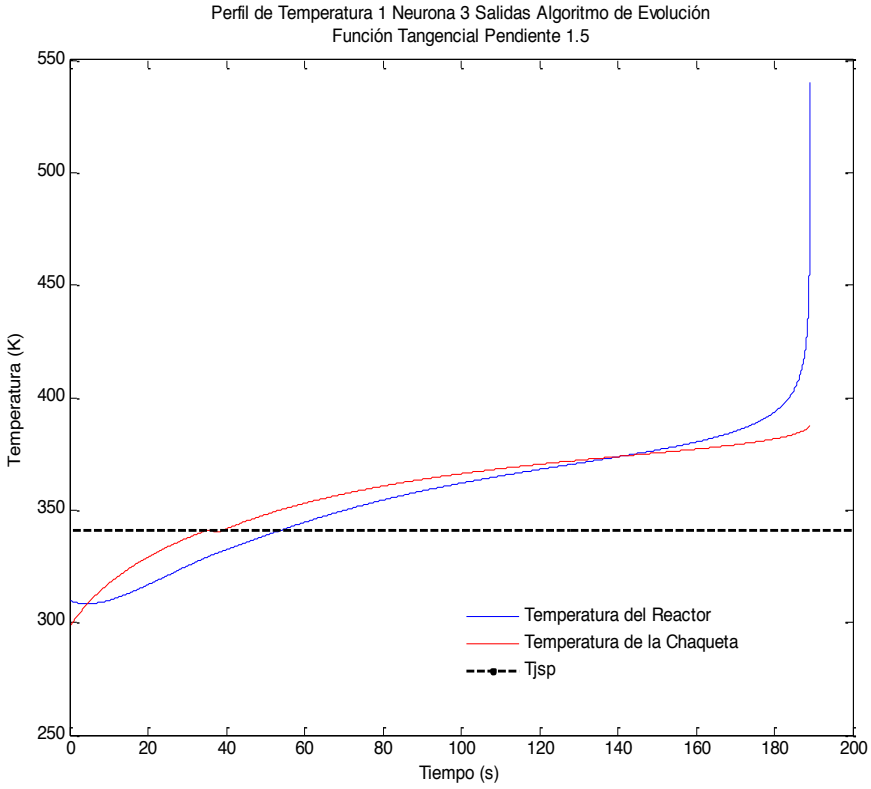


Figura 4.28 Perfil de Temperatura con tres salidas y una pendiente de 1.5

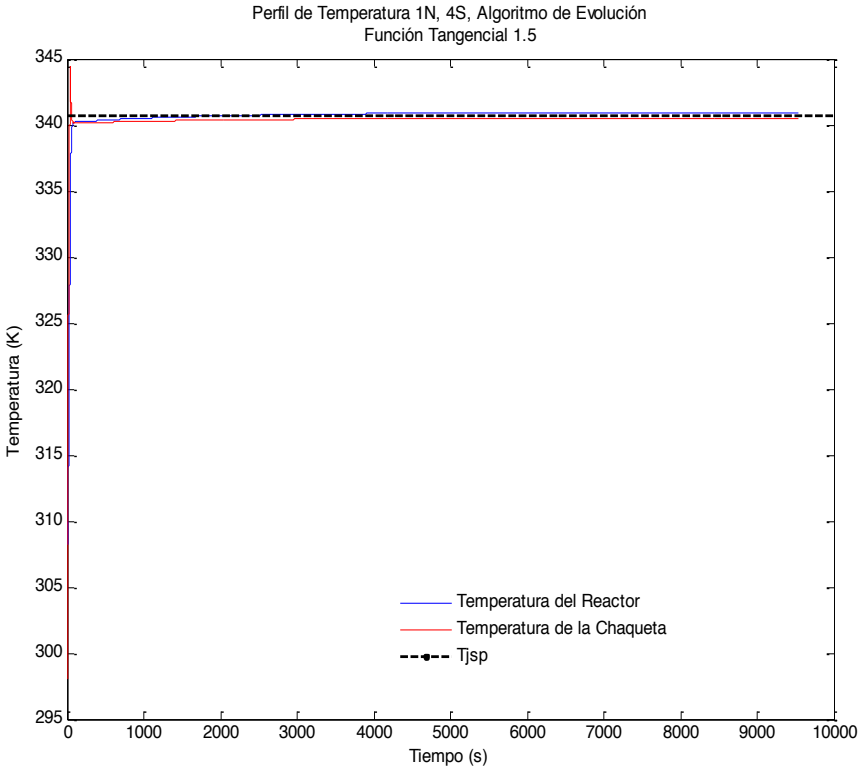


Figura 4.29 Perfil de Temperatura con cuatro salidas y una pendiente de 1.5

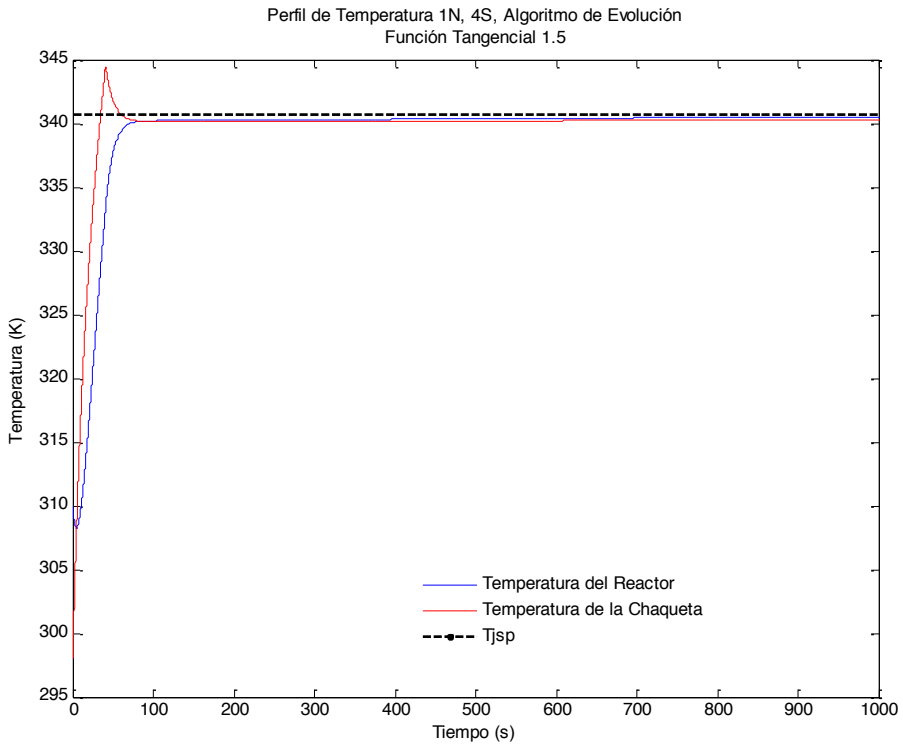


Figura 4.30 Perfil de Temperatura (ampliación) con cuatro salidas y una pendiente de 1.5

Red neuronal evolutiva, función tangencial, pendiente de 2.0

En las Figuras 4.31-4.32 se muestra la respuesta del control neuronal evolutivo con dos salidas y una pendiente de 2.0, se observa que el control llega al set-point en un tiempo aproximado de 190 segundos, además, presenta un sobre tiro de temperatura hasta los 348 K. Las Figuras 4.33-4.34, el control se estabiliza en un tiempo aproximado de 110 segundos, además, presenta un sobre tiro de temperatura hasta los 344 K, lo cual indica respuesta del controlador es mejor con respecto a un sistema con dos salidas y la misma pendiente, comprobándose la evolución del sistema neuronal. En la Figura 4.35 se muestra que el sistema neuronal evolutivo no logra controlar a la variable.

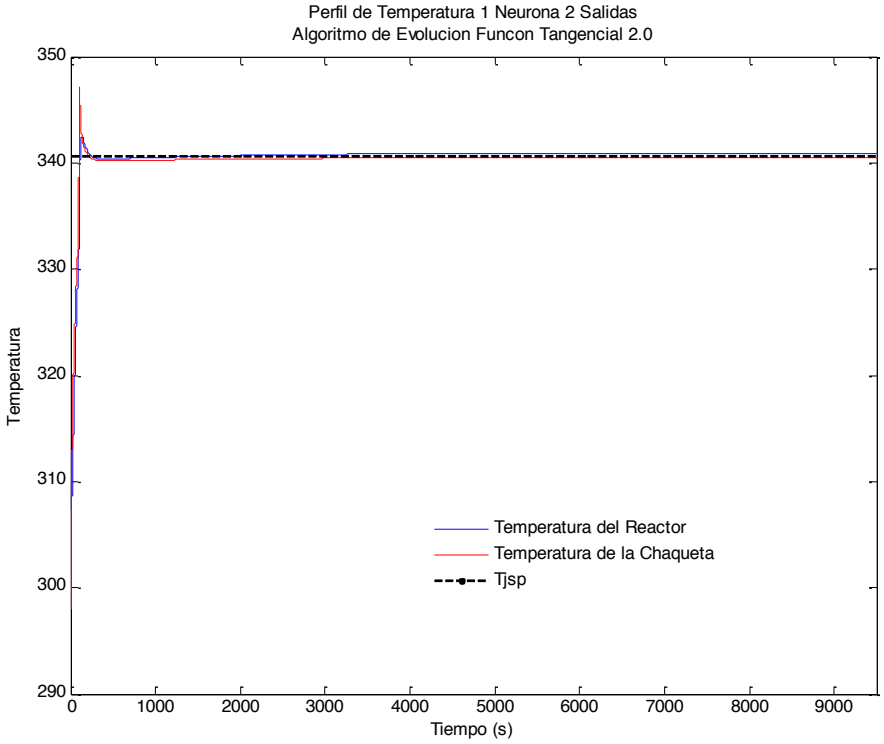


Figura 4.31 Perfil de Temperatura con dos salidas y una pendiente de 2.0

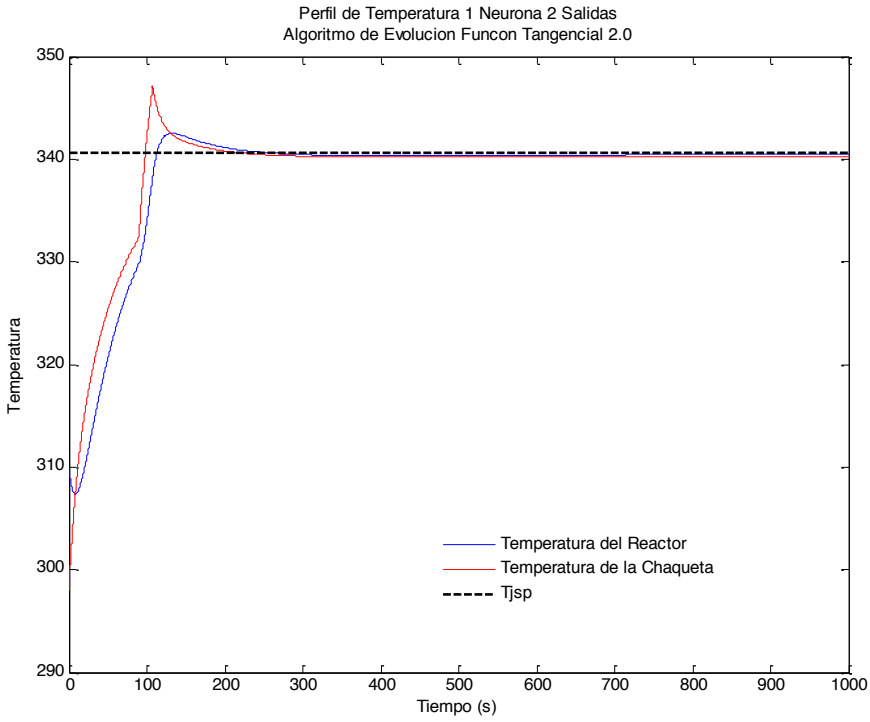


Figura 4.32 Perfil de Temperatura (ampliación) con dos salidas y una pendiente de 2.0

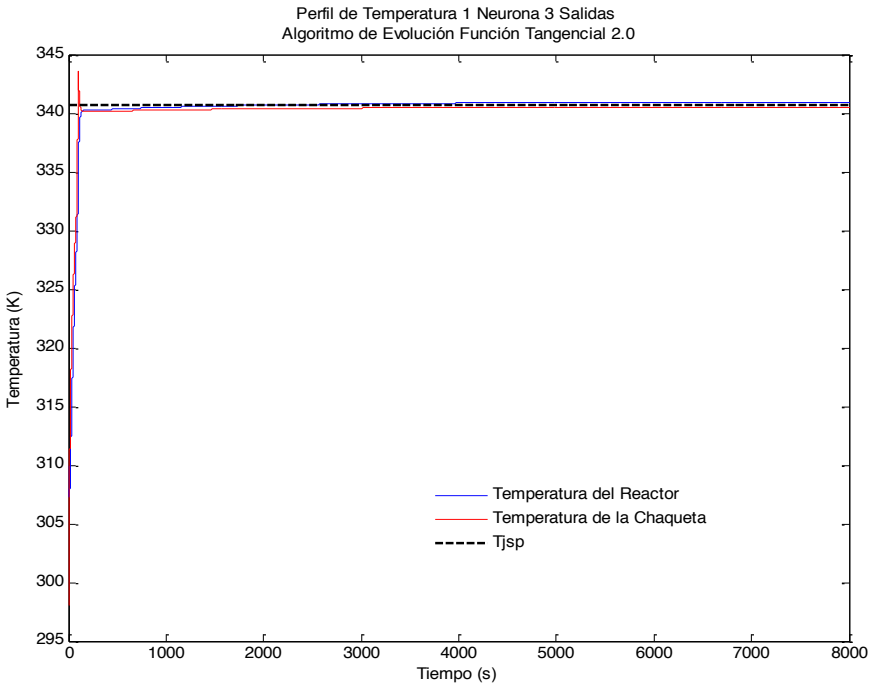


Figura 4.33 Perfil de Temperatura con tres salidas y una pendiente de 2.0

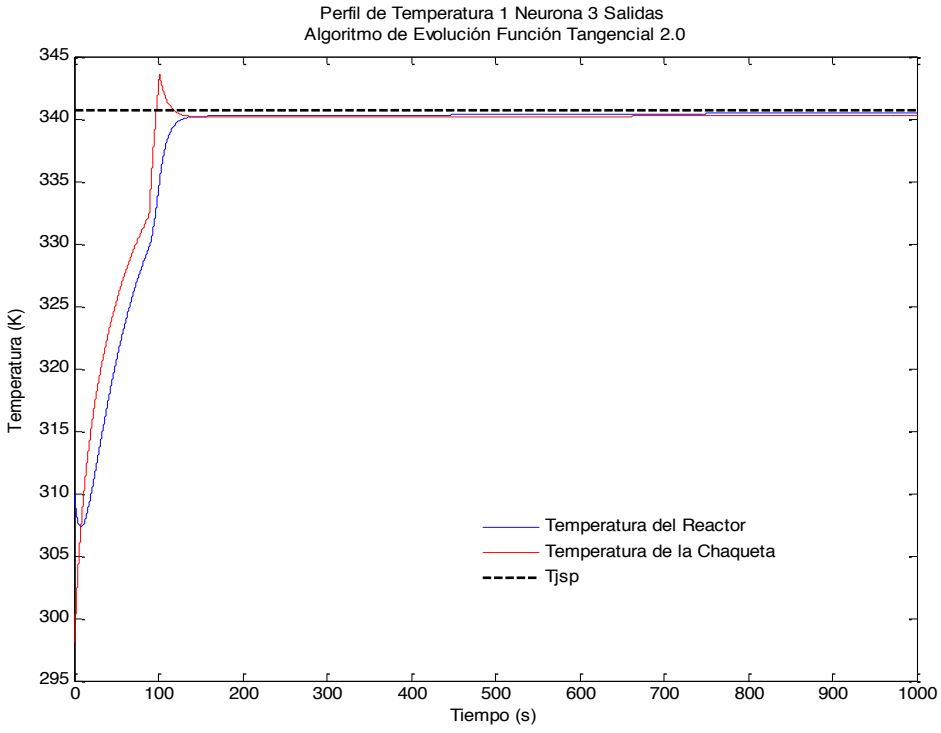


Figura 4.34 Perfil de Temperatura (ampliación) con tres salidas y una pendiente de 2.0

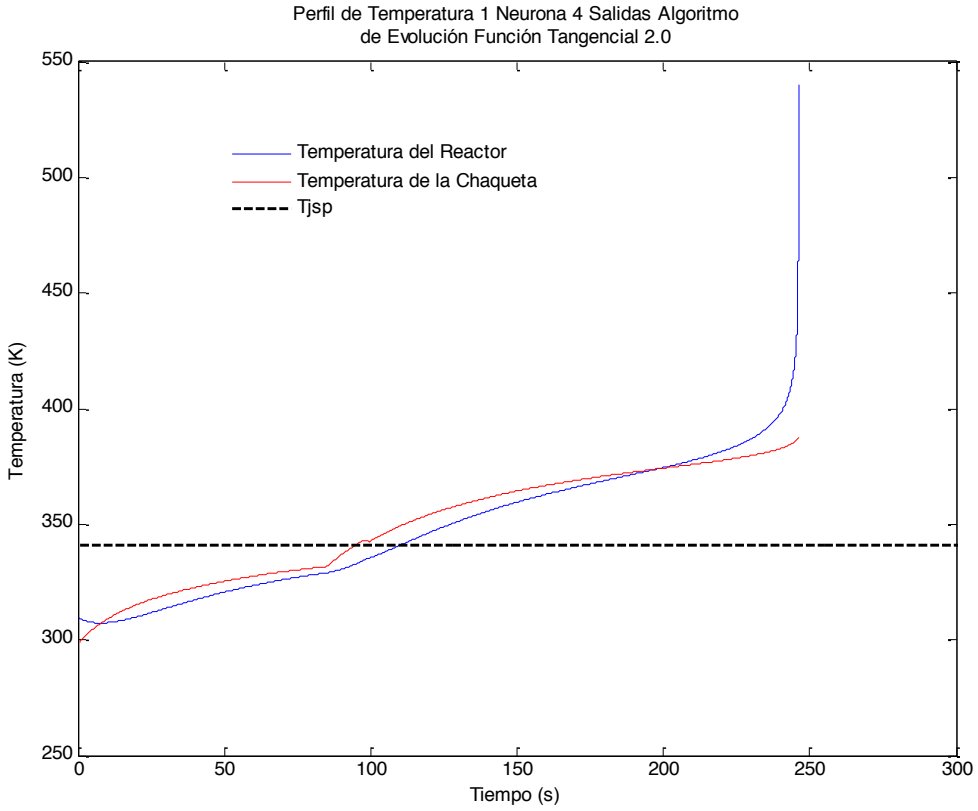


Figura 4.35 Perfil de Temperatura con cuatro salidas y una pendiente de 2.0

Algoritmo de Evolución Función Logarítmica

El algoritmo de evolución basado en la función de excitación puede proporcionar resultados similares que los proporcionados por un algoritmo de evolución similar, las arquitecturas que se establecen en este trabajo se mantienen como constantes durante el periodo de evolución en los cuales se observa durante los entrenamientos se modifican las cantidades de épocas o generaciones de neuronas evolucionadas, esto mediante únicamente la modificación de la pendiente y del número de salidas de la neurona, como se planteo para la función tangencial se realiza el mismo procedimiento para la función de tipo logarítmica modificando la pendiente entre un intervalo de 0.25-2.0 y el número de salidas de 2-4 salidas y una arquitectura constante de una neurona de entrada, una neurona en la capa oculta y una neurona de salida con propagación de información generando una red neuronal dinámica.

En la Figura 4.36 se muestra la respuesta del control una red neuronal con arquitectura constante con una pendiente de 0.25 y una cantidad de salida de las

neuronas de dos, la respuesta representa que el sistema no tiene control dentro del punto base (set-point) esto para obtener la conversión óptima de la reacción.

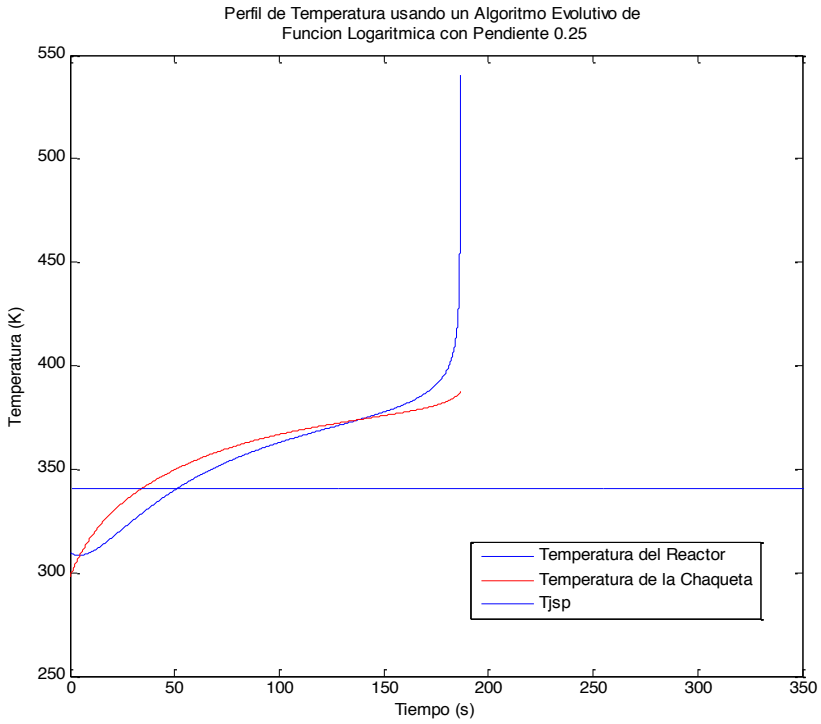


Figura 4.36 Perfil de Temperatura Función Logaítmica pendiente 0.25

Para el sistema de control basado en la arquitectura constante de una neurona de entrada, una neurona en capa oculta y una neurona de salida con un número de salida de dos pero con una pendiente de 0.5 en la función logarítmica, la Figura 4.37 muestra que el control se presenta 15 grados por encima del set-point, pero no se presenta descontrol de la temperatura en el reactor, lo que puede significar que el entrenamiento de la red neuronal establecido surgió efecto pero no dentro de lo establecido.

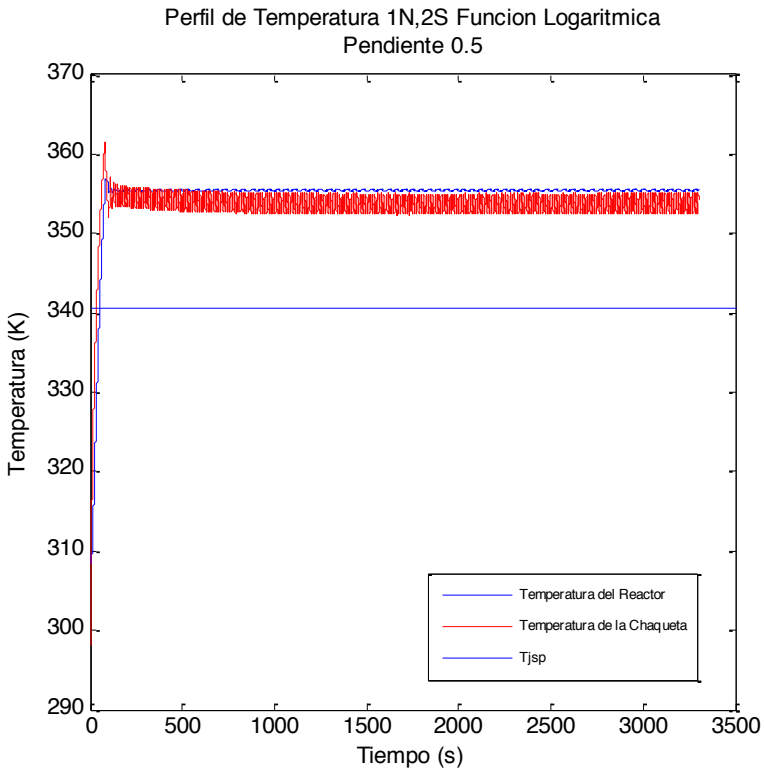


Figura 4.37 Perfil de Temperatura Función Logarítmica pendiente 0.5

La modificacion de la pendiente de 0.5 a 1.0 con la misma arquitectura muestra que el perfil de temperatura se vuelve incontrolable tanto para la temperatura del reactor, como para la temperatura de la chaqueta, esto significa que tanto para una pendiente de 0.5 a 1.0 el sistema de control neuronal evolutivo no funciona de forma adecuada (Figura 4.38).

La respuesta del control para modificación de la pendiente de 1.0 a 1.5 se presenta en la Figura 4.39, la respuesta es similar para pendientes de 1.0 y 0.25 lo que no llega al objetivo de control.

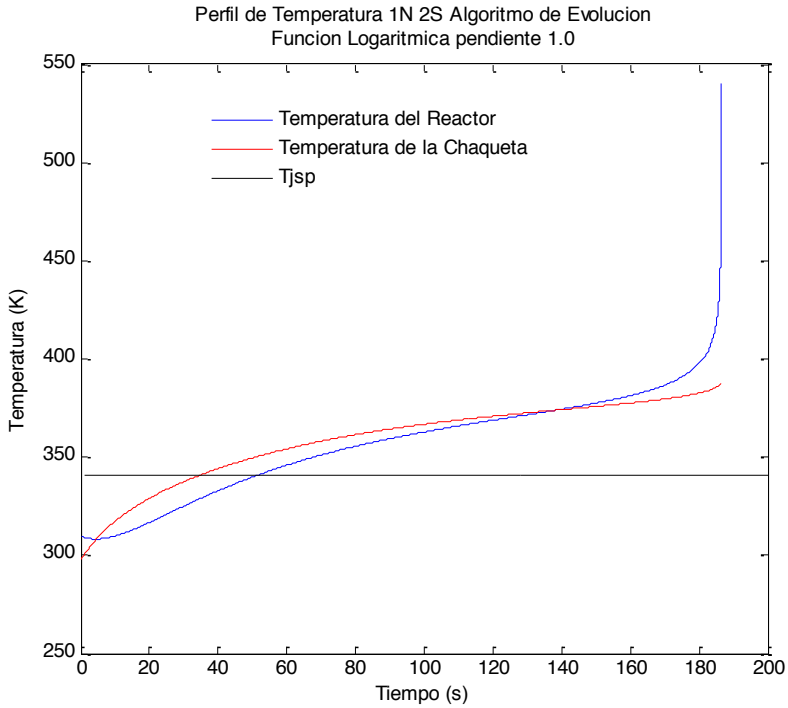


Figura 4.38 Perfil de Temperatura Función Logarítmica pendiente 1.0

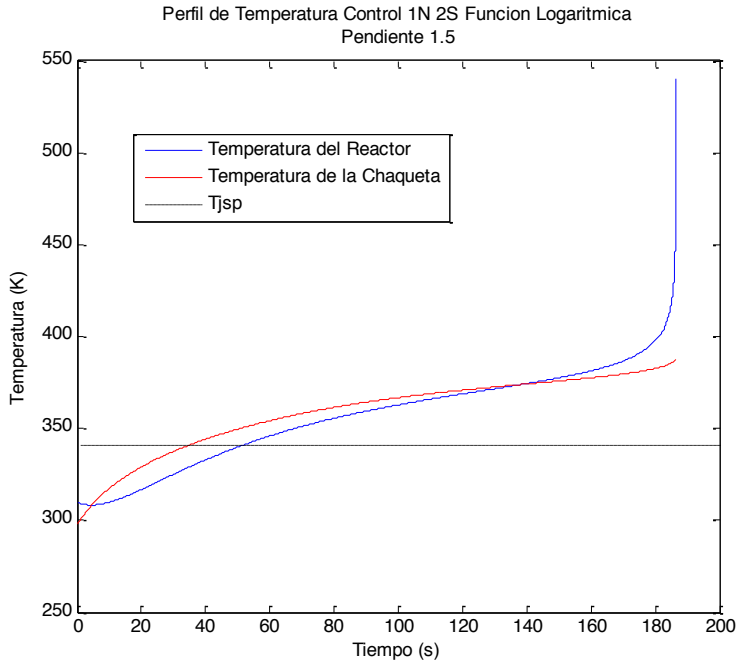


Figura 4.39 Perfil de Temperatura Función Logarítmica pendiente 1.5

Para cuando la pendiente cambia de 1.5 a 2.0 con la arquitectura constante, la respuesta del control neuronal evolutivo es satisifactoria con respecto al perfil de temperatura (Figura 4.40), alcanzando el set-point con un sobre tiro de relativamente disminuido (Figura 4.41).

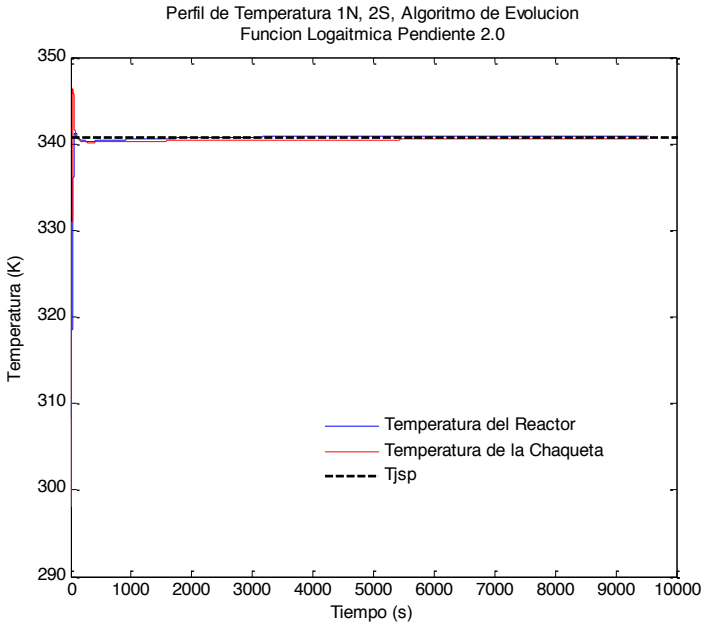


Figura 4.40 Perfil de Temperatura Función Logarítmica pendiente 2.0

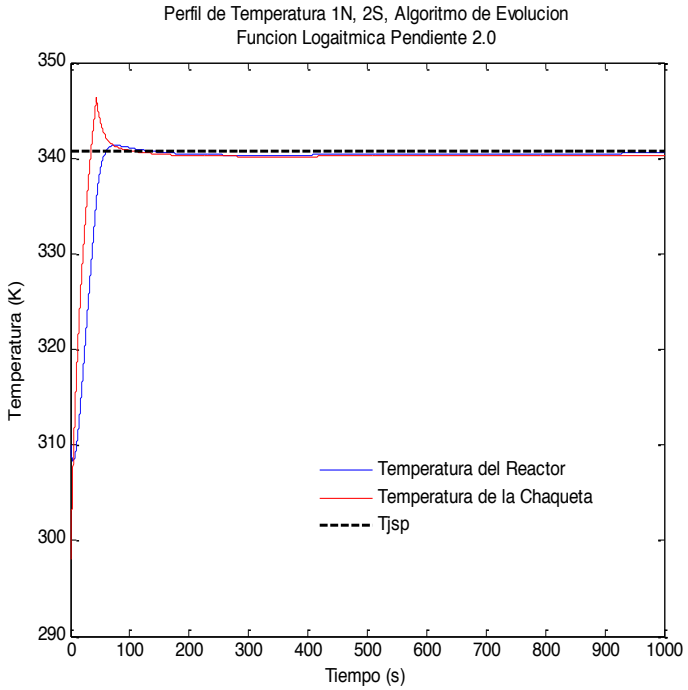


Figura 4.41 Perfil de Temperatura Función Logarítmica pendiente 2.0 (amp)

En comparación con la arquitectura anterior con dos salidas, la arquitectura con tres salidas y con diferentes pendientes pueden demostrar la evolución de la red neuronal (Figura 4.42-4.47), solo se presentó un incremento de la temperatura tanto para la chaqueta, como para el reactor, para una red neuronal con una pendiente de 1.5, alcanzando el control 15 grados por encima del set-point, lo cual no indica que sea un buen control (Figura 4.48).

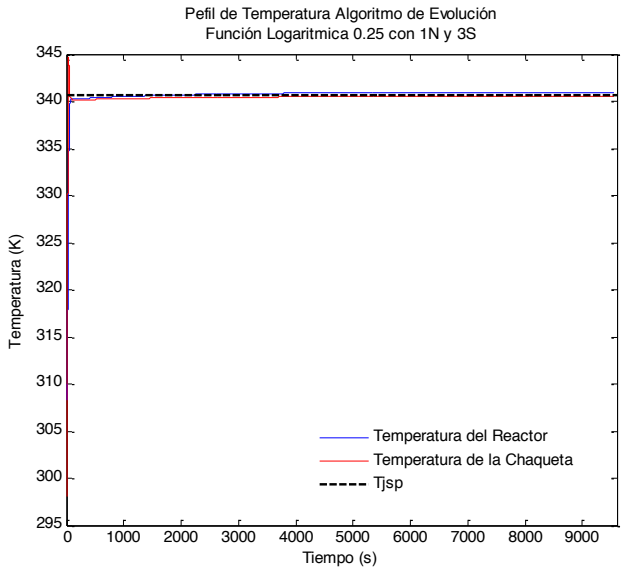


Figura 4.42 Perfil de Temperatura Función Logarítmica, pendiente 0.25, tres salidas

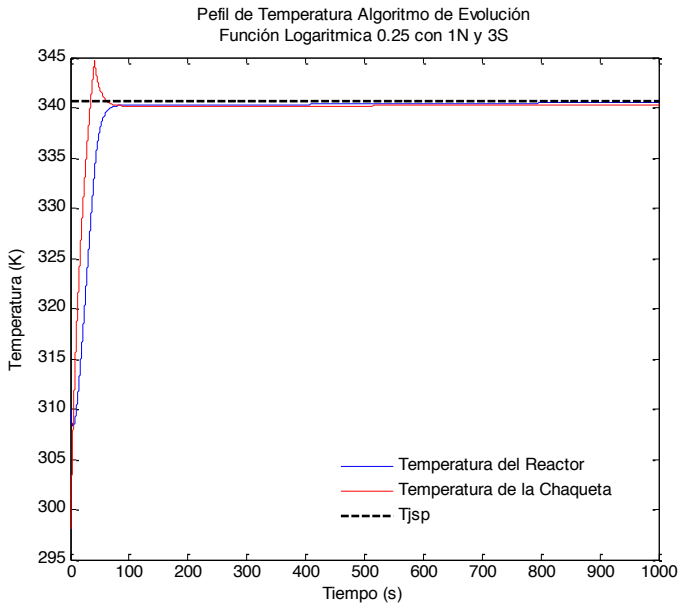


Figura 4.43 Perfil de Temperatura Función Logarítmica, pendiente 0.25, tres salidas (ampliación)

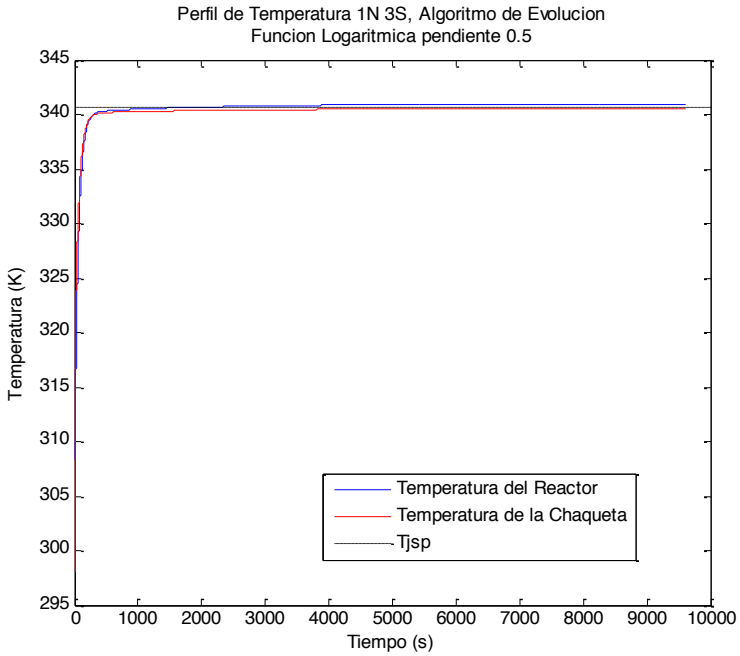


Figura 4.44 Perfil de Temperatura Función Logarítmica, pendiente 0.5, tres salidas

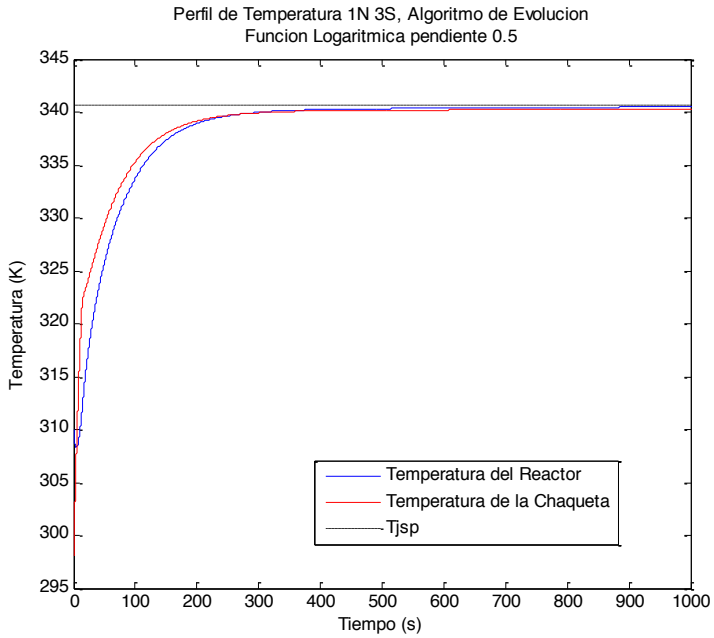


Figura 4.45 Perfil de Temperatura Función Logarítmica, pendiente 0.5, tres salidas (ampliación)

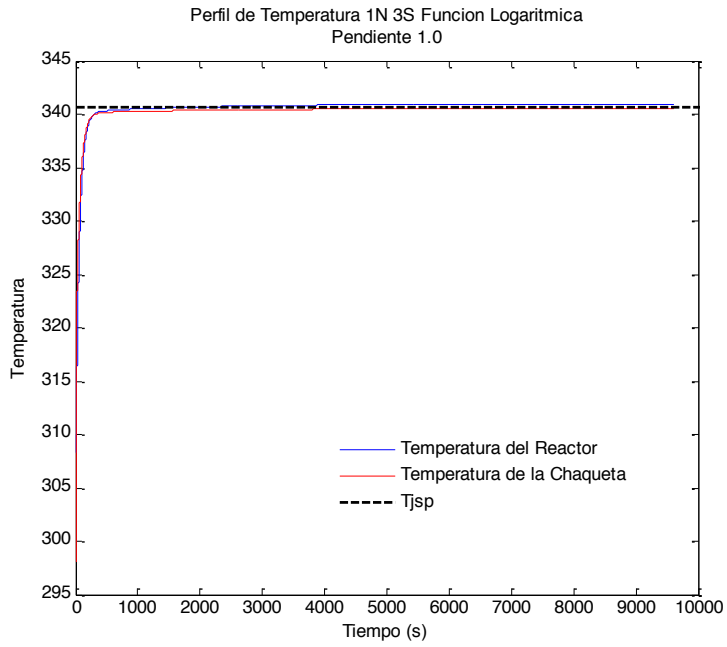


Figura 4.46 Perfil de Temperatura Función Logarítmica, pendiente 1.0, tres salidas

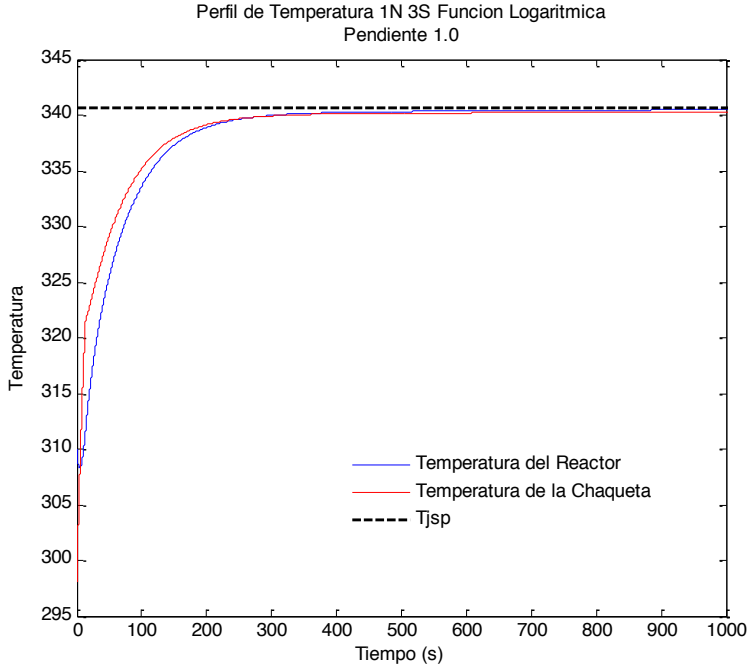


Figura 4.47 Perfil de Temperatura Función Logarítmica, pendiente 1.0, tres salidas (ampliación)

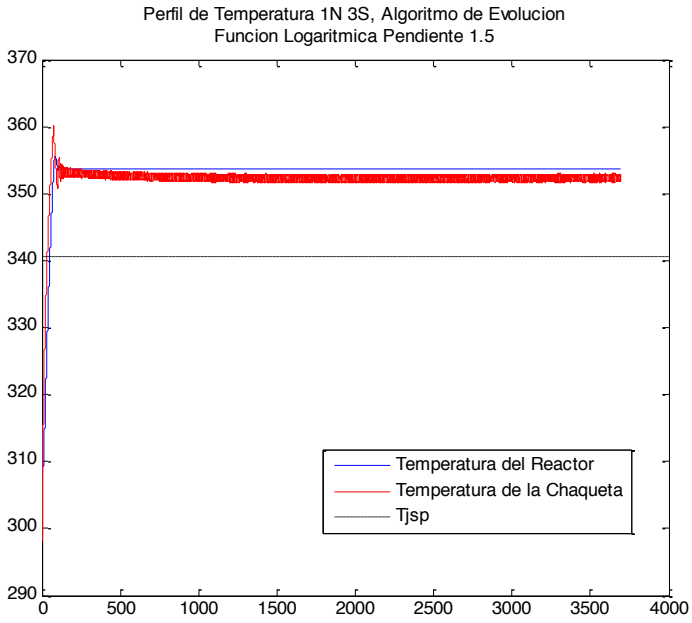


Figura 4.48 Perfil de Temperatura Función Logarítmica, pendiente 1.5, tres salidas

Cuando se realiza el incremento del número de salidas de la red neuronal se busca la evolución de la misma desde el parámetro de la arquitectura, para este trabajo se implementaron tres modificaciones en la arquitectura de la red neuronal, pendiente, número de salidas y tipo de función de excitación, para cuando la arquitectura pasa de tres salidas en cada una de las neuronas a cuatro salidas se puede demostrar la evolución de la red neuronal en comparación con las arquitecturas de dos y tres salidas (Figura 4.49-52), esto mediante la desaparición del sobre tiro que estaba presentando la temperatura de la chaqueta, con una rampa de calentamiento más cercana al sistema de reacción experimental, es decir, la rampa de calentamiento se encuentra dentro de un intervalo de tiempo entre 0 s - 550 s (0 min – 9 min) hasta alcanzar el set-point en 340.66 K (Figura 4.53-55), lo cual no sucede con la función de excitación de tipo tangencial en donde la rampa de calentamiento para este tipo de función se encuentra entre 0 s – 100 s, que puede considerarse rápida para el sistema experimental, para el sistema de cuatro salidas la red neuronal que presenta inestabilidad es la red con una pendiente de 1.0 que puede demostrar sobre-aprendizaje en el entrenamiento de la red neuronal no logrando un control de temperatura (Figura 4.53).

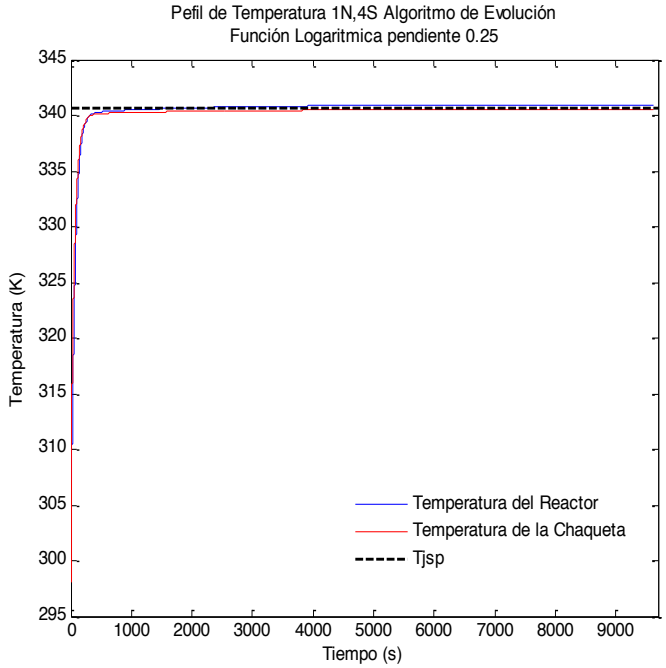


Figura 4.49 Perfil de Temperatura Función Logarítmica, pendiente 0.25, cuatro salidas

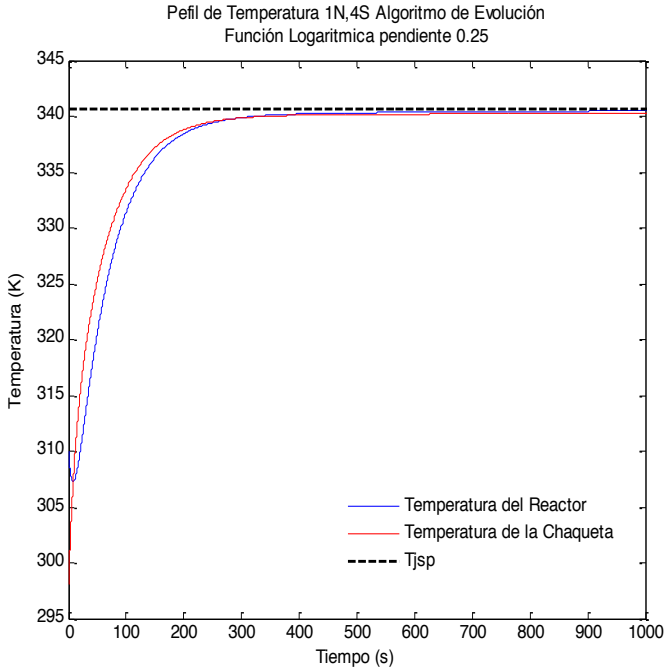


Figura 4.50 Perfil de Temperatura Función Logarítmica, pendiente 0.25, cuatro salidas (ampliación)

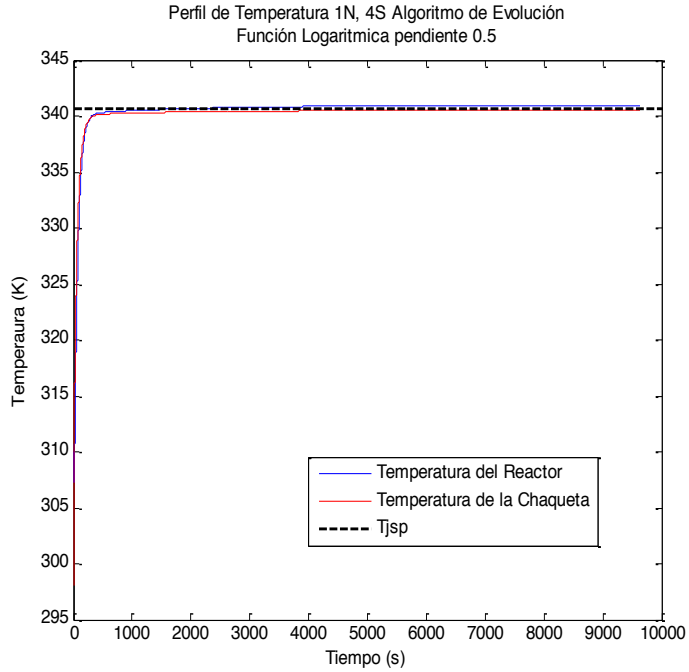


Figura 4.51 Perfil de Temperatura Función Logarítmica, pendiente 0.5, cuatro salidas

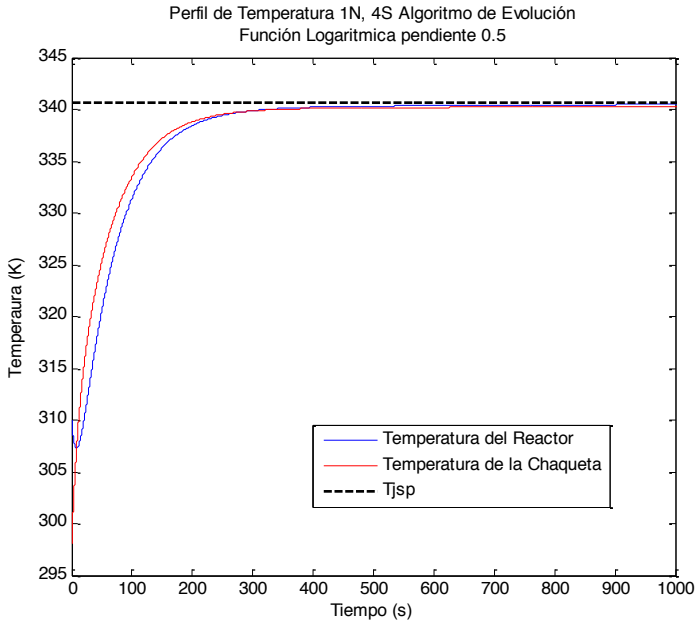


Figura 4.52 Perfil de Temperatura Función Logarítmica, pendiente 0.5, cuatro salidas (ampliación)

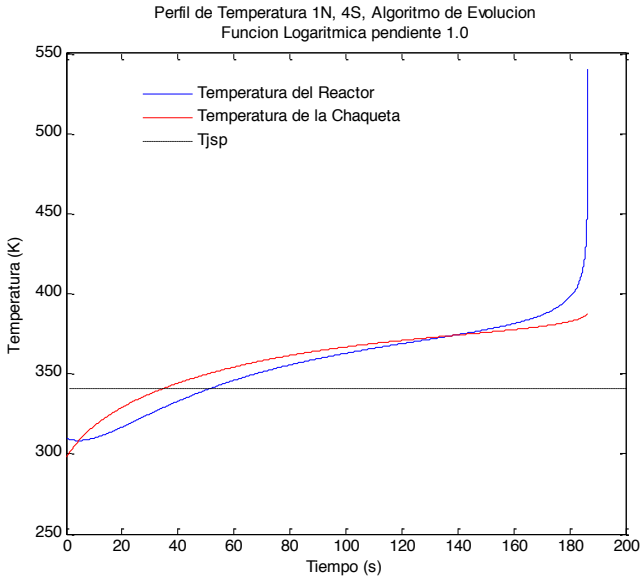


Figura 4.53 Perfil de Temperatura Función Logarítmica, pendiente 1.0, cuatro salidas

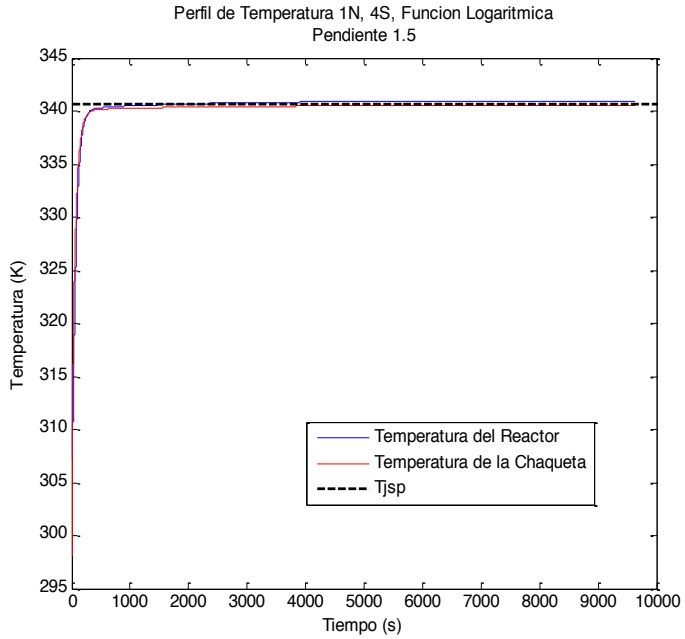


Figura 4.54 Perfil de Temperatura Función Logarítmica, pendiente 1.5, cuatro salidas

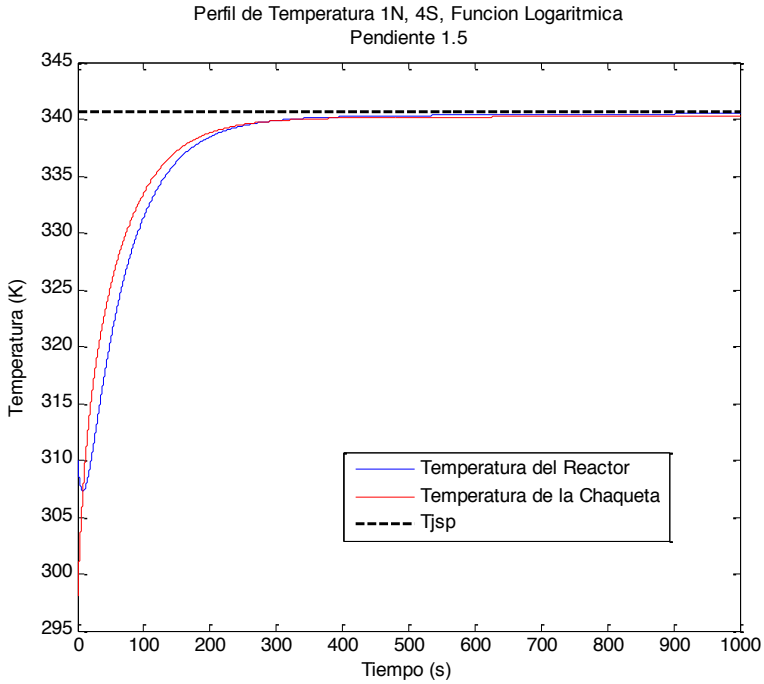


Figura 4.55 Perfil de Temperatura Función Logarítmica, pendiente 1.5, cuatro salidas (ampliación)

Algoritmo de Evolucion Función Radial Tipo Sesgo

Otra de las funciones usadas para la evolución de redes neuronales es la función de tipo radial con sesgo de tipo gaussiana, en este trabajo se estableció la función dicha función como algoritmo de evolución, mediante la modificación de la pendiente y el número de salida de la neurona. La respuesta del control para las pendientes de 0.25, 0.5, 1.0 y 1.5 con dos salidas, presentan un incremento no controlable de la temperatura del reactor (Figura 4.56 – 59), pero para la misma arquitectura pero con una pendiente de 2.0 la respuesta del control es aceptable alcanzando el set-point, eliminando de forma concreta el sobre tiro que se presenta tanto en la temperatura del reactor como en la chaqueta (Figura 4.60-4.61).

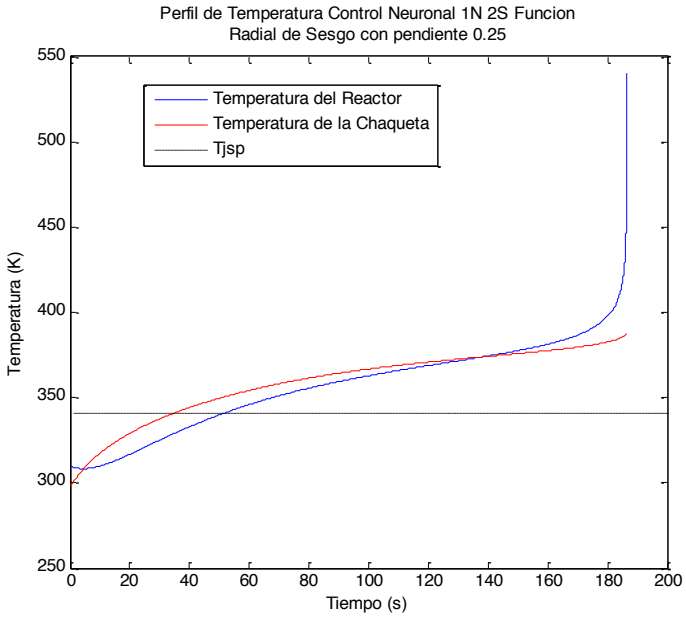


Figura 4.56 Perfil de Temperatura Función Base Radial, pendiente 0.25, dos salidas

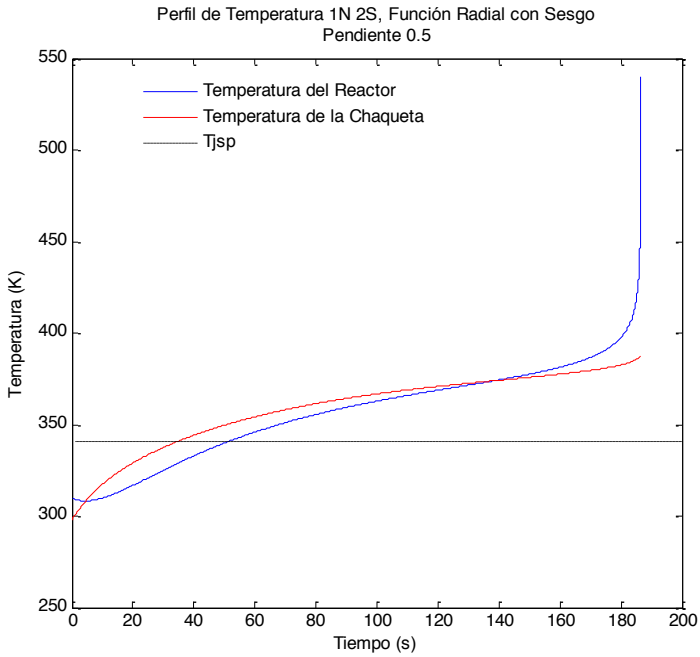


Figura 4.57 Perfil de Temperatura Función Base Radial, pendiente 0.5, dos salidas

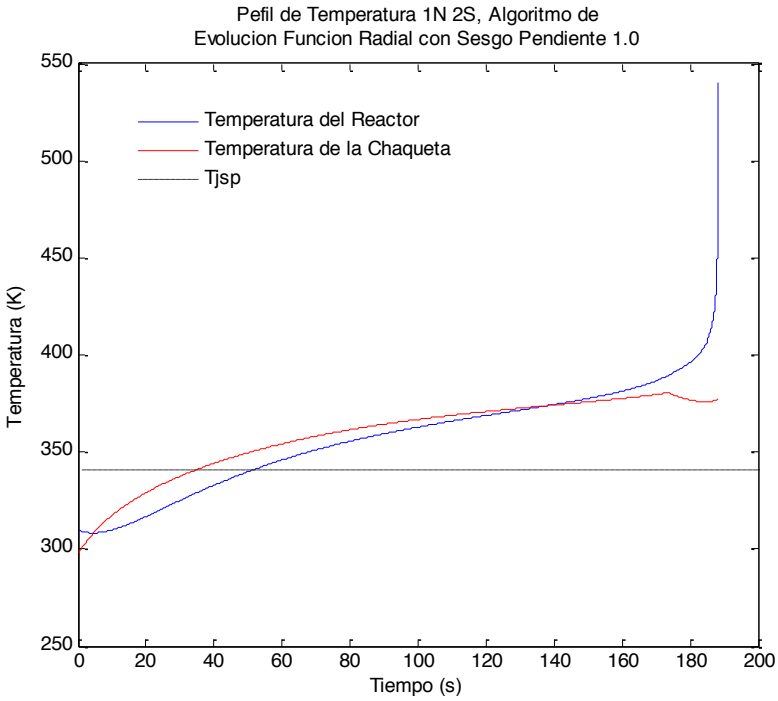


Figura 4.58 Perfil de Temperatura Función Base Radial, pendiente 1.0, dos salidas

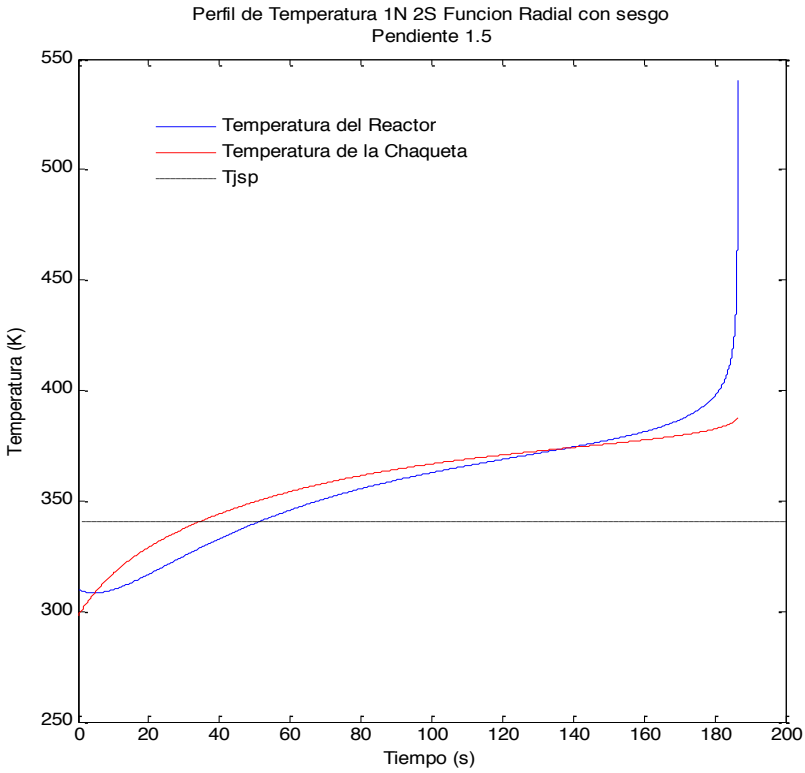


Figura 4.59 Perfil de Temperatura Función Base Radial, pendiente 1.5, dos salidas

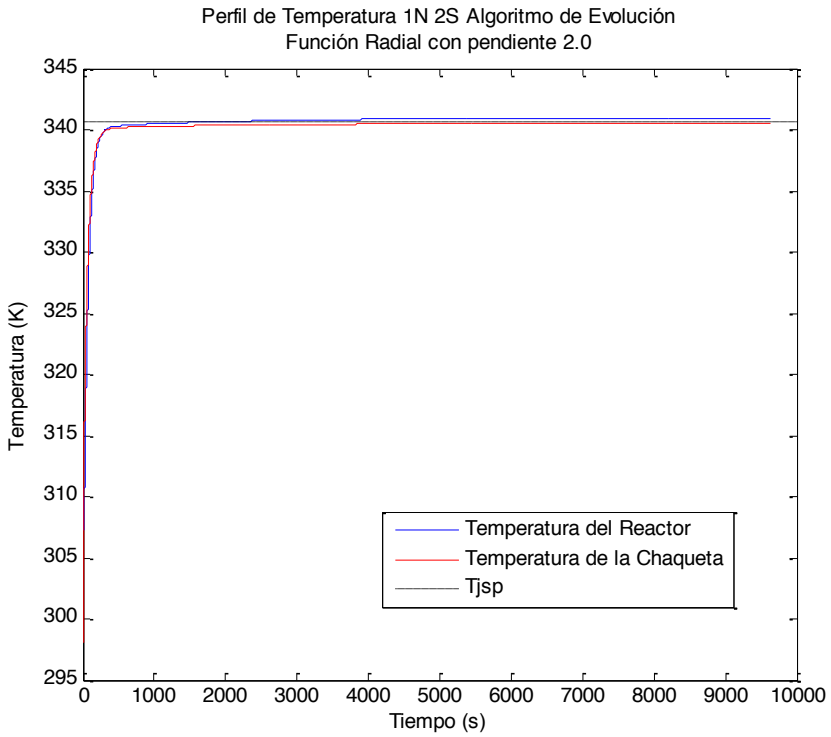


Figura 4.60 Perfil de Temperatura Función Base Radial, pendiente 2.0, dos salidas

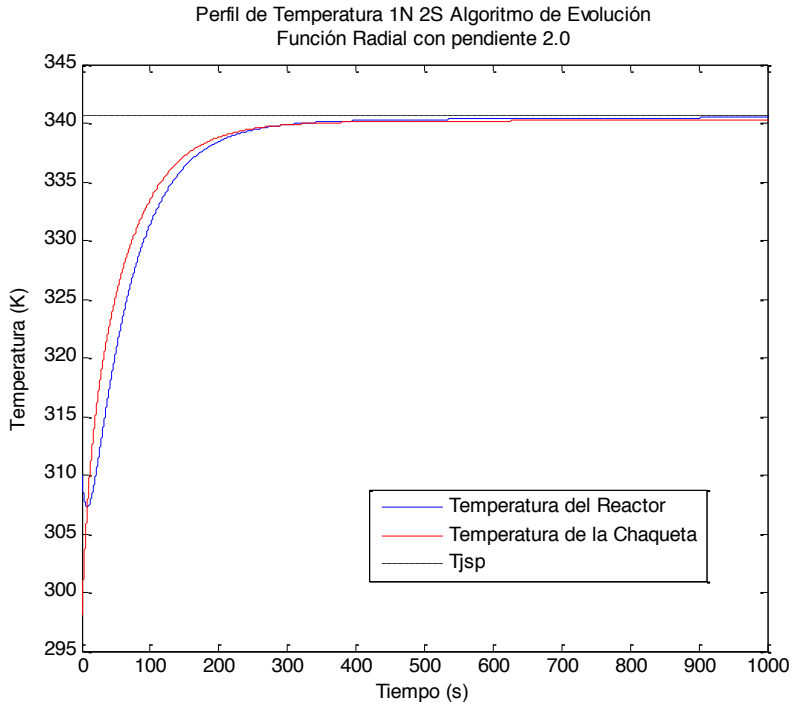


Figura 4.61 Perfil de Temperatura Función Base Radial, pendiente 2.0, dos salidas (ampliación)

Para cuando la arquitectura se ve modificada con respecto al número de salidas pasando de dos salidas a tres salidas de las neuronas, la evolución de la red neuronal se hace presente para pendientes de la función de excitación que para el caso de dos salidas la respuesta de control no fue adecuada, pero para el sistema de control con pendiente 0.25 la respuesta de control se comporta de forma semejante que para la arquitectura de dos salidas (Figura 4.62) , para la pendiente de 0.5 la respuesta del control es aceptable (Figura 4.63 - 4.64), pero para la pendiente de 1.0 la respuesta del control se encuentra alrededor de 35 grados por encima del set-point (Figura 4.65), para la pendiente de 1.5 la respuesta del control es semejante a la respuesta de control de dos salidas (Figura 4.66), a diferencia que en la arquitectura de dos salidas en donde el control elimina de forma sustancial el sobre tiro para una arquitectura de tres salidas el sobre tiro en la chaqueta aun se hace presente (Figura 4.67-4.68).

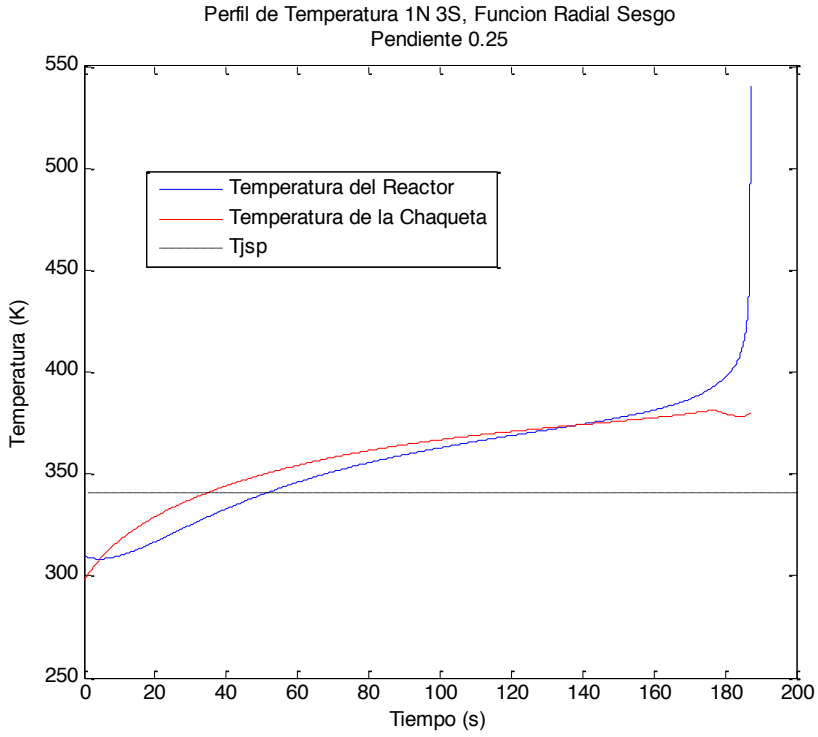


Figura 4.62 Perfil de Temperatura Función Base Radial, pendiente 0.25, tres salidas

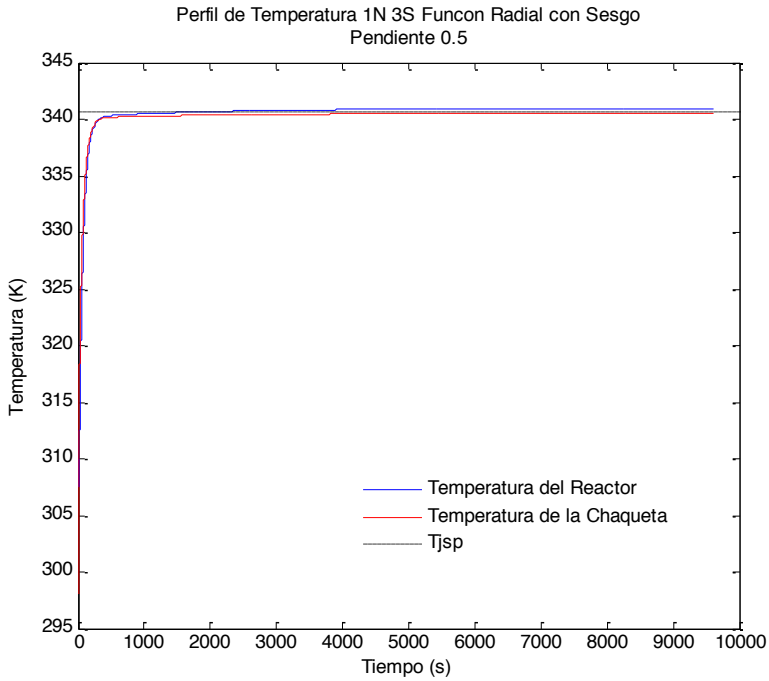


Figura 4.63 Perfil de Temperatura Función Base Radial, pendiente 0.5, tres salidas

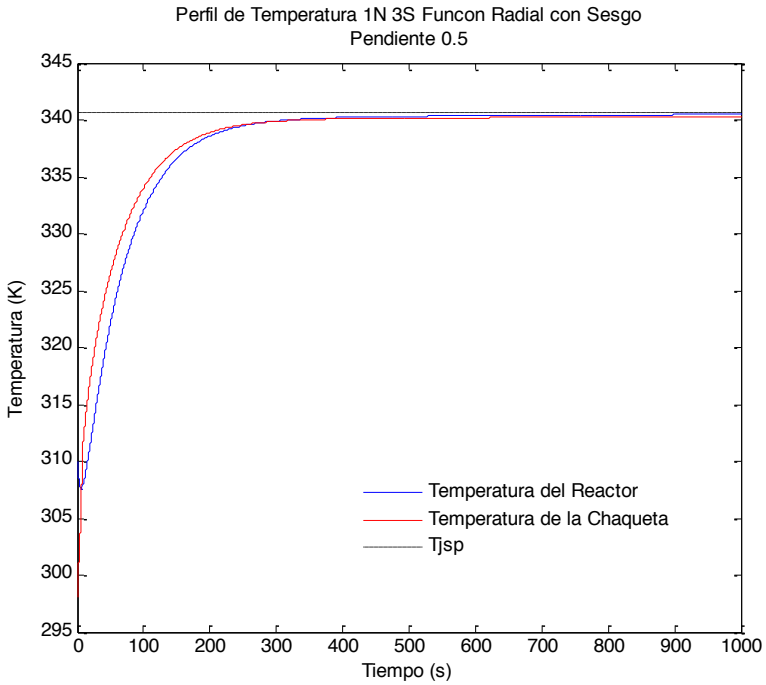


Figura 4.64 Perfil de Temperatura Función Base Radial, pendiente 0.5, tres salidas (ampliación)

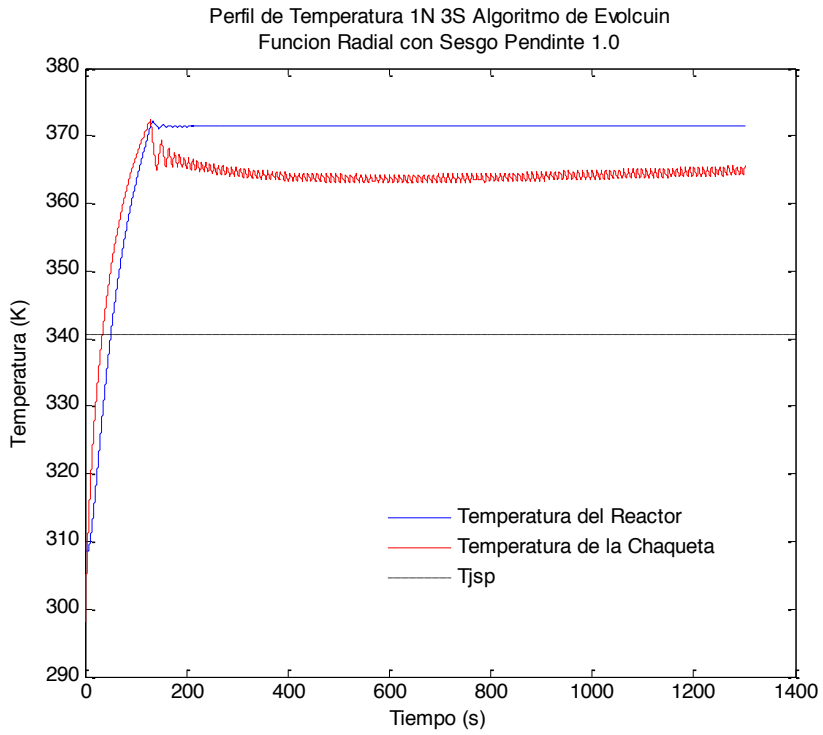


Figura 4.65 Perfil de Temperatura Función Base Radial, pendiente 1.0, tres salidas

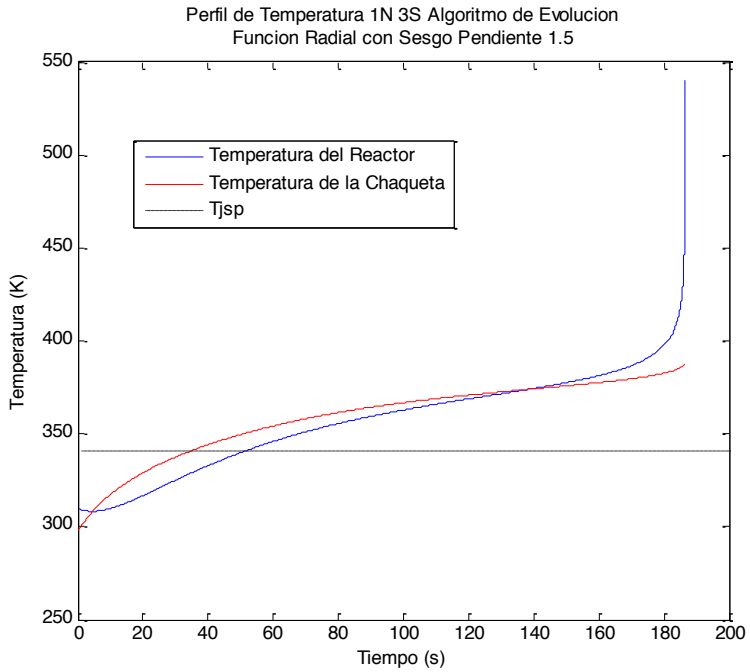


Figura 4.66 Perfil de Temperatura Función Base Radial, pendiente 1.5, tres salidas

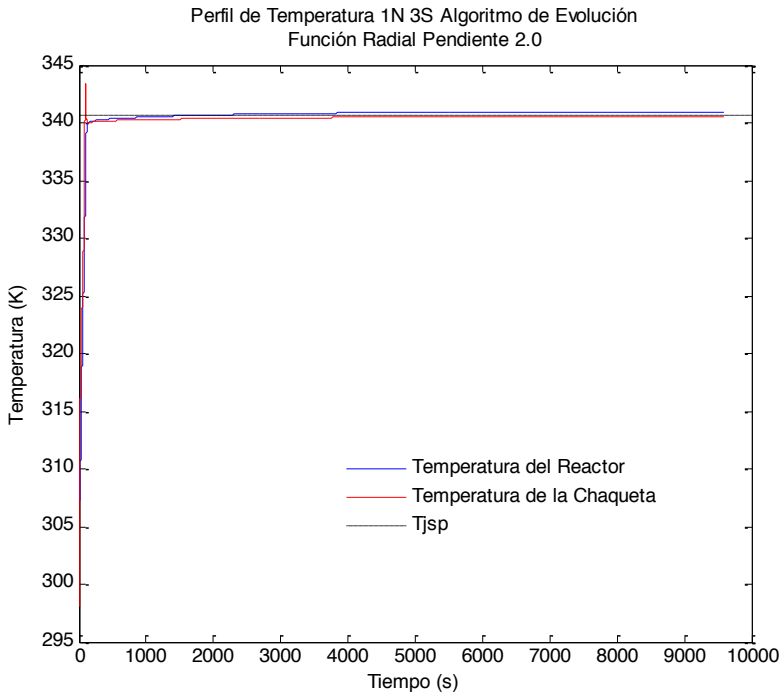


Figura 4.67 Perfil de Temperatura Función Base Radial, pendiente 2.0, tres salidas

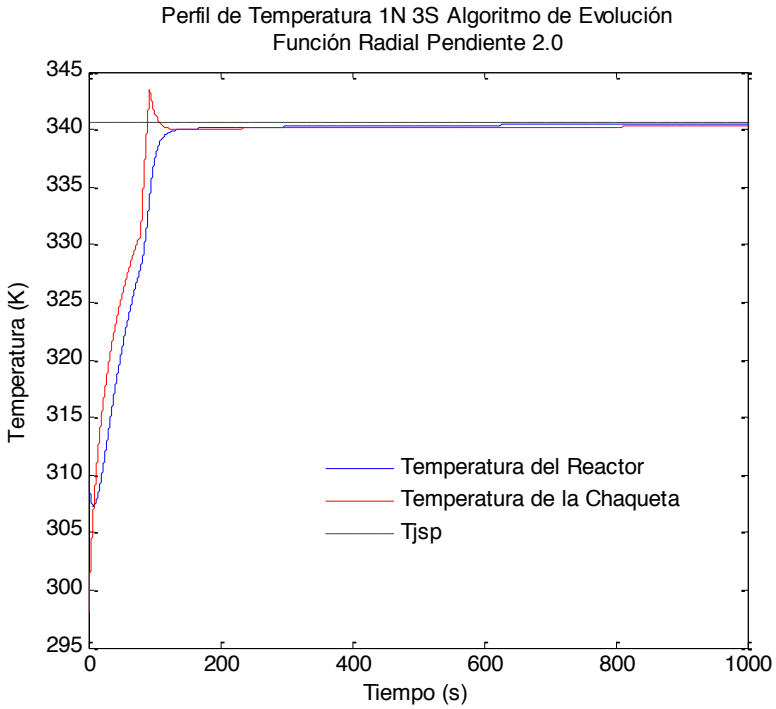


Figura 4.68 Perfil de Temperatura Función Base Radial, pendiente 2.0, tres salidas (ampliación)

El aumento del número de salidas en la arquitectura de la red neuronal muestra evolución en la red neuronal con respecto a la respuesta del control de temperatura, para un incremento de salidas (4 salidas) la respuesta del control para una pendiente de 0.25 elimina el sobre tiro de la temperatura de la chaqueta y del reactor, alcanzando el set-point en un tiempo aceptable dentro del intervalo del equipo experimental (Figura 4.69-4.70), el control para una pendiente de 0.5 es deficiente mostrando un incremento en la temperatura del reactor volviendo al sistema no controlable (Figura 4.71), para la pendiente de 1.0 la respuesta de control es semejante a la obtenida con la pendiente de 0.5 lo que significa que no se obtiene un control adecuado (Figura 4.72), al incrementar la pendiente a 1.5, no se muestra mejora en la respuesta del control lo que significa que la red neuronal está presentando sobre aprendizaje con respecto a los pesos de entrenamiento y de la arquitectura de la red neuronal volviendo redundante los valores no óptimos de los pesos sinápticos de la red neuronal (Figura 4.73), esto demuestra que los estados de aprendizaje de la red neuronal son menores que los mostrados por otro tipo de función de excitación, esta red neuronal con la pendiente de 1.5 solo presenta un estado de aprendizaje lo que significa que solo se obtienen 2 neuronas evolucionadas en la capa oculta en la red neurona lo cual no es suficiente para el conocimiento de la dinámica del proceso. La red neuronal con una pendiente de 2.0 demuestra una adaptación a la dinámica del proceso representado un respuesta de dentro del intervalo y alcanzando el set-point de la reacción (Figura 4.74-4.75), para la arquitectura con la pendiente 2.0 y cuatro salidas los estados de aprendizaje son 5 lo que significa que la red neuronal evoluciona hasta alcanzar un número de 64 neuronas evolucionadas, que probablemente provoca que el sistema de control no elimine el sobre tiro de la chaqueta que aún se hace presente en la respuesta del control.

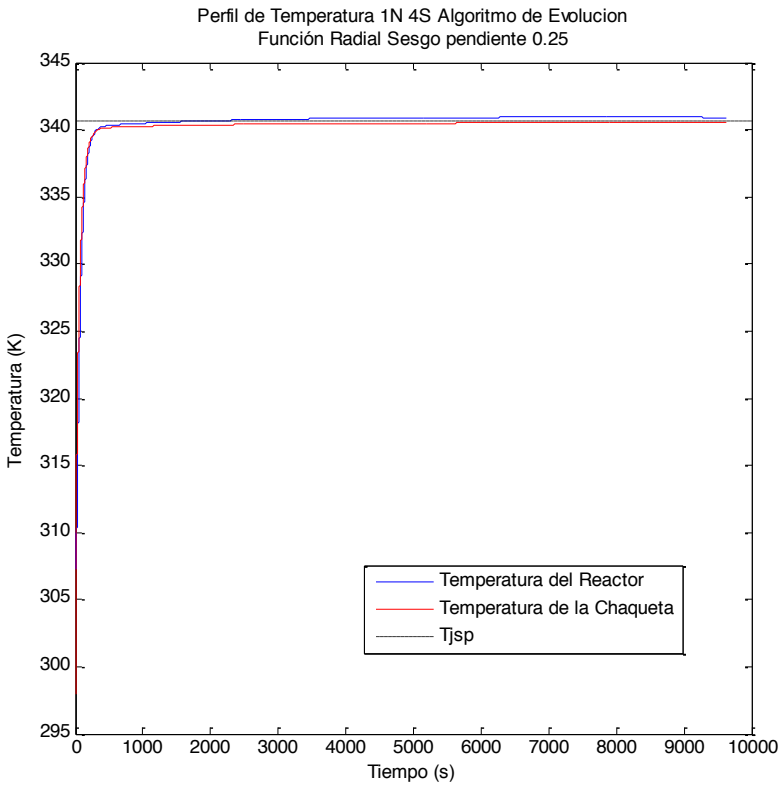


Figura 4.69 Perfil de Temperatura Función Base Radial, pendiente 0.25, cuatro salidas

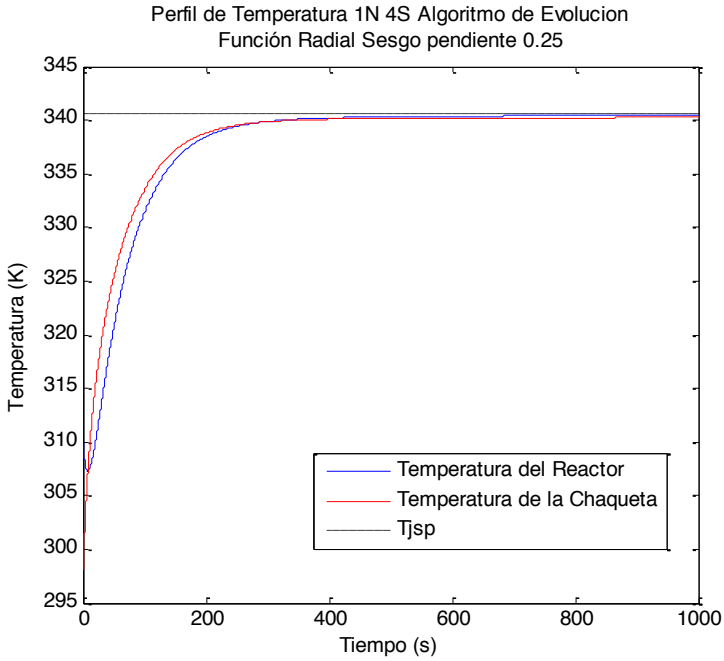


Figura 4.70 Perfil de Temperatura Función Base Radial, pendiente 0.25, cuatro salidas (ampliación)

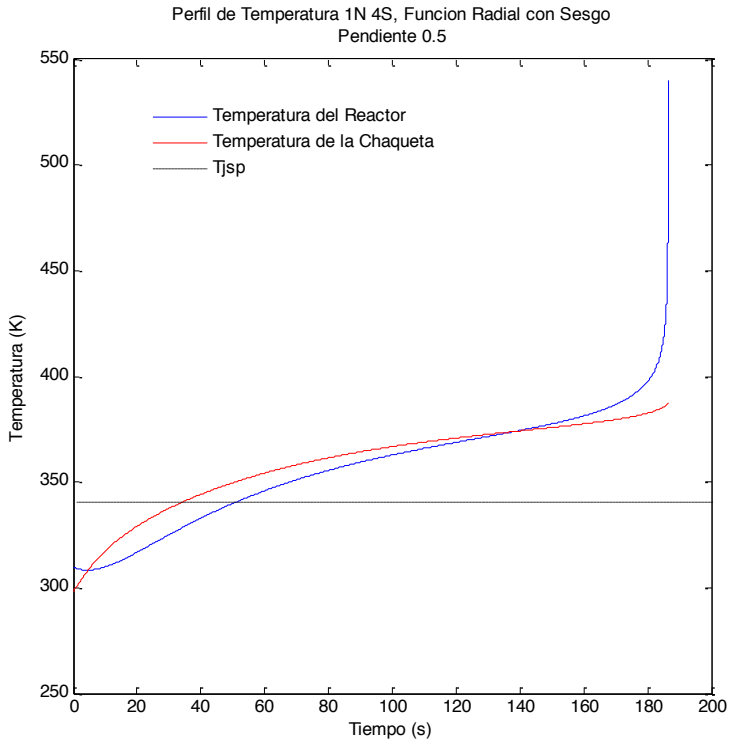


Figura 4.71 Perfil de Temperatura Función Base Radial, pendiente 0.5, cuatro salidas

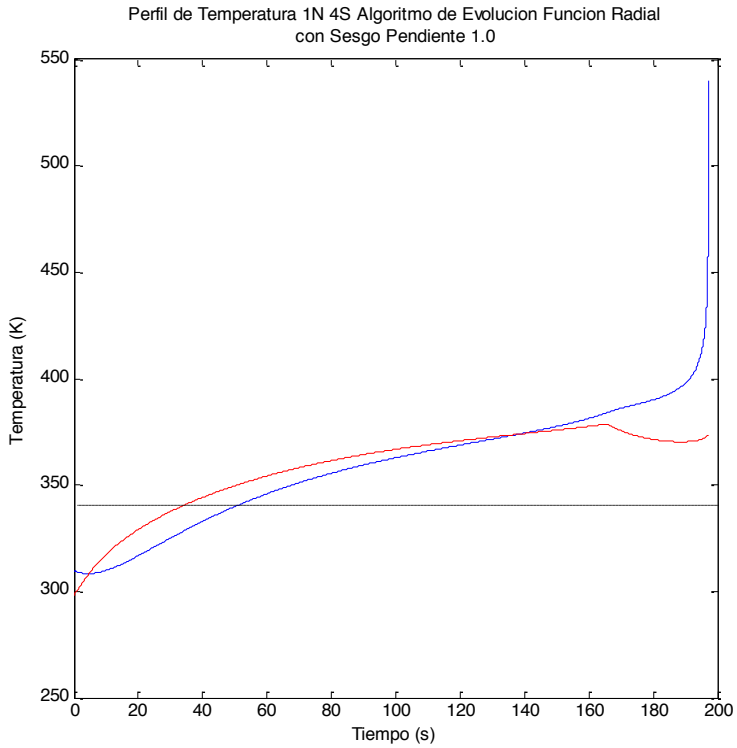


Figura 4.72 Perfil de Temperatura Función Base Radial, pendiente 1.0, cuatro salidas

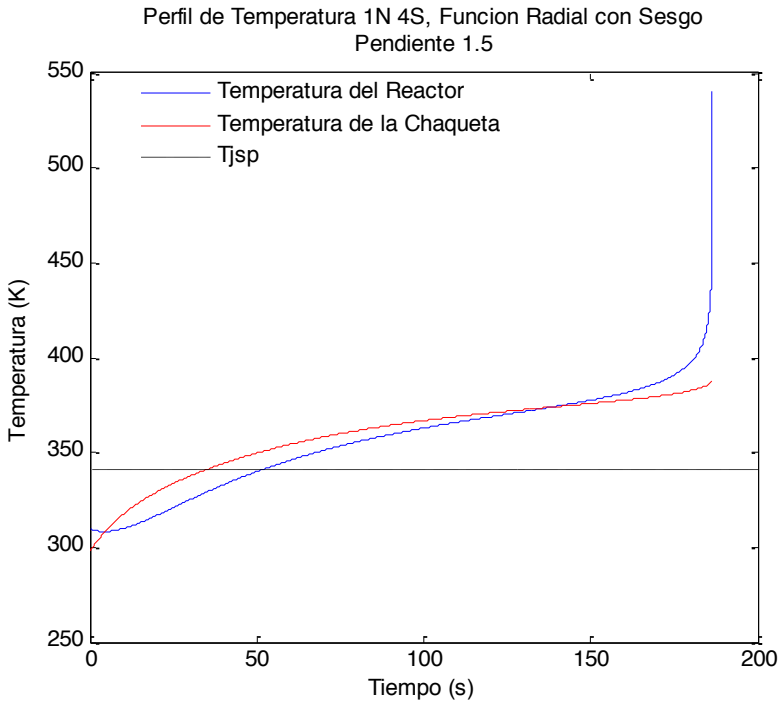


Figura 4.73 Perfil de Temperatura Función Base Radial, pendiente 1.5, tres salidas

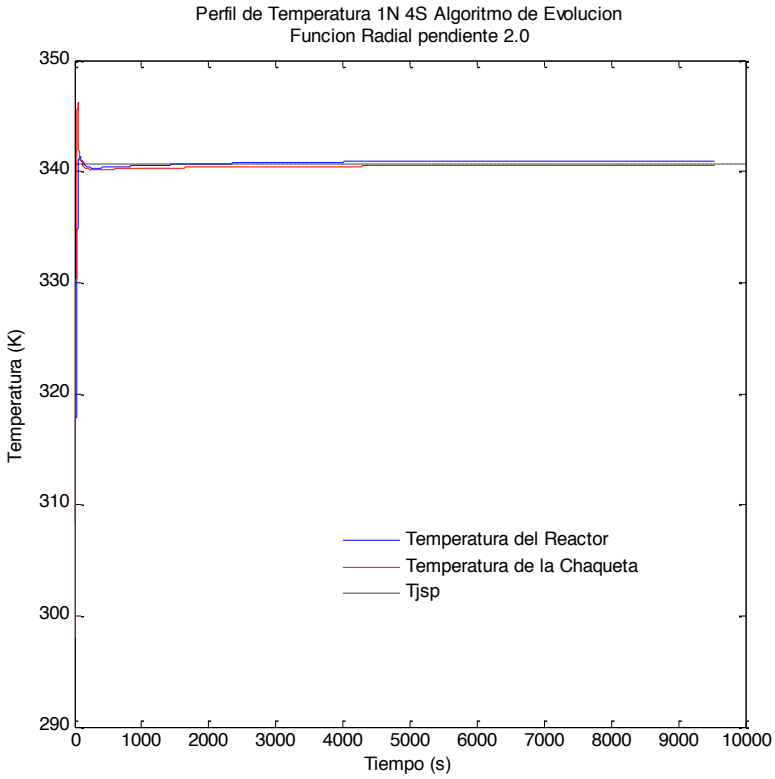


Figura 4.74 Perfil de Temperatura Función Base Radial, pendiente 2.0, cuatro salidas

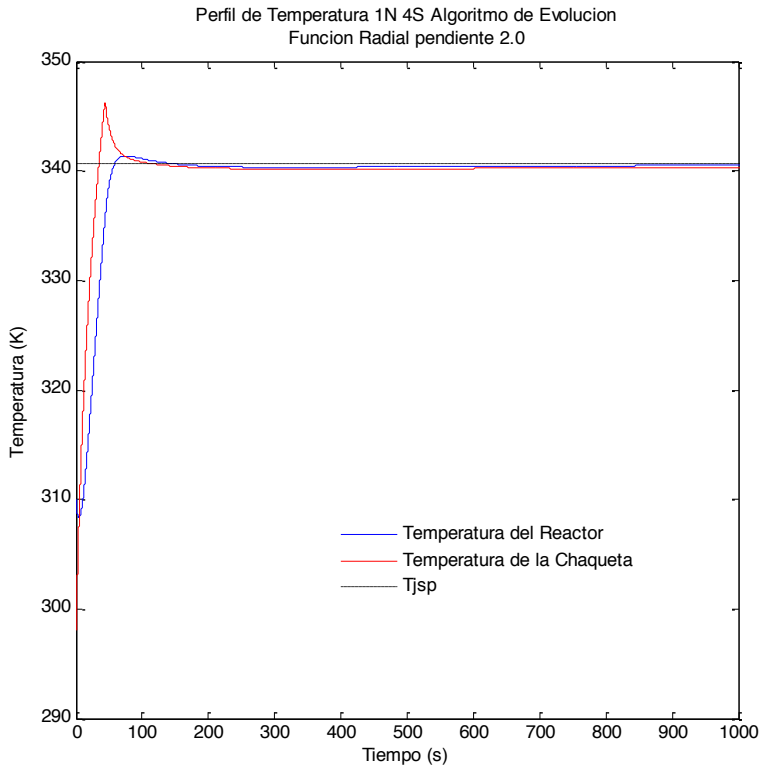


Figura 4.75 Perfil de Temperatura Función Base Radial, pendiente 2.0, cuatro salidas (ampliación)

Error de Control

Una de las formas establecidas por los especialistas en control de procesos para la selección de un control óptimo es mediante el cálculo del error de respuesta del control, para esto existen diferentes métodos establecidos para dicho cálculo, pero existe una limitante con respecto a estos métodos, la cual radica principalmente en el ajuste a sistemas de tipo lineal de un orden especifico. Para el caso de sistemas no lineales o con dinámica poco predecible los métodos de cálculo del error pueden ser poco efectivos y no demostrar que dicho valor estimado sea un punto de referencia para seleccionar el control óptimo, para lo cual se deben hacer diferentes consideraciones antes de establecer el método adecuado del cálculo del error. Para este trabajo se optó por utilizar el método de la Integral del cuadrado del error (ISE *por sus siglas en inglés*) (Ec. 4.1), además se seleccionaron respuestas de control en donde existiera disminución del sobre tiro, tiempo de respuesta y arquitectura evolucionada más completa.

$$ISE = \int_0^\infty e^2(t)dt \tag{4.1}$$

Para esto se han seleccionado tres estructuras con arquitectura básica una neurona de entrada, una neurona en la capa oculta y una neurona en la capa de salida (1, 1, 1), además se optaron por las estructuras que presentaban un mayor número de estados de aprendizaje, también se analizaron las tres funciones de excitación con los valores de pendiente en donde se presentaron los estados de aprendizaje con mayor estabilidad y óptimos para la red neuronal, la Tabla 4.2 muestra los resultados del error calculado para los sistemas de control seleccionados

Tabla 4.2 Integral del cuadrado del Error (ISE)

Arquitectura Neuronal (1,1,1)	**Función de Excitación (FT, FL, FBR)**	**Integral del cuadrado del error (ISE) 10^{-2}**	**Estados de Evolución**
Cuatro salidas	Tangencial pendiente 1.5	0.3191	1.0737×10^{9}
Cuatro salidas	Logarítmica pendiente 1.5	0.2966	1.0737×10^{9}
Tres salidas	Base Radial Gaussiana 0.5	0.1887	27

Los resultados obtenidos del error de las diferentes estructuras de las redes neuronales proporcionan una línea de selección del sistema de control a implementar de forma experimental, para la red neuronal que presenta una función de tipo base radial de tipo gaussiana es la red que muestra el error menor con respecto a otras funciones de tipo tangencial y logarítmica, además que también presenta una menor cantidad de salidas lo cual puede ser en beneficio en la disminución del sobre aprendizaje que pueda presentar la red neuronal. La selección de dicha red también se sustenta en la magnitud de error con respecto al control de tipo PID y GMC que se muestran en la Tabla 4.3, tales valores tienen un incremento mayor que los proporcionados por el control de tipo neuronal para cualquier tipo de función de excitación y algoritmo de evolución

Tabla 4.3 Error del control tipo convencional

Tipo de Control	**ISE**
PID	0.3662
GMC	0.0198

4.1.6 Perturbaciones

La forma de comprobar la eficiencia del sistema de control en mediante la implementación de perturbaciones en el control, las perturbaciones se implementan mediante un cambio escalón con respecto al tiempo, plantean dos tipos de cambios escalón un incremento de la temperatura y un decremento de la misma (Figura 4.76-4.77), estos cambios se implementa para los tres tipos de control analizados en este trabajo (PID, GMC, RNAE).

La respuesta del controlador PID para la perturbación muestra que el sistema no llega a controlar la perturbación que se presenta (Figura 4.78), provocando un inestabilidad en la temperatura del reactor y la chaqueta. El control de tipo GMC demuestra ser un control con mejor tendencia a controlar sistemas no lineales como los sistemas batch, al someter al control GMC a la perturbación con respecto al tiempo la respuesta de este, es mejor con respecto con el control PID (Figura 4.70-4.80), el cual sigue la trayectoria planteada por la perturbación con respecto al tiempo, aunque existe cierto esfuerzo del control al inicio de la perturbación con un sobre-tiro al regresar al set-point.

Para el sistema de control con redes neuronales evolutivas se observa que el sistema mejora de forma significativa la respuesta con respecto al control GMC, pues el control basado en RNAE elimina el esfuerzo cuando se presenta la perturbación al igual cuando regresa al set-point, lo que significa que la propuesta de control basado en redes neuronales puede ser una alternativa viable (Figura 4.81-4.82) principalmente porque la red neuronal no requiere de linealizar el modelo matemático como lo requiere el modelo GMC, aunque la red neuronal requiere de un entrenamiento previo para conocer la dinámica del proceso que puede considerarse como una sintonización del mismo.

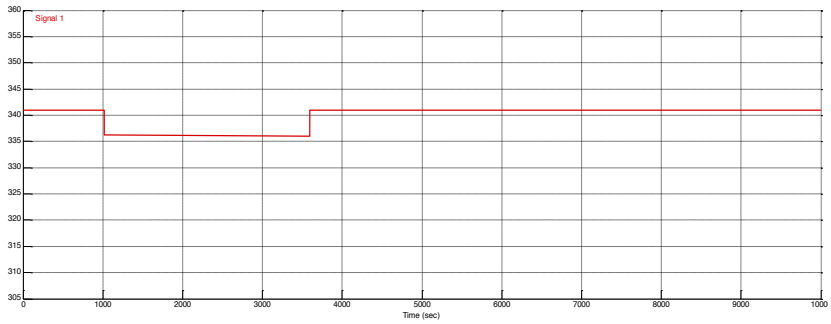


Figura 4.76 Perturbación cambio escalón decremento de la temperatura

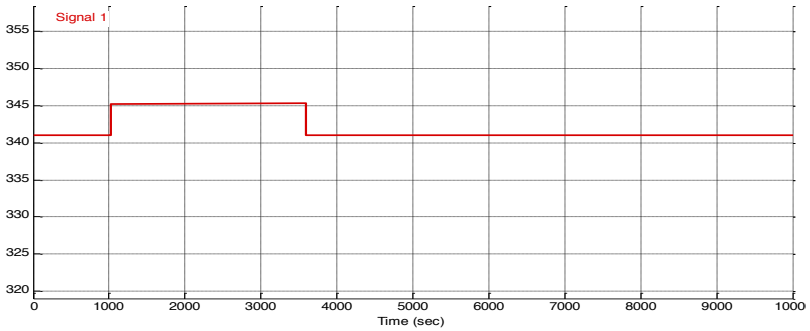


Figura 4.77 Perturbación cambio escalón incremento de la temperatura

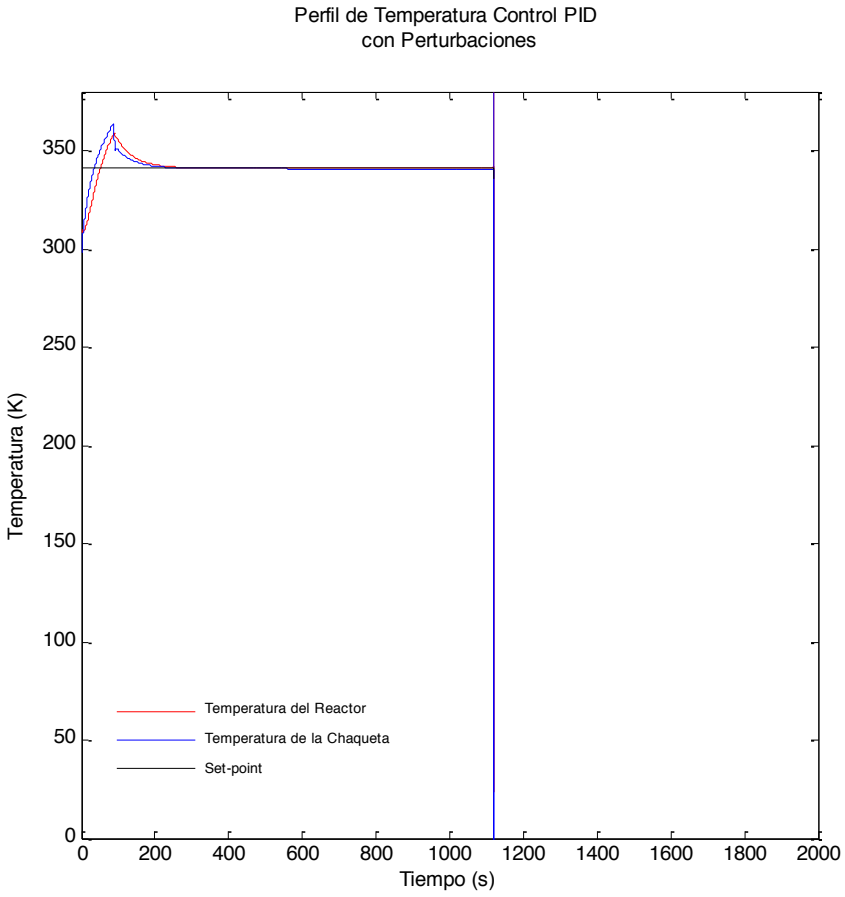


Figura 4.78 Perfil de Temperatura con perturbación control PID

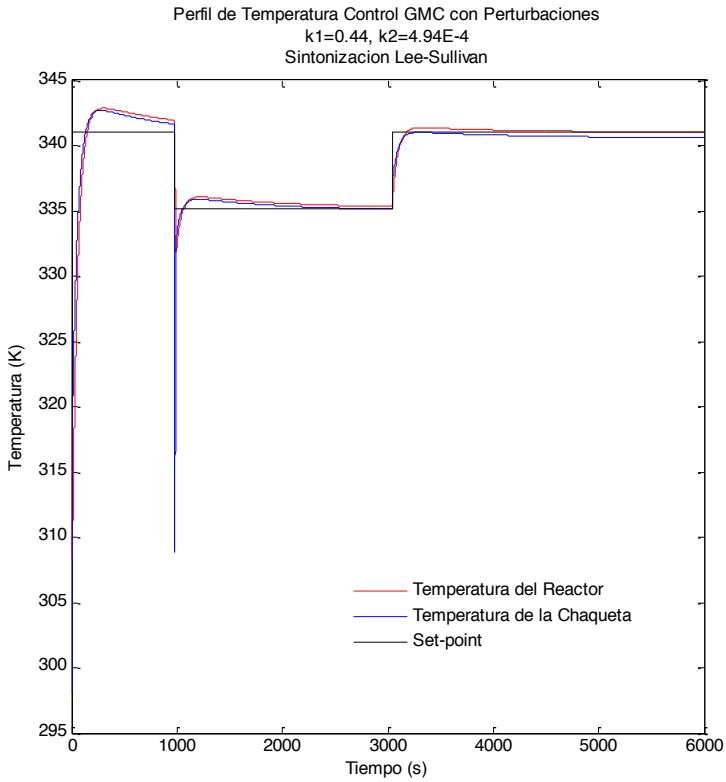


Figura 4.79 Perfil de Temperatura perturbación decremento en la temperatura control GMC

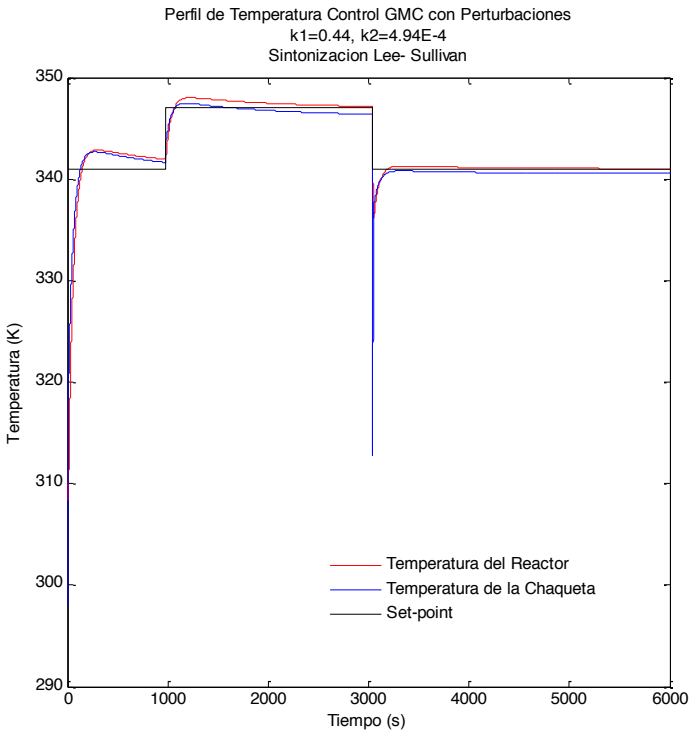


Figura 4.80 Perfil de Temperatura perturbación incremento en la temperatura control GMC

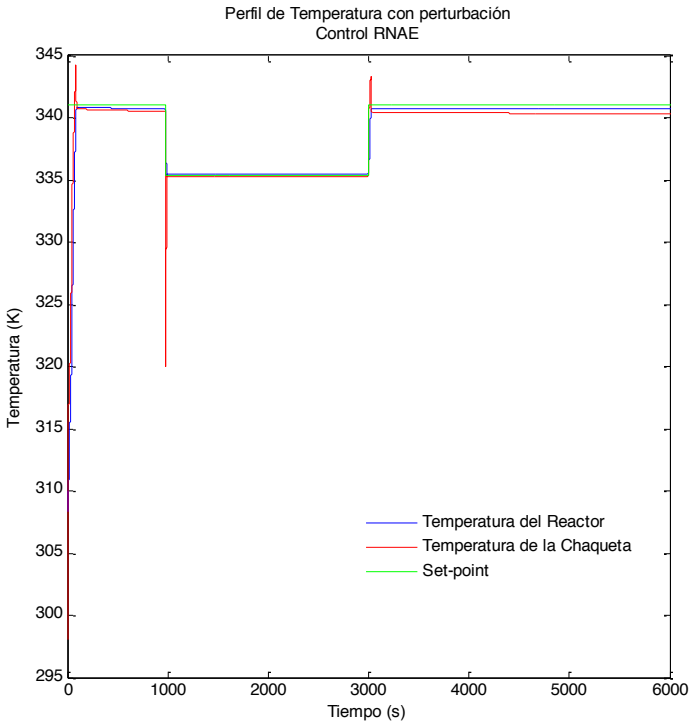


Figura 4.81 Perfil de Temperatura perturbación decremento de la temperatura control RNAE

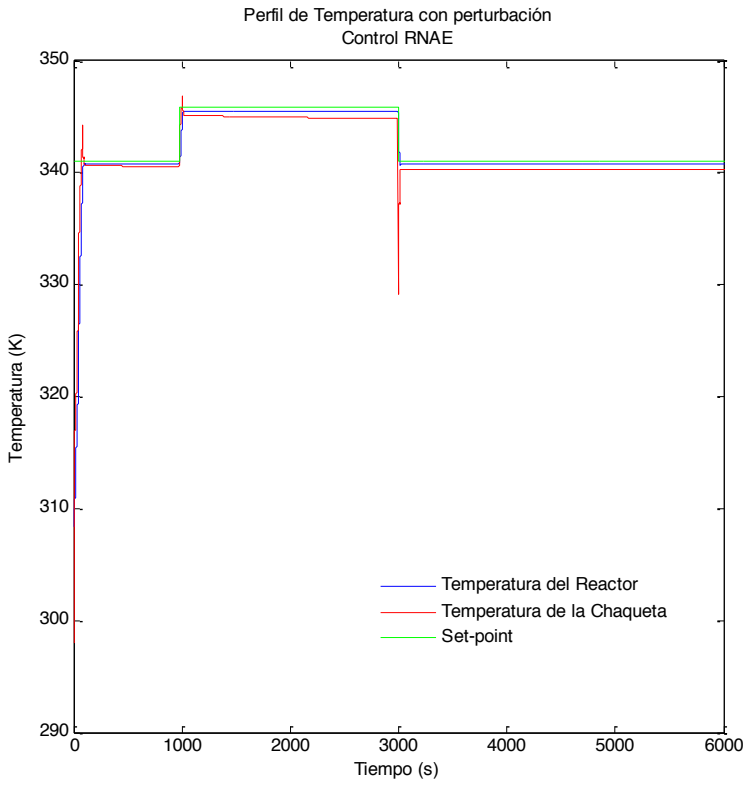


Figura 4.82 Perfil de Temperatura perturbación incremento de la temperatura control RNAE

4.2 Parte Experimental

En la sección experimental se presentan los resultados obtenidos de la instrumentación virtual del reactor batch de laboratorio, es decir, se presenta la aplicación de los resultados teóricos en la síntesis de la red neuronal en LabView.

4.2.1 Resultados Experimentales

Una de las características mas importantes del control de procesos es su implementación en sistemas experimentales o reales, para el caso de las redes neuronales como controladores son pocos los trabajos reportados en donde la aplicación se de forma experimental, esto debido principalmente a que las redes neuronales son concebidas como sistemas expertos aplicables a robótica y reconocimiento de imágenes. En el presente trabajo se implemento la red neuronal evolutiva como sistema de control de temperatura en un reactor tipo batch de forma experimental.

Para poder establecer el control de forma experimental se debe seleccionar en primera cuenta el tipo de arquitectura de red neuronal a utilizar en el procedimiento experimental, para este trabajo se seleccionó la red neuronal que presento la menor magnitud del error integral al cuadrado (*ISE por sus siglas en inglés*), tal red neuronal seleccionada fue la red neuronal con una neurona de entrada, una neurona en la capa oculta y una neurona en la capa de la salida, con una función de tipo base radial gaussiana (*FBR por sus siglas en inglés*) con una pendiente de 0.5, tres salidas de cada neurona y un entrenamiento de Quasi-Newton de gradiente descendente (*BFGS por sus siglas en inglés*). La implementación se realiza en el reactor batch de tamaño laboratorio, el cual tiene implementado un sistema de monitoreo de temperatura mediante un sensor de temperatura (*TST por sus siglas en inglés*) el cual mide intensidad que posteriormente transforma en voltaje, para así obtener la medición de la temperatura mediante instrumentos virtuales aplicados en programación modular en LabView.

Para la comprobación del sistema de control establecido de forma teórica, se realizan tres experimentos en donde se demuestre el seguimiento del perfil de temperatura obtenido en la simulación de la red neuronal evolutiva.

Experimento 1

El experimento muestra la comprobación del control implementado de forma modular en LabView, se obtiene de forma secuencial el perfil de temperatura de la chaqueta (Figura 4.83), de forma similar se muestra el perfil de temperatura del reactor (Figura 4.84), para la apreciación de los perfiles y su analogía con los obtenidos en la simulación del modelo matemático, las escalas del tiempo se ven modificadas para obtener una ampliación del perfil de temperatura y demostrar el seguimiento de la trayectoria del control (Figura 4.85 – 4.86). La modificación de las escalas de tiempo de reacción en los perfiles obtenidos demuestra que la trayectoria del control se cumple.

La Figura 4.87 muestra la respuesta del control inteligente implementado en el sistema de reacción, comparando los diferentes tipos de perfiles obtenidos tanto de forma teórica como experimental. Al realizar la ampliación del perfil de temperatura, se observa que los perfiles de temperatura del reactor y de la chaqueta de calentamiento muestran una tendencia lineal para alcanzar la temperatura del set point, esto se debe principalmente a la secuencia de calentamiento que muestra el equipo experimental, el cual está basado en una resistencia eléctrica sensible a las variaciones de voltaje e intensidad de la línea eléctrica.

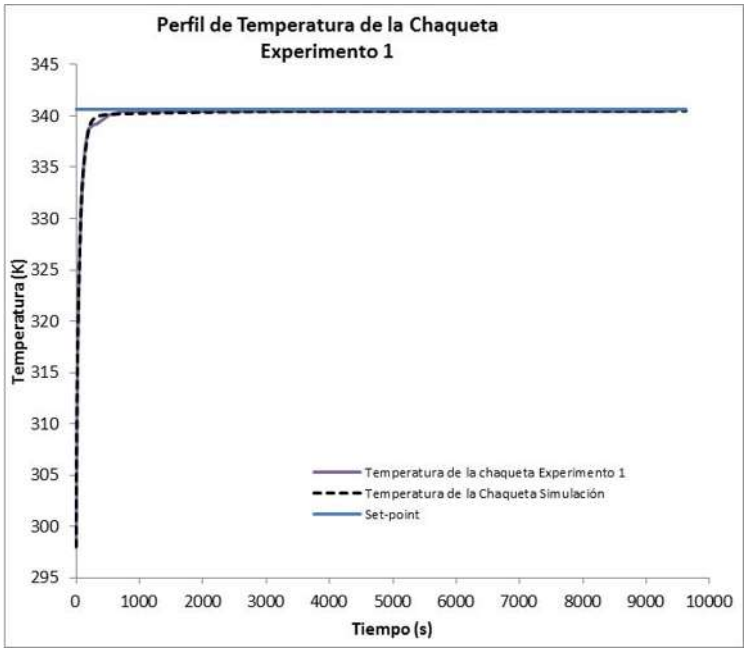


Figura 4.83 Perfil de Temperatura de la Chaqueta (Experimento 1)

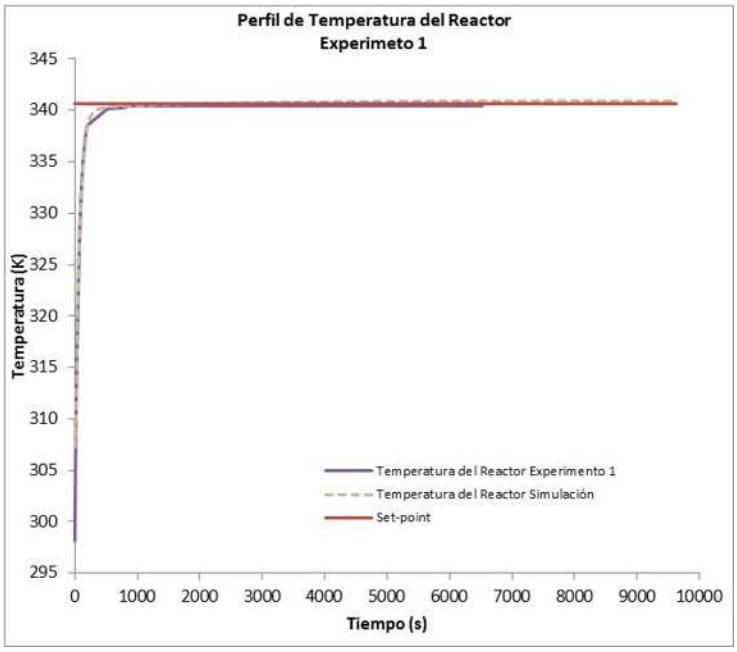


Figura 4.84 Perfil de Temperatura del Reactor (Experimento 1)

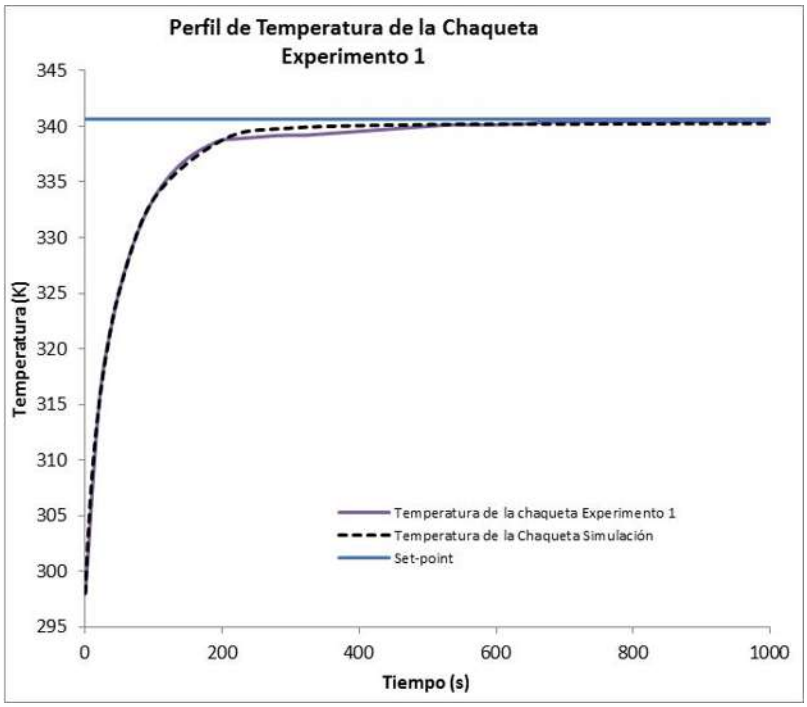


Figura 4.85 Perfil de Temperatura de la Cahqueta (Experimento 1)

Ampliación

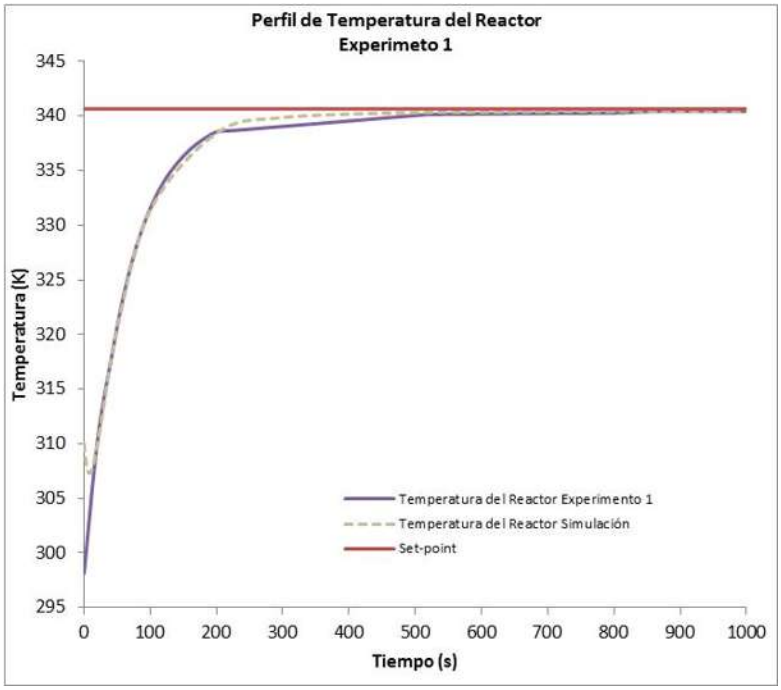


Figura 4.86 Perfil de Temperatura del Reactor (Experimento 1)

Ampliación

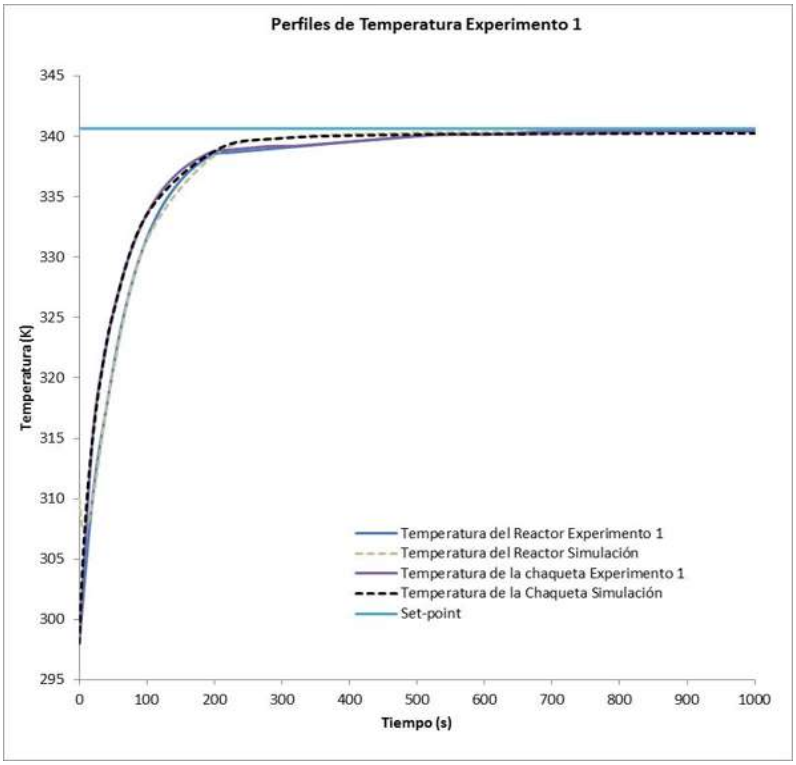


Figura 4.87 Perfiles de Temperatura Teóricos y Experimentales

Experimento 2

Para el experimento 2, la respuesta del control implementado experimentalmente proporciona perfiles de temperatura con respuesta semejante a los perfiles de temperatura obtenidos en las simulaciones del control inteligente, para la chaqueta la trayectoria de control se cumple (Figura 4.88), para el caso de la temperatura del reactor el perfil de temperatura cumple con la trayectoria de control obtenido en la simulación (Figura 4.89). Para observar el inicio del perfil de las temperatura, los límites del eje del tiempo son acotados a un tiempo de 1000 los cuales no representan el total de la reacción pero sirve como parámetro para establecer el comportamiento de la reacción, la acción de acotamiento de los ejes muestra que la tendencia lineal en un intervalo de tiempo de la temperatura del reactor y la chaqueta para el experimento 2 no se hace tan presente como en el experimento 1 (Figura 4.90-4.91).

La Figura 4.92 representa el conjunto de perfiles de temperatura tanto experimentales como los obtenidos por las simulaciones, acotados al inicio de la reacción, se observa que los perfiles experimentales siguen la trayectoria de los perfiles de las simulaciones, lo cual indica que el sistema de control experimental cumple con las expectativas de control basado en redes neuronales.

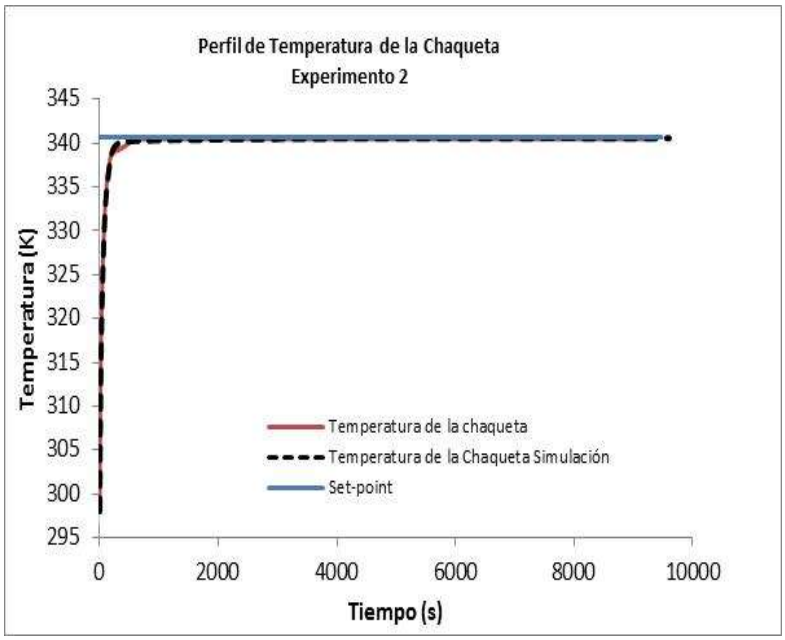


Figura 4.88 Perfil de Temperatura de la Chaqueta (Experimento 2)

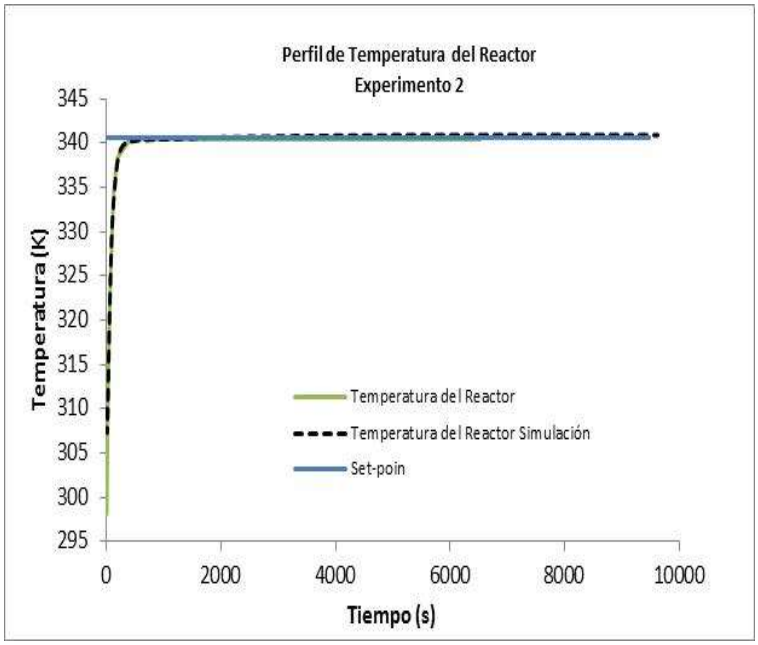


Figura 4.89 Pefil de Temperatura del Reactor (Experimento 2)

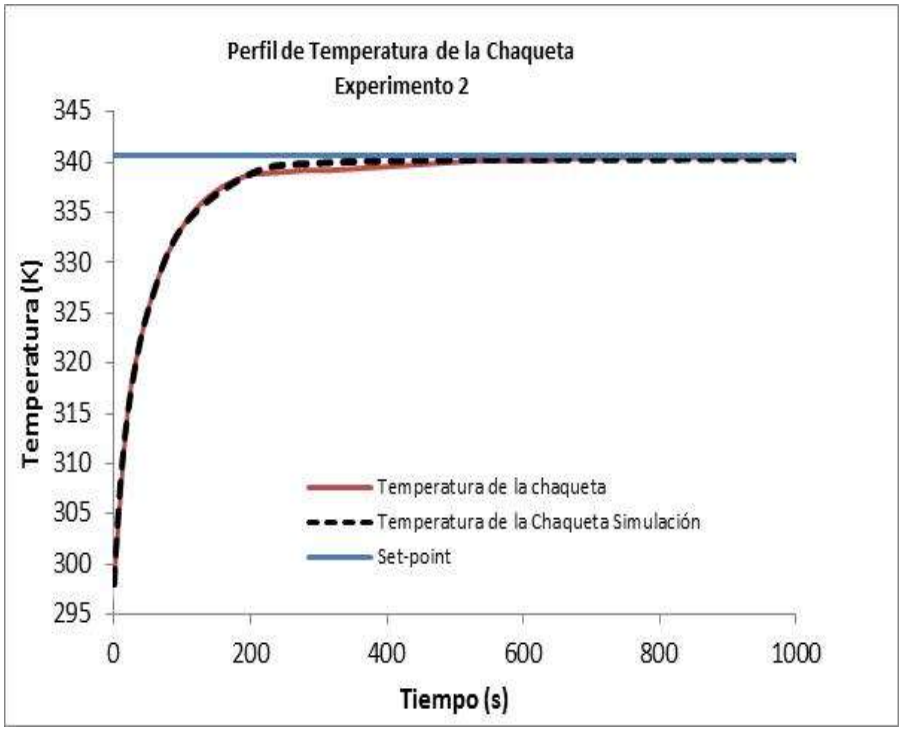


Figura 4.90 Perfil de Temperatura de la Chaqueta (Experimento 2) Ampliación

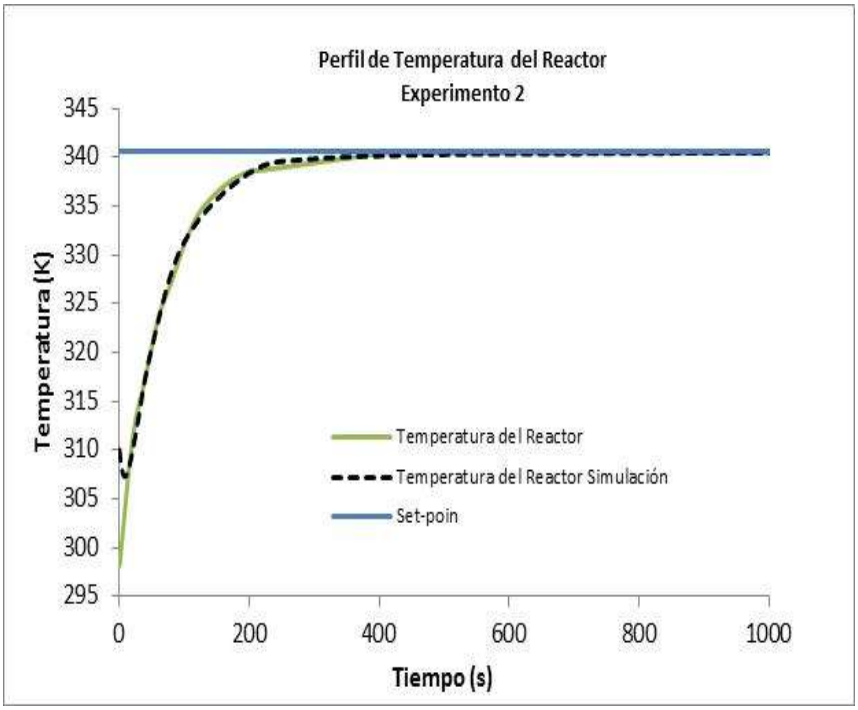


Figura 4.91 Perfil de Temperatura del Reactor (Experimento 2)

Ampliación

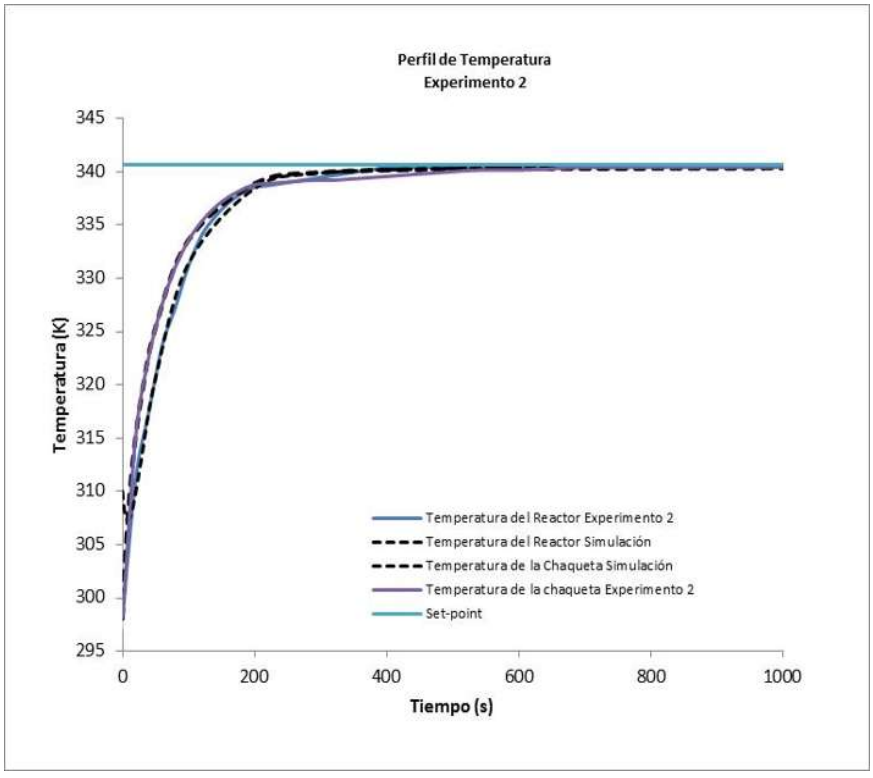


Figura 4.92 Perfiles de Temperatura Experimentales y Teóricos

Experimento 3

Para el experimento 3 el sistema de control proporciona resultados semejantes a los mostrados en los dos experimentos anteriores, el perfil de temperatura de la chaqueta (Figura 4.93) y el perfil de temperatura del reactor (Figura 4.94), establece que el control experimental basado en redes neuronales evolutivas esta cumpliendo con el perfil establecido por el control mediante redes neuronales basado en el modelo matematico. Los perfiles de temperatura del sistema reaccionante siguen presentando en un intervalo de tiempo una tendencia lineal, esto se comprueba al realizar la acotación de los ejes correspondiente a la variable tiempo (Figuras 4.95 - 4.96), esta tendencia lineal en los perfiles de temperatura confirman la teoría de cierta dependencia del sistema de calentamiento con la intensidad de corriente suministrada.

La Figura 4.97 muestra el conjunto de perfiles de temperatura y su comportamiento con respecto al tiempo, al observar la figura se puede determinar que los perfiles de temperatura experimentales con respecto a los teóricos muestran la tendencia lineal en un periodo corto de tiempo, lo cual se explica con anterioridad, también está figura puede establecer un punto de partida de comparación del sistema de control con respecto a los mostrados en los experimentos anteriores, lo que indica que los perfiles se cumplen.

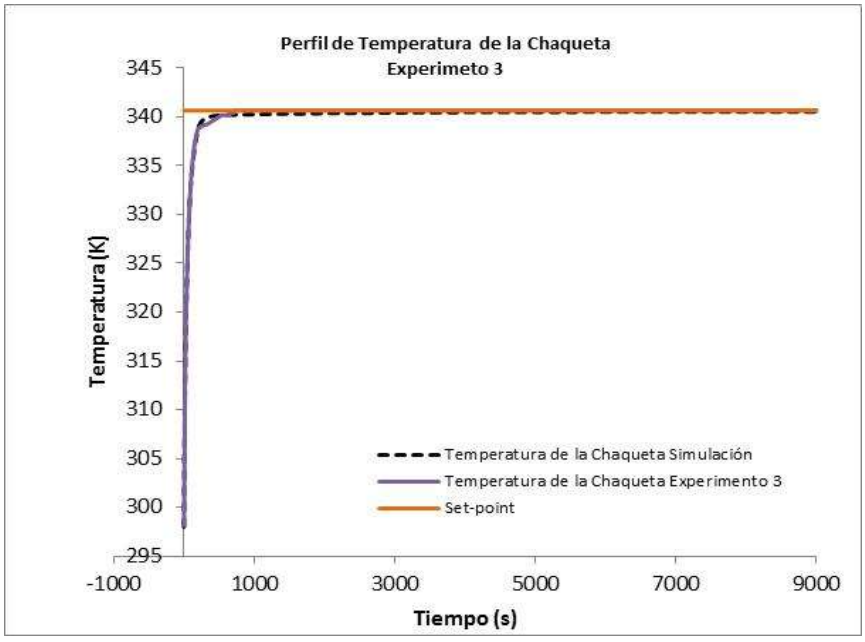


Figura 4.93 Perfil de Temperatura de la Chaqueta (Experimeto 3)

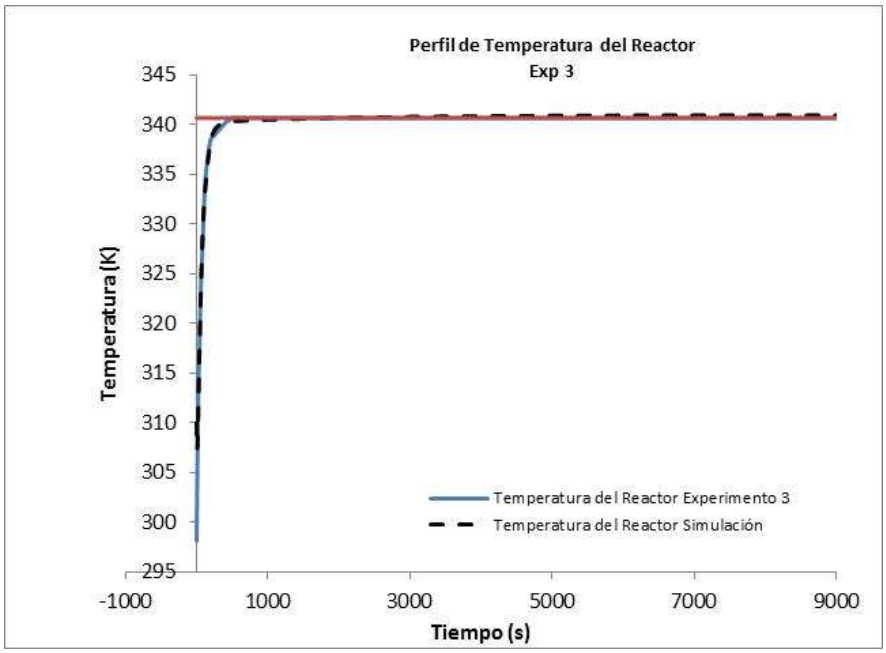


Figura 4.94 Perfil de Temperatura del Reactor (Experimento 3)

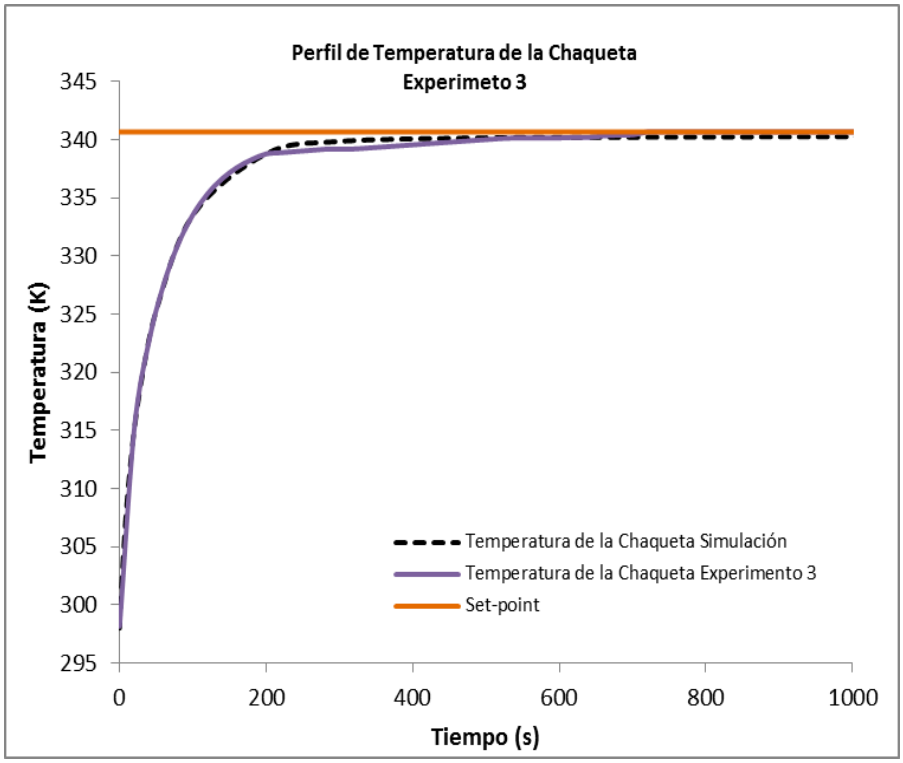


Figura 4.95 Perfil de Temperatura de la Chaqueta (Experimento 3) ampliación

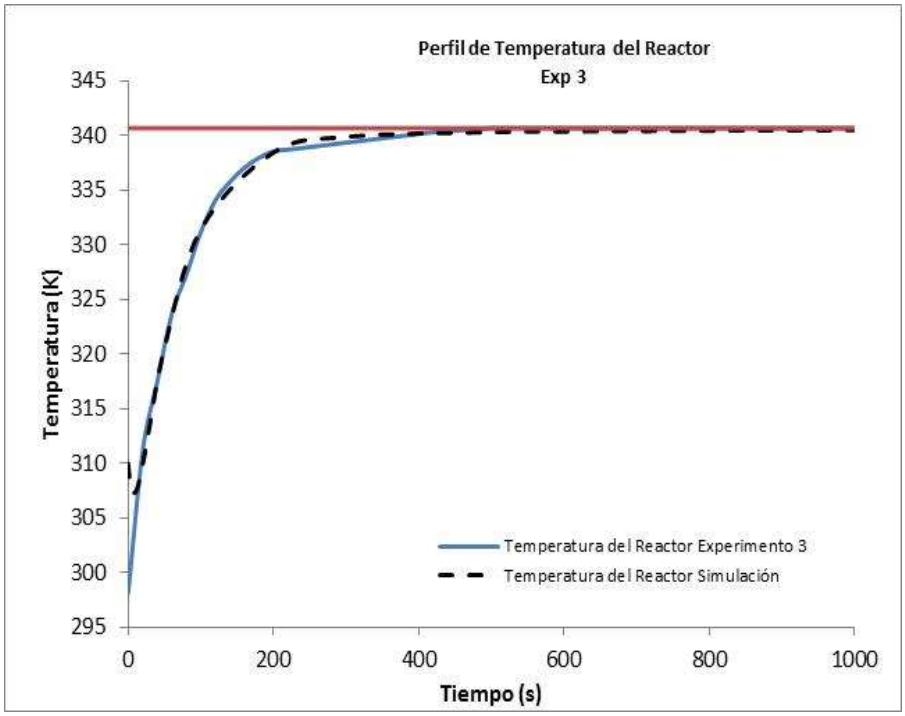


Figura 4.96 Perfil de Temperatura del Reactor (Experimento 3) Ampliación

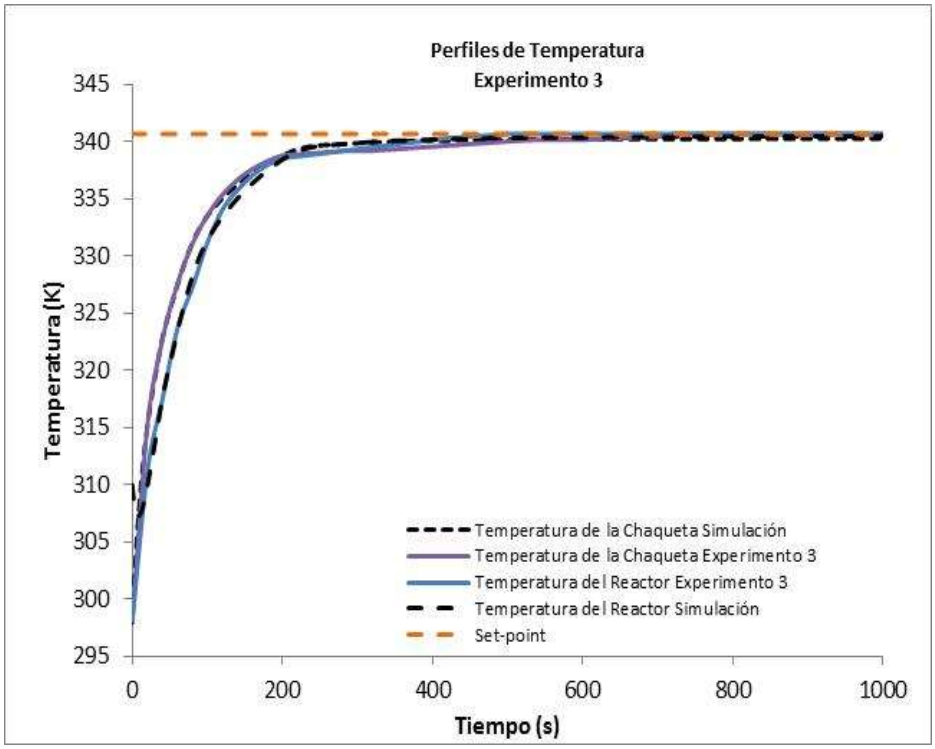


Figura 4.97 Perfiles de Temperatura Teóricas y Experimentales

4.2.2 Rendimiento experimental

Para cada uno de los experimentos se realizo el calculo del rendimiento, los resultados se presentan en la Tabla 4.4, los cuales son semejantes a los reportados en la literatura para este tipo de reacción con tiempos de reacción semejantes.

Tabla 4.4 Rendimiento de los experimentos

Experimento	% Rendimiento
1	**63.4**
2	**60.5**
3	**64.3**

4.2.3 Comprobación del Control Experimental con Perturbaciones

Para establecer el funcionamiento adecuado del sistema de control en forma experimental se establecen perturbaciones en el sistema de tal manera que el control tenga una acción conforme a lo establecido en el set-point del control. Los sistemas de reacción de tipo batch (intermitente) son altamente no lineales lo que involucra que su control sea complicado, pero esta características no los exenta de presentar perturbaciones que puedan modificar las acciones de control, al introducir perturbaciones al sistema de reacción o de calentamiento indica que el sistema de control planteado tiende a presentar una robustez con respecto a sus acciones.

La Figura 4.98 muestra el resultado de la simulación de la introducción de la perturbación al sistema de control de forma experimental, se observa la modificación del set-point con respecto al tiempo y la acción del control realizada, lo que implica que el control evoluciona con respecto a la dinámica del sistema a controlar, cabe resaltar que solo presenta en un intervalo definido de tiempo esto debido a cuestiones prácticas, para demostrar la acción del control de forma explícita.

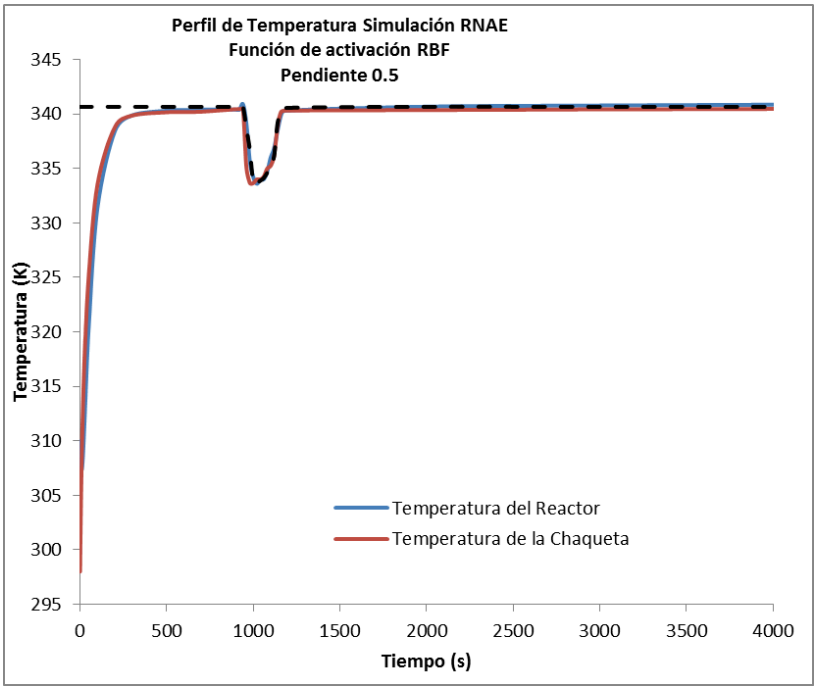


Figura 4.98 Perfil de Temperatura Teórico con Perturbación

La reproducibilidad de las trayectorias de control implica que la acción de control de forma experimental se llevan a cabo de forma correcta, para este trabajo se realizan tres experimentos en donde se implementa una perturbación al sistema de calentamiento que consiste en el decremento de la temperatura en el orden de magnitud de 5 K de la temperatura del set-point, dado el complimiento de la trayectoria de control implica que el sistema funciona de forma correcta ajustando sus parámetros a la dinámica del proceso.

Experimento 1 con perturbación

El experimento 1 muestra que la trayectoria de control con respecto al tiempo se cumple de acuerdo con la obtenida en la simulación de control con perturbación (Figura 4.99), durante la obtención del perfil de temperatura con la perturbación se observa que existe un incremento en la temperatura del reactor la cual desaparece en intervalo corto de tiempo, este incremento de la temperatura del reactor puede tener una explicación física, puede estar basada principalmente en la naturaleza de la reacción o simplemente a la dinámica del sistema de reacción.

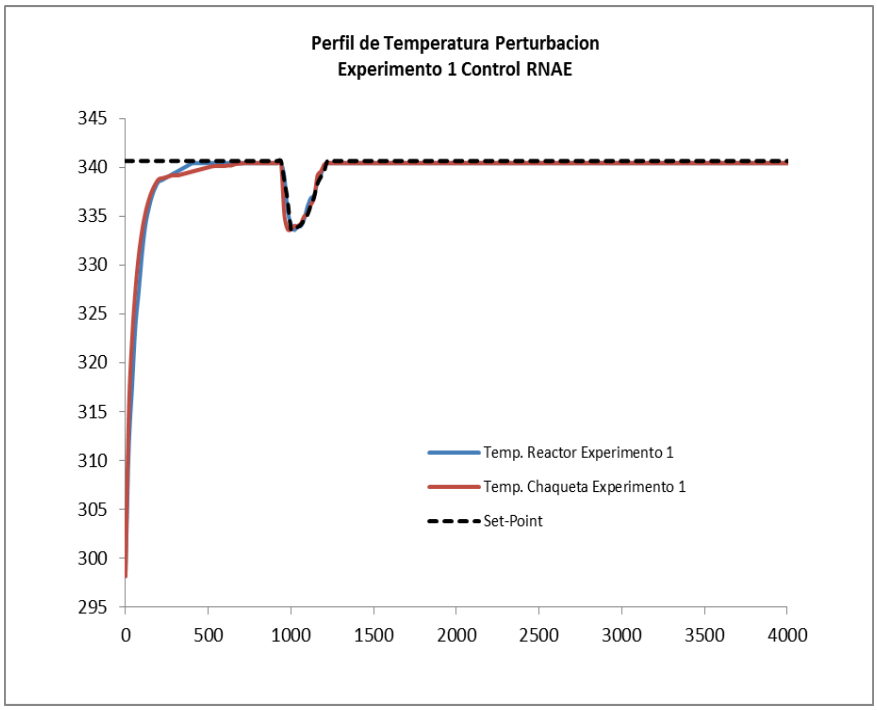


Figura 4.99 Perfil de Temperatura Reactor – Chaqueta (Experimento 1 con Perturbación)

Experimento 2 con Perturbaciones

La reproducibilidad de las trayectorias del control se puede comprobar con el experimento 2, se observa que los perfiles de temperatura del reactor y chaqueta (Figura 4.100) siguen de forma secuencial la trayectoria sugerida por el set-point, pero de igual forma que para el experimento 1 se continua presentando un incremento de la temperatura durante la rampa de calentamiento del reactor, lo cual se puede relacionar con la dinámica del sistema de reacción, la otra razón de este incremento que aunque es insignificante se presenta puede estar relacionado con el entrenamiento de la red neuronal que cabe resaltar se realiza en tiempo real y de forma dinámica al sistema de control.

Experimento 3 con perturbaciones

El experimento 3 demuestra de forma clara el comportamiento del sistema de control neuronal evolutivo experimental (Figura 4.101), además presenta de forma similar la misma perturbación en la temperatura del reactor, teniendo la misma acción de control que se presenta para los experimentos anteriores.

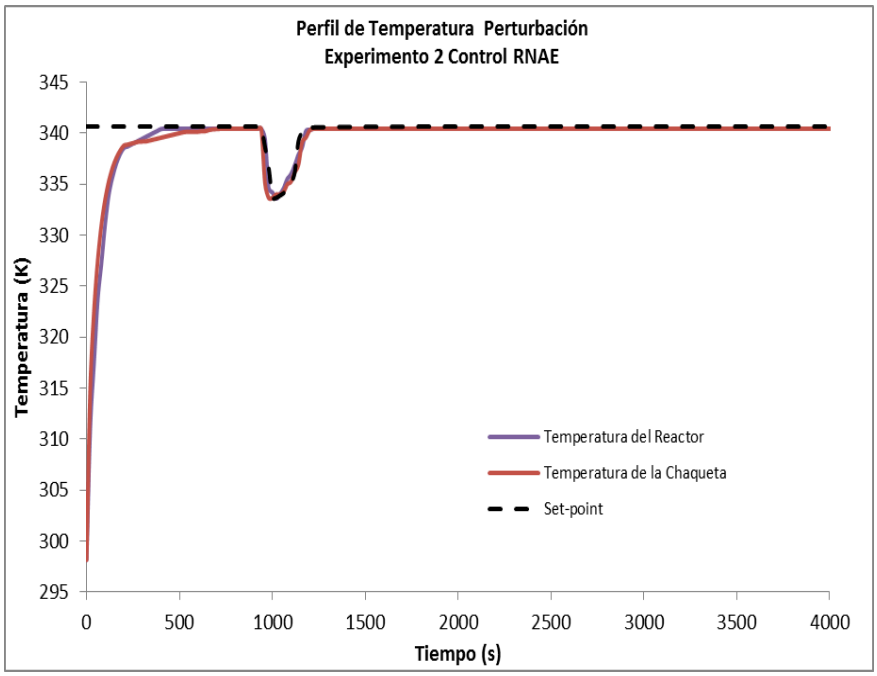


Figura 4.100 Perfil de Temperatura Reactor – Chaqueta (Experimento 2 con perturbación)

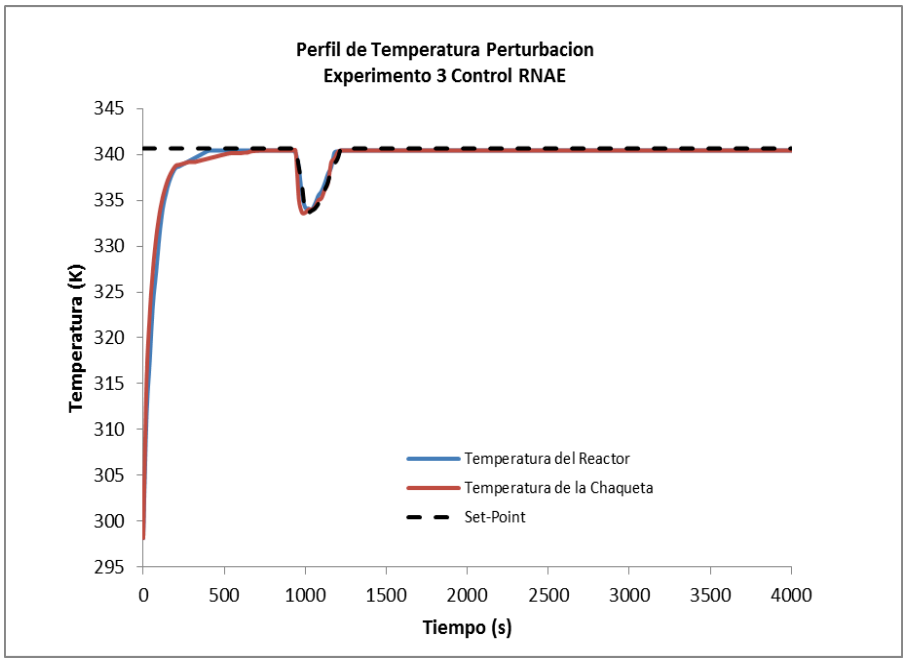


Figura 4.101 Perfil de Temperatura Reactor – Chaqueta (Experimento 3 con perturbación)

CONCLUSIONES Y RECOMENDACIONES

A continuación se presentan las conclusiones obtenidas del trabajo de investigación, así como los aportes realizados al área de ingeniería de procesos y control de procesos.

- La implementación de técnicas de control basadas en inteligencia artificial proporcionan una nueva vertiente del control de procesos químicos, específicamente en procesos tipo batch que además presenten un desprendimiento de calor (reacciones exotérmicas), estas técnicas pueden establecer un control robusto con capacidad de ajuste a la dinámica del proceso.
- La combinación de redes neuronales y estrategias evolutivas lo que da como resultado a las redes neuronales evolutivas (RNAE) en el control de procesos batch, proporcionan excelentes resultados con respecto a los obtenidos por técnicas de control tradicionales, tal es el caso del control *PID por sus siglas en inglés* y el control de tipo *GMC por sus siglas en inglés*, los cuales ofrecen un amplio espectro de aplicación, pero con la limitante de estar diseñados para sistemas continuos y su ajuste para procesos batch requiere de realizar una aproximación del comportamiento del modelo a un sistema continuo, lo cual provoca que el sistema de control presente puntos de inestabilidad además de generar un excedente de trabajo en la sintonización del controlador.
- Los problemas de sintonización del controlador son reducidos con la aplicación del control basado en redes neuronales evolutivas, pero esta técnica de control también presenta limitantes con respecto al sobre aprendizaje, lo cual puede provocar que el sistema genere un excelente control o simplemente no cumpla con la trayectoria de control, en este trabajo se muestra que el sobre aprendizaje está vinculado con los estados evolutivos de la red neuronal, además la función de excitación desempeña un papel importante en la disminución del sobre aprendizaje de la red

neuronal evolutiva, teniendo control sobre estas variables el control basado en redes neuronales evolutivas es una excelente técnica de control debido a su rápido ajuste a la dinámica del proceso, lo cual no sucede con los controladores de tipo convencional que requieren un ajuste de acuerdo a la dinámica que presente el proceso.

- A diferencia del proceso biológico neuronal, las redes neuronales artificiales no requieren de un gran número de neuronas para realizar un proceso en específico, es decir, si el sistema neuronal artificial presenta un exceso de neuronas esto puede provocar que el ajuste a la dinámica del proceso sea tardado, y presentar sobre aprendizaje lo cual provocaría que no se cumpla con el control de la variable sensible.
- La presencia del sobre aprendizaje en las redes neuronales implica un excedente en el tiempo de computo, en este trabajo se muestra que las técnicas evolutivas implementadas en la red proporcionan resultados adecuados a la dinámica del proceso, incluyendo la disminución a la tendencia al sobre aprendizaje, este no se presenta debido principalmente a la arquitectura de la red neuronal y a los estados de evolución, los resultados presentados por red que utiliza una función de tipo radial con una pendiente de 0.5, con un número de salidas de 3, presenta únicamente 27 estados de evolución, lo cual es aproximadamente 27 neuronas internas en la capa oculta, esto es un número mínimo de neuronas que favorece a no presentar sobre aprendizaje.
- Las funciones tangencial y logarítmica generalmente proporcionan excelentes resultados en la aplicación en procesos de control, para este sistema en específico no se opta por esta funciones de excitación, esto debido principalmente a dos razones:
    - La presencia de un error integral cuadrático con magnitud mayor (0.003191, 0.002966 respectivamente), con respecto a la presentada por la red neuronal que utiliza un función de base radial (0.001887).
    - El número de estados de evolución, 1.0737×10^9 estados de evolución para ambas funciones de excitación, con respecto a 27

estados de evolución que presenta la función de base radial, lo cual no significa que un número grande de estados de evolución implica un buen control, los expertos en redes neuronales evolutivas indican que un sistema con una cantidad grande de neuronas en sus capas ocultas aumenta su probabilidad de presentar sobre aprendizaje, además de inducir a la red neuronal evolutiva a permanecer de forma estática, con respecto a las redes neuronales que evolucionan hasta un número óptimo de neuronas en la capa oculta las cuales se comportan de forma dinámica con respecto al proceso.

- La implementación de las redes neuronales evolutivas a un sistema experimental proporciona resultados de control aceptables con respecto al comportamiento de la trayectoria del control, además de ser una opción viable para proceso de dinámica poco predecible. El ajuste dinámico de los vectores de los pesos sinápticos de la red neuronal se obtienen a partir de mediciones de temperatura en tiempo real, lo que implica que la dinámica de la red neuronal evolutiva implementada en el reactor experimental se reajusta en tiempo real proporcionando un excelente control.
- Una de las limitantes que presento el sistema de control basado en redes neuronales de forma experimental, se debió a la perturbación relacionada con el suministro de energía eléctrica la cual es una variable incontrolable tanto para el sistema de calentamiento como para el sistema de adquisición de datos, aunque en este último se optó por introducir un amplificador que considerara una compensación ajustable a las perturbaciones presentes.
- El sistema reaccionante presenta además de la temperatura diferentes variables que deben controlarse, la agitación, pureza del monómero, solvente y cantidad de iniciador, aunque la temperatura es la variable con mayor influencia en la polimerización, las variables aquí mencionadas también deben tomarse en consideración para el buen funcionamiento del sistema de reacción. El control basado en redes neuronales evolutivas podría controlar estas variables en tiempo real pero habrá que diseñar la red neuronal para establecer los máximos y mínimos de operación.

Recomendaciones y trabajo a futuro

El presente trabajo puede ser un punto de partida para realizar investigaciones posteriores como:

- Sistemas híbridos de entrenamiento para redes neuronales evolutivas, es decir, la búsqueda de vectores de mejoramiento para el entrenamiento de sistemas que presente no linealidades.
- Combinación de redes neuronales evolutivas con lógica difusa.
- Creación de sistemas dinámicos de control mediante la combinación de redes neuronales evolutivas y control de tipo convencional como el PID.
- Implementación experimental de los sistemas de control dinámico, basados en redes neuronales evolutivas con lógica difusa o simplemente con un control de tipo PID.

BIBLIOGRAFIA

Angeline, P.J., Saunders, G.M. and Pollack, J.B., 1994. *"An evolutionary algorithm that constructs recurrent neural networks"*. IEEE Transactions on Neural Networks, 5 (1): 54-65.

Beasley, D., Bull, D. R., and Martin, R. R. 1993. *"An overview of genetic algorithms: Part 1, fundamentals"*. University Computing, 15(2):58-69.

Billmeyer F. G., 1973 *"Ciencia de los Polímeros"*, 2da Ed. Reverte, Barcelona.

Blickle, T. and Thiele, L. 1995. *"A comparison of selection schemes used in genetic algorithms".* Technical Report 11, Computer Engineering and Comunnication Network Lab (TIK), Gloria strasse 35, 8092 Zurich, Switzerland.

Bruce J. and Miikkulainnen R. 2001 "*Evolving Populations Of Expert Neural Networks"* Department of Computer Sciences, The University of Texas at Austin To appear in Proceedings of the Genetic and Evolutionary Computation Conference 251--257.

Casanovas J., Alemán C., 2002 *"Introducción a la Ciencia de los Materiales"* Cálamo Producciones Editoriales, SLU., Barcelona.

Cohen, S. Intrator, N. 2002, *"Forward and backward selection in regression hybrid network";* Third International Workshop on Multiple Classifier Systems.

Cybenko G. 1989. *"Continuos Value Neural Networks with two Hidden Layers are Sufficient"*, Math Control Signal & Systems, vol. 2, pp. 303-314.

Daosud et, al. 2005 *"Neural network inverse model based controller for the control of steel pickling process"*. Computers and Chemical Engineering, vol. 29, pp. 2110-2119.

Darwin, C. 1859. *"On the Origin of Species by Means of Natural Selection",* John Murray, London.

Dirion et al. 1996 *"Development of adaptive neural networks for flexible control batch process"*. Chemical Engeneering Journal, pp. 63-65.

Drakopoulos, 2005 *"Training neural networks whit heterogeneous data".* Neuronal Networks. Vol. 18. pp. 595-601.

Ekpo E. E., Mujtabal. M. 2008 *"Evaluation of neuronal networks-based based on controllers in batch polymerization of methyl methacrylate"* Neurocomputing, vol. 71 pp.1401-1412.

Faggin, 1991 *"VLSI implementation of neural networks"* F. Faggin International Joint Conference on Neural Networks. Seattle, WA, 1991

Fogel, D.B., 1994. *"An introduction to simulated evolutionary optimization"* IEEE Transactions on Neural Networks, 5: 3-14.

Fogel, L. J., Owens, A. J., and Walsh, M. J. 1966. *"Artificial Intelligence through Simulated Evolution".* John Wiley & Sons, New York.

Frean, M. 1990. *The upstart algorithm: A method for constructing and training feedforward neural networks.* Neural Computation, 2(2): 198-209.

Flory, L. 1953 *"Introduccion a la polimerización por radicales libres",* Ed. Lamusa.

Galvan I.M., Zaldivar J.M. 1992 *"Control in batch reactors"* . Chemical Engineering and Processing pp. 505-518.

Galvan I.M., Zaldivar J.M. 1999 *"Control in batch reactors using controllers PID hibrys"* . Chemical Engineering and Processing vol. 23.

García-Pedrajas, N., Hervás-Martínez, C. and Muñoz-Pérez, J., 2002. *"Multiobjetive cooperative coevolution of artificial neural networks".* Neural Networks, 15(10): 1255-1274.

Goldberg, D. E. 1989. *"Genetic Algorithms in Search, Optimization and Machine Learning"*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Hancock, P., 1992. *"Genetic algorithm and permutation problems: A comparison of recombination operators for neural net structure specification".* In: D.y.S. Whitley, J. D. (Editor), Proc. Int. Workshop of Combinations of Genetics Algorithms and Neural Networks (COGANN-92),. IEEE Computer Soc. Press., Los Alamitos, CA, pp. 108-122.

Harp, S.A. Samad, T., Guha, A. 1989 *"Toward the genetic synthesis of neural networks"* In: J.D. Schaffer (Editor), 3er, Int. Conf. Conf. Genetic Algorithms and Their Applications. Morgan Kaufmann, San Mateo, CA, pp. 360-369.

Hertz, J., Krong, A. Palmer, R. 1991 *"Introduction to the Theory of Neural Computation"* Reading MA, Addison - Wesley.

Hebb D. 1949. *"Training neuronal networks "* IEEE press. New York.

Hecht-Nielsen, 1987 *"Kolomogorov's mapping neural network existence"*

Hecht-Nielsen, 1991 "On the algebraic structure of feedforward network spaces" Advanced neural computers (R. Eckmiller, ed.), pp.129- 135 North-Holland, Amsterdam, Netherlands.

Helmult A. M and Schawager R. 2002. "*Differentiation of Neuron Types by Evolving Activation Functions Templates for Artificial Neuronal Networks"*. IEEE press

Hinton, G.E., McClelland, J.L., Rumelhart, D.E., 1986. *"Distributed representations"* In: D.E.a.M. Rumelhart, J.L. editors (Editor), Parallel Distributed Processing: Explorations in the Microstructure of Cognition. MIT Press, Cambridge, MA, pp. 77-109.

Holland, J. H. 1975. *"Adaptation in Natural and Artificial Systems"*. University of Michigan Press, Ann Arbor. Republished by the MIT press, pp. 62.

Hopfield, 1982 "*Neural Networks and phisical systems with emergent*",Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Hunt. K.J.,Sbarbaro D., Zbikowski R., y Gawthrop P.J., 1992 *."Neuronal Networks for Control Systems-A survey"*, Automatica, vol.28, no 6, pp. 1083-1112.

Janson, D.J. and Frenzel, J.F., 1993. *"Training product unit neural networks with genetic algorithms" I*EEE Expert, 8(5): 26-33.

Jong, K. A. D. 1975. *"An analysis of the behavior of a class of genetic adaptive systems".* PhD thesis, University of Michigan.

Katime I, 1994, *"Química Física Macromolecular*", Servicio Editorial de la UPV, Bilbao.

Katime I., 1994 "*Problemas de Química Física Macromolecular",* Servicio Editorial de la UPV, Bilbao.

Kinnebrock, W., 1994. "*Accelerating the standard backpropagation method using a genetic approach"* Neurocomputing, 6(5-6): 583-588.

Kirk-Othmer 1998 *" Enciclopedia de Tecnología Química",* Ed., Limusa, México

Kitano, H., 1990. *"Designing neural networks using genetic algorithms with graph generation system"* Complex Syst., 4(4): 461-476.

Knerr, S., Personnaz, L. and G., D., 1992. "*Handwritten digit recognition by neural networks with single-layer training" I*EEE Transactions on Neural Networks, 3: 962-968.

Koza, J. R. 1992. *"Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)".* The MIT Press.

Lang, K.J., Waibel, A.H. and E., H.G., 1990. *"A time-delay neural network architecture for isolated word recognition"* Neural Networks, 3(1): 33-43.

Luus, 1994 *"Optimal control of reactors by iterative dynamic programming".* Journal Process Control, vol. 4, pp. 218.

Luyben L. W., 1989 "*Process Modeling, Simulation and control for Chemical Engineers"* Ed. McGraw-Hill, New York.

Luyben L. W., 1998 *"Tuning Temperature Controllers on Open loop Unestable Reactors",* American Chemical Society, vol. 37, pp. 4322-4331

Ihme M., Marsden A. L., Pitsch H. 2008. *"Generation of optimal artificial neuronal networks using a pattern search algoritm: application to approximation of chemical systems",* MIT, Neuronal Computation, vol. 20 pp. 573-601

Miller, G.F., Todd, P.M. and Hedge, S.U., 1989 *"Designing neural networks using genetic algorithms"* In: J.D. Schaffer (Editor), Proc. 3er Int. Conf. Genetic Algorithms and Their Applications. Morgan Kaufmann, San Mateo, CA, pp. 379-384.

Morrison R. T., Boyd R. N., 1998, *"Química Orgánica",* Cap. 36, 5ª Edición, Fondo Addison Wesley Longman de México, México, (1998).

Muñoz Guerra S, 1997 *"Introducción a los Polímeros"*, Fundación Politécnica de Cataluña, Barcelona.

Odian G., 1970, *"Principles of Polymerization"*, Mc. Graw Hill inc., New York.

Ponce Cruz P. 2009 *"Introduction Intelligent System whit LabView"*, Springer-IEEE, ITESM, Campus Cd. De Mexico.

Parker J., Ihme M., Pitsch H. 2002, *"Universal Approximation using Basis Function network"*. MIT. Neuronal Computing, vol. 3, pp. 246-257.

Ramón y Cajal, 1911 *"Histología del sistema nervioso del hombre y de los vertebrados"* Santiago Ramón y Cajal Consejo Superior de Investigaciones Científicas Madrid.

Rechenberg,I. 1973. *"Evolutions strategie: Optimierung technischer SystemenachPrinzipien der biologischen Evolution"*. Frommann- Holzboog, Stuttgart.

Rosemblatt I. G. 2001. *"Introduction of methods in neuronal networks "* Neuronal Networks Computing. pp. 345-350.

Rumelhart et al., 1986a *"Learning representations by back-propagating errors"*, R.J. Willliams Nature (London), 323, pp.533-536, 1986

Rumelhart et al., 1986b *"Learning internal representations by error propagation"*, R.J. Willliams Parallel Distributing Processing: Explorations in the Microstructure of Cognition (D.E. Rumelhart, J.L. McClelland, eds), vol.1, chapter 8, Cambridge, MA: MIT Press, 1986

Sanz M. A., 2005. "*Redes Neuronales y Sistemas Difusos"* Ed. Alfaomega.

Schwefel H-P. 1965, *"Kybernetische Evolution also Strategie der experimentellenForschung in der Stromungstechnik,"* Diplomarbeit, TechnischeUniversitatBerlin.

Schwefel H-P. 1975, *"Evolutions strategie und numerische Optimierung,"* Dissertation, Technische Universitat Berlin.

Skinner, A.J. and Brougthon, J.Q., 1995. *"Neural networks in computational materials science: Training algorithms"* Modeling and Simulation in Materials Science and Engineering, 3(3): 371-390.

Smith-Corripio, 1996 *"Control Automático de Procesos"* Ed. Limusa. p. 277

Souto, M.C.P., Yamazaki, A. and Ludermir, T.B., 2002 *"Optimization of neural network weigths and architectures for odor recognition using simulated annealing"* Proceedings of the 2002 International Joint Conference on Neural Networks, pp. 547-552.

Thierens, D., 1996 "*Non-redundant genetic coding of neural networks"* IEEE Int. Conf. Evolutionary Computation, ICEC'96, pp. 571-575.

Tomassini, M. 1995. *"A survey of genetic algorithms. Annual"* Reviews of Computational Physics, III:87-118.

Wade L. G., 1993 "*Química Orgánica"* Cap. 26, 2ª Edición, Prentice-Hall Hispanoamericana S.A., México.

Werbos, 1988 ***"****Generalization of backpropagation with application to a recurrent gas model"*. Neural Networks, vol.1, pp.339-356.

White, D. and Ligomenides, P., 1993. "*GANNet: A genetic algorithm for optimizing topology and weights in neural networks design*" Int. Workshop Artificial Neural Networks (IWANN'93), Lecture Notes in Computer Science. Springer-Verlag, Berlin, Germany, pp. 322-327.

Whitley, D., Starkweather, T. and Bogart, C., 1990. *"Genetic's algorithms and neural networks: Optimizing connections and connectivity"* Parallel Computation, 14(3): 347-361.

Yao, X. and Liu, Y., 1997 "*A new evolutionary system for evolving artificial neural networks"* IEEE Transactions on Neural Networks, 8 (3): 694-713.

Yao, X., 1999 *"Evolving artificial neural network"* Proceedings of the IEEE, 9 (87): 1423-1447.