# DESARROLLO DE UN CONTROLADOR BASADO EN REDES NEURONALES PARA UN SISTEMA MULTIVARIABLE DE NIVEL Y CAUDAL

ANA MARIA BARRERA FERNÁNDEZ
LAURA CAMILA POLANCO CEDEÑO

UNIVERSIDAD SURCOLOMBIANA
FACULTAD DE INGENIERÍA
INGENIERÍA ELECTRÓNICA
NEIVA-HUILA
2015

# DESARROLLO DE UN CONTROLADOR BASADO EN REDES NEURONALES PARA UN SISTEMA MULTIVARIABLE DE NIVEL Y CAUDAL

ANA MARIA BARRERA FERNÁNDEZ
LAURA CAMILA POLANCO CEDEÑO

Proyecto de grado presentado como requisito para optar al título de:
INGENIERO ELECTRÓNICO

Director
FAIBER IGNACIO ROBAYO BETANCOURT
Docente Programa de Ingeniería Electrónica

UNIVERSIDAD SURCOLOMBIANA
FACULTAD DE INGENIERÍA
INGENIERÍA ELECTRÓNICA
NEIVA-HUILA
2015

Nota de aceptación:

_____

_____

_____

_____

_____

_____

_____

_____

Firma del Director del Proyecto

_____

Firma del Primer Jurado

_____

Firma del Segundo Jurado

Neiva, Agosto de 2015

A Dios, por darnos la sabiduría y la fortaleza requerida para realizar este trabajo y a nuestras familias, por brindarnos su apoyo de manera permanente e incondicional.

# AGRADECIMIENTOS

Agradecemos a la Universidad Surcolombiana por su labor educativa.

Al ingeniero Faiber Ignacio Robayo Betancourt, nuestro director de proyecto por su constante apoyo y acertada dirección.

A todo el cuerpo docente y administrativo del programa de ingeniería electrónica por brindarnos su acompañamiento en el transcurso de estos años.

Finalmente a nuestros amigos y compañeros de estudio que nos respaldaron incondicionalmente en toda esta etapa haciéndola más agradable.

CONTENIDO

LISTA DE CUADROS

Pág.

LISTA DE FIGURAS

# LISTA DE ANEXOS

# GLOSARIO

**Backpropagation:** (retropropagación), algoritmo de aprendizaje supervisado que se usa para entrenar redes neuronales artificiales.

**Matlab:** abreviatura de *Matrix Laboratory* (Laboratorio de matrices), programa de análisis numérico creado por *The Mathworks*, disponible para la plataforma Unix, Windows y Mac Os X.

**Neurona artificial:** unidad de procesamiento elemental de una red neuronal artificial, que modela el comportamiento de una neurona biológica.

**Red feed-forward:** consisten en una serie de capas, la primera tiene una conexión de la entrada de red, cada capa posterior esta interconectada con la anterior y la capa final produce la salida de la red.

**Simulink:** paquete de software que se ejecuta acompañado a Matlab® para modelar, simular y analizar sistemas dinámicos.

**Sistema multivariable:** son sistemas con varias entradas y salidas, en los que casi siempre una entrada afecta a varias salidas y recíprocamente una salida es afectada por varias entradas.

**Red neuronal artificial:** modelo matemático inspirado en sistemas biológicos, adaptados y simulados en computadoras convencionales.

**Toolbox:** caja de herramientas, por su traducción al inglés; conjunto de herramientas de software pertenecientes a una librería, útiles para generar aplicaciones de diversas índoles dentro de un programa superior (Matlab).

## RESUMEN

Este trabajo presenta el desarrollo de un control neuronal basado en el modelo inverso para un sistema hidráulico multivariable de nivel y caudal. El controlador es implementado en Matlab utilizando el *toolbox* de redes neuronales y *Simulink* como interfaz de monitoreo y de control. El rendimiento del controlador es evaluado mediante simulaciones y pruebas realizadas en tiempo real. También se realiza un análisis comparativo frente al desempeño de un controlador *fuzzy*, desarrollado en un proyecto de grado anterior para el mismo sistema.



**Palabras clave:** *Control Neuronal, sistema multivariable, modelo inverso, Matlab, Simulink.*



## ABSTRACT

This project presents the development of a neural control based on the inverse model for multivariable hydraulics system of level and flow. The controller is implemented in Matlab using the neural network toolbox and Simulink as interface monitoring and control. The controller performance is evaluated through simulations and tests in real time. Was also performed a comparative analysis against the performance of a fuzzy controller, developed in a graduation project previous for the same system.



**Keywords:** *Neural Control, multivariable system, inverse model, Matlab, Simulink.*

# INTRODUCCIÓN

En los últimos años, con el acelerado crecimiento de la industria, los procesos han alcanzado un alto grado de complejidad debido a su no linealidad y a que presentan más de un lazo de control, por lo que se ha despertado cierto interés en el uso de diferentes métodos inteligentes que sean capaces de ejercer controles más óptimos. Los sistemas por redes neuronales son una de las técnicas que desde los años 90 toman importancia por el intento de diseñar controladores inteligentes[1].

Las redes neuronales artificiales se caracterizan por su habilidad de aprender de los ejemplos en lugar de tener que programarse en un sentido convencional. Su uso posibilita que la dinámica de los sistemas complejos sea modelada y un control preciso sea logrado a través del entrenamiento, sin tener información a priori sobre los parámetros del sistema[2].

Por estas razones, en este trabajo se desarrolla el diseño e implementación de un controlador basado en redes neuronales para el sistema hidráulico MIMO (*Multiple Input Multiple Output*) de nivel y caudal de la universidad Surcolombiana. Para cumplir con este propósito, se realiza la evaluación de la interacción entre los lazos, de la cual se obtienen dos sistema SISO (*Simple Input Simple Output*) que se trabajan de forma independiente, haciendo uso de las herramientas con las que cuenta Matlab® se diseña e implementa un controlador para cada subsistema y se evalúa su desempeño frente un controlador *fuzzy* aplicado a este mismo sistema.

---

[1] Ponce, Cruz Pedro. Inteligencia Artificial con Aplicaciones a la Ingeniería. 2010. Alfaomega (Ed).

[2] Parker. Optimal algorithms for adaptive networks: Second order backpropagation, second order direct propagation and second order Hebbian learning. 1995 En: D.B. Parker IEEE 1st Int. Conf. on Neural Networks, vol.2, pp.593 600, San Diego, CA.

# 1. OBJETIVOS

## 1.1 OBJETIVO GENERAL

Desarrollar un controlador de nivel y caudal utilizando redes neuronales artificiales para el sistema de tanques del Laboratorio de Control Inteligente de la Universidad Surcolombiana.

## 1.2 OBJETIVOS ESPECÍFICOS

Desarrollar una interfaz en Matlab® que permita controlar, manipular y visualizar el comportamiento de las variables en tiempo real.

Obtener el modelo del sistema que será utilizado en la simulación haciendo uso del *System Identification Toolbox* de Matlab®.

Validar la respuesta del controlador mediante la relación de los resultados obtenidos en la simulación y en la planta real.

Realizar un análisis comparativo entre el desempeño del controlador por redes neuronales y el algoritmo de control por lógica difusa implementado con Microcontrolador Especial en una tesis anterior para el mismo sistema.

# 2. GENERALIDADES

## 2.1 REDES NEURONALES ARTIFICIALES

**2.1.1 Introducción.** Una característica que diferencia al ser humano del resto de los animales es su capacidad de razonamiento, esto ha motivado al hombre a estudiar el funcionamiento del cerebro para construir modelos que simulen su comportamiento.

**2.1.1.1 ¿Qué son las redes neuronales artificiales (RNA)?.** Existen varias definiciones para una red neuronal artificial, más comúnmente aceptada es la de Robert Hetch-Nielsen[3]:
"Una red neuronal es un sistema de computación que consta de un gran número de elementos simples, muy interconectados, que procesan la información respondiendo dinámicamente frente a unos estímulos externos".

Otra definición de una red neuronal vista como una máquina de adaptación es la de Simon Haykin[4]:

"Una red neuronal es un procesador masivamente paralelo distribuido que es propenso por naturaleza a almacenar conocimiento experimental y hacerlo disponible para su uso. Este mecanismo se parece al cerebro en dos aspectos:

El conocimiento es adquirido por la red a través de un proceso que se denomina aprendizaje.

El conocimiento se almacena mediante la modificación de la fuerza o peso sináptico de las distintas uniones entre neuronas".

**2.1.1.2 Beneficios de usar RNA.** El uso de redes neuronales proporciona las siguientes propiedades y capacidades[4]:

**No linealidad:** Una neurona artificial puede ser lineal o no lineal. Una red neuronal, compuesta de una interconexión de neuronas no lineales, es en sí misma no lineal; esta propiedad es importante ya que permite la solución de problemas no lineales.

[3] Pino, Raúl; Gómez, Alberto; De Abajo, Nicolás. Introducción a la Inteligencia Artificial: Sistemas Experto, Redes Neuronales Artificiales y Computación Evolutiva. Oviedo, España. 2001. Servicios de Publicaciones Universidad de Oviedo (Ed).
[4] Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

**Adaptabilidad:** Las redes neuronales tienen la capacidad adaptar sus parámetros para responder adecuadamente a cambios que se produzcan en el ambiente de trabajo. Cabe aclarar que la adaptabilidad no siempre conduce a la robustez, ya que si esta es muy grande el sistema puede responder a perturbaciones pequeñas degradando el rendimiento del sistema.

**Tolerancia a Fallos:** Una red neuronal, implementada en forma de hardware, tiene el potencial de ser inherentemente tolerante a fallos, en el sentido de que pueden fallar algunos de sus elementos de procesamiento pero la red continúa trabajando.

**Uniformidad de análisis y diseño:** las redes neuronales poseen universalidad como procesadores de información, esto quiere decir, que la misma notación es usada en todos los ámbitos relacionados con la aplicación de estas.

**Analogía neurobiológica:** El diseño de una red neuronal es una analogía con el cerebro, esto indica que el procesamiento en paralelo con tolerancia a fallos es posible físicamente, rápido y potente.

**2.1.2 Modelos neuronales.** Un modelo neuronal cuenta con tres elementos básicos[5]:

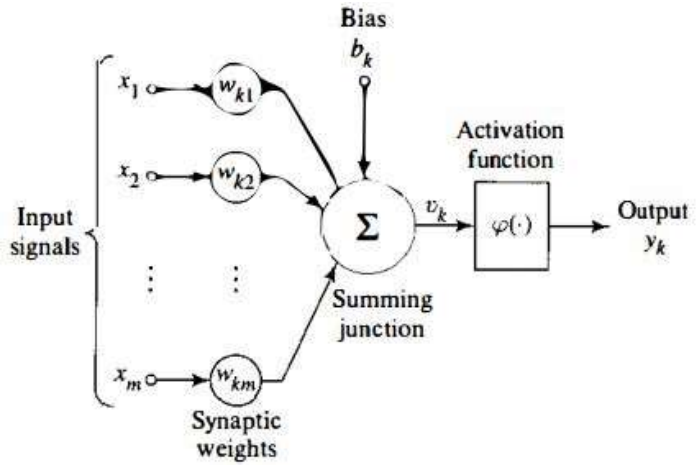
**Un conjunto de sinapsis o enlaces de conexión**, cada enlace se caracteriza por tener su propio peso o fuerza. Estos enlaces determinan el comportamiento de la neurona.

**Un sumador**, suma las señales de entrada, ponderadas por la respectiva sinopsis de la neurona.

**Una función de activación**, limita la amplitud de salida de una neurona.

[5] Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

**Figura 1. Modelo de una neurona**




Fuente. Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

El modelo de una neurona (*Figura 1*) también incluye una entrada externa llamada *bias* (bk), esta entrada aumenta y disminuye la entrada de la función de activación, dependiendo de si es positivo o negativo, respectivamente.

En términos matemáticos una neurona artificial se puede representar de la siguiente forma:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \qquad \textit{Ecuación 1.}$$

$$y_k = \varphi(u_k + b_k) \qquad \textit{Ecuación 2.}$$

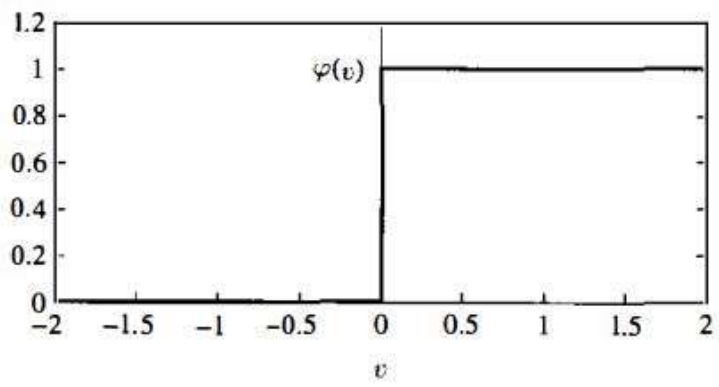
Existen tres tipos básicos de **funciones de activación:**

**Función Umbral:**

Una neurona que usa esta función se conoce como modelo de McCulloch-Pitts.

$$\varphi(v) = \begin{cases} 1 & if\ v \geq 0 \\ 0 & if\ v < 0 \end{cases} \qquad \textit{Ecuación 3.}$$

**Figura 2. Función umbral**



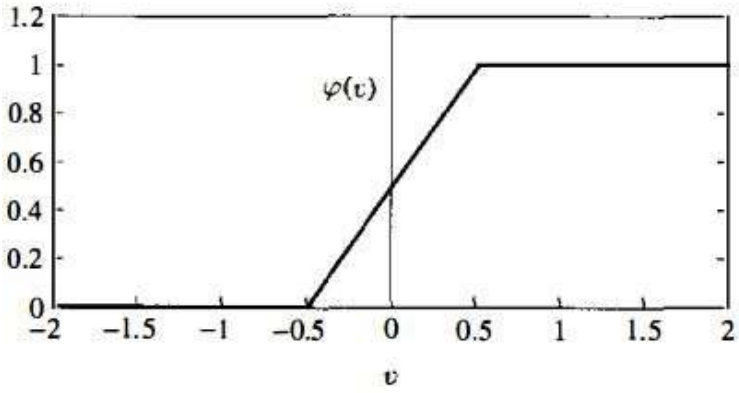

Fuente. Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

## Función lineal a tramos

$$\varphi(v) = \begin{cases} 1, & v \geq +\frac{1}{2} \\ v, & +\frac{1}{2} > v > -\frac{1}{2} \\ 0, & v \leq -\frac{1}{2} \end{cases}$$

*Ecuación 4.*

**Figura 3. Función lineal a tramos**



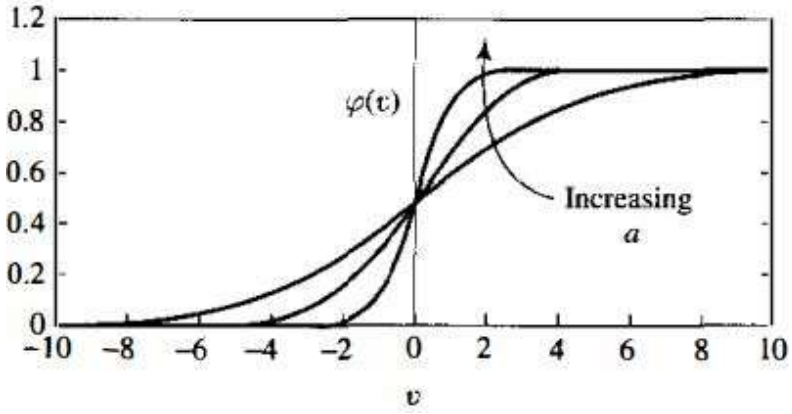

Fuente. Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

**Función sigmoidal**

$$\boldsymbol{\varphi(v) = \frac{1}{1 + exp(-av)}}$$

*Ecuación 5.*

**Figura 4. Función sigmoidal**




Fuente. Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

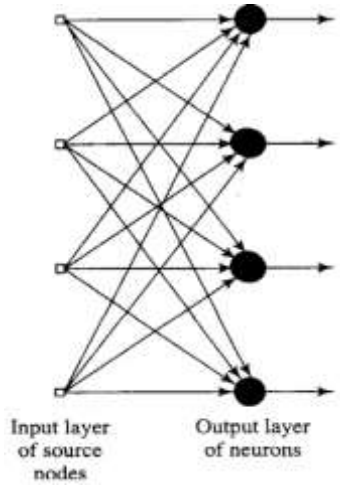
Es la función de activación más comúnmente usada, es una función continua no lineal con un rango de valores entre 0 y 1.

**2.1.3 Arquitecturas Neuronales.** La forma en la que están interconectadas las neuronas en una red neuronal, es lo que se conoce como arquitecturas neuronales. Existen tres clases fundamentales[6]:

**Redes neuronales monocapa**: Es la forma más simple de una red neuronal, consta de una capa de entrada y una capa de salida, en esta última capa es en la que se realizan los cálculos.

[6] Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

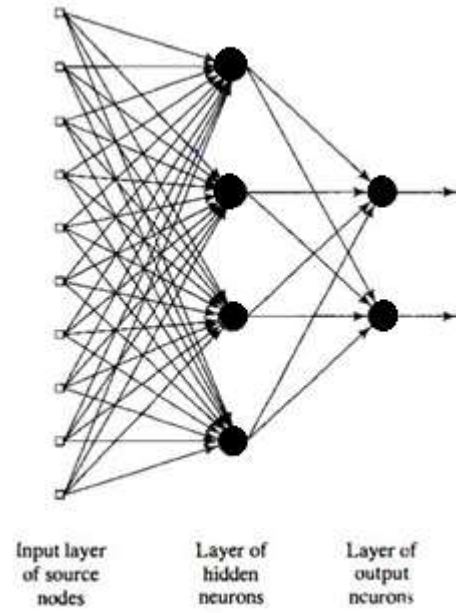
**Figura 5. Red monocapa**




Fuente. Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

**Redes neuronales multicapa:** se caracteriza por tener una o más capas ocultas, estas capas se encuentran entre la capa de entrada y la de salida.

**Figura 6. Red multicapa**

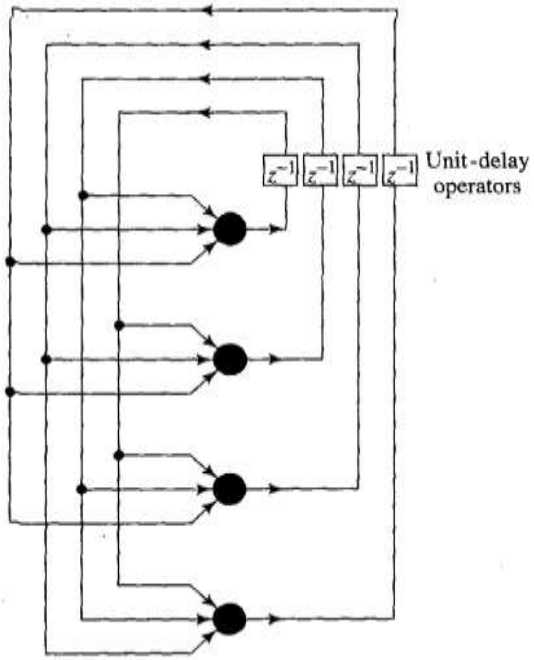



Fuente. Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

**Redes neuronales recurrentes:** Esta red se caracteriza porque tiene al menos un bucle de realimentación, estos bucles tienen un gran impacto en la capacidad de

aprendizaje y rendimiento de la red. Los bucles de realimentación implican el uso de unidades de retardo ($Z^{-1}$).

**Figura 7. Recurrente monocapa**




Fuente. Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

**2.1.4 Métodos de aprendizaje.** Los algoritmos de aprendizaje se pueden dividir en dos categorías[7]

**Figura 8. Métodos de aprendizaje**


Fuente. Rojas, Raul. Neural Network: A Systematic Introduction. 1996. Springer (Ed).

[7] Rojas, Raul. Neural Network: A Systematic Introduction. 1996. Springer (Ed).

**2.1.4.1 Aprendizaje supervisado**. En este método se proporciona a la red una señal de entrada, la salida entregada por la red es analizada y se corrigen los pesos de acuerdo a la magnitud del error. Este tipo de aprendizaje también se llama aprendizaje con un maestro.

El aprendizaje supervisado se divide en dos métodos, aprendizaje por refuerzo y por corrección de errores. El **aprendizaje por refuerzo** se utiliza cuando solo se sabe si la red entrega el resultado deseado o no y en el **aprendizaje con corrección de errores** las correcciones de los pesos son determinadas por la magnitud del error y el vector de entrada.

**2.1.4.2 Aprendizaje no supervisado**. En el aprendizaje no supervisado no se conoce la salida que debe entregar la red para una entrada dada, la red debe organizarse de tal forma que sea capaz de descubrir por sí sola relaciones de interés, como patrones, correlaciones o categorías en los datos de entrada.

**2.1.5 Control neuronal con modelo inverso.** Aunque existen otros enfoques para el control de sistemas, se hace énfasis en el modelo inverso ya que es la estrategia básica de control neuronal[8].

El modelo inverso se puede usar de tres maneras para crear un controlador neuronal[9].

**2.1.5.1 Inverso sencillo.** Este sistema de control es el inverso del modelo del proceso, si el modelo es no lineal su inverso también lo será, por lo que puede ser implementado como una red neuronal. El conjunto de controlador y modelo forman una red feedforward (propagación hacia adelante), el entrenamiento es realizado minimizando la diferencia entre el estado de referencia y la salida de la red; los únicos parámetros que pueden ser modificados son los pesos y bias del controlador, los parámetros del modelo se mantienen constantes durante el entrenamiento. Esta metodología da buenos resultados donde el problema es sencillo y el objetivo es una función estática.

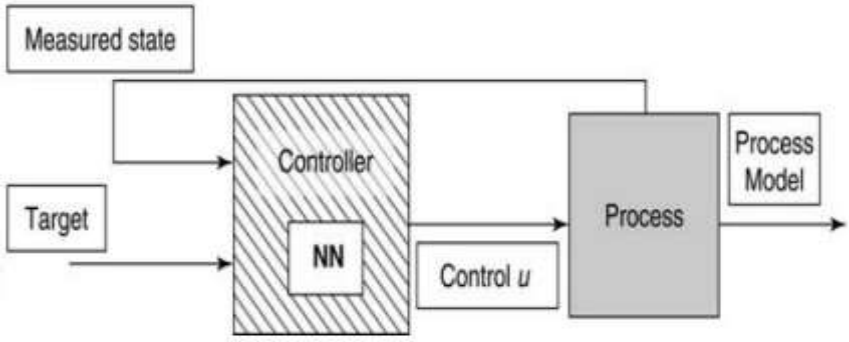
En palabras más sencillas, el control por modelo inverso básicamente busca cancelar la dinámica del sistema a controlar conectando la red neuronal en

[8] Norgaard, M. (2000) Neural Networks for Modelling and Control of Dynamic Systems. Springer – Verlag, London.

[9] Dreyfus, Gérard. Neural Networks: Methodology and Applications. 2005. Springer (Ed).

cascada al sistema, siendo ésta una aproximación matemática del inverso del sistema[10].

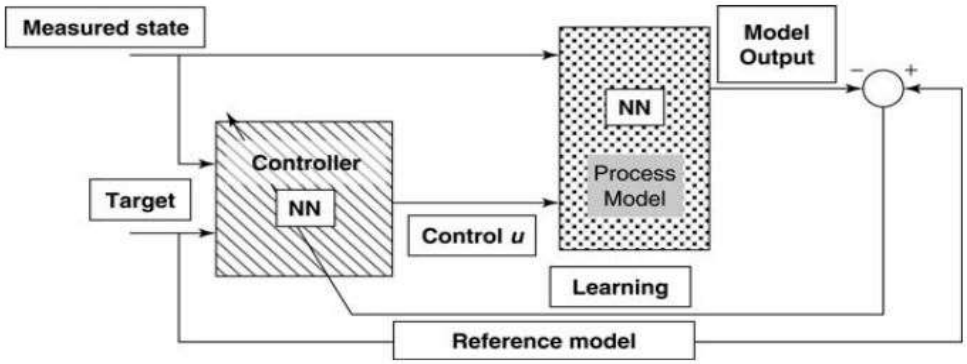
**Figura 9. Modelo inverso**




*Fuente. Dreyfus, Gérard. Neural Networks: Methodology and Applications. 2005. Springer (Ed).*

**2.1.5.2 Control adaptativo por modelo de referencia (MRAC).** En este método se utiliza el conocimiento previo del sistema, especialmente en las capacidades del actuador, para construir la ley de control. El método de modelo de referencia resulta ser útil en numerosas aplicaciones a problemas del mundo real. Se utiliza generalmente para mejorar el rendimiento de los sistemas dinámicos controlados por métodos clásicos.

**Figura 10. Control por modelo de referencia**

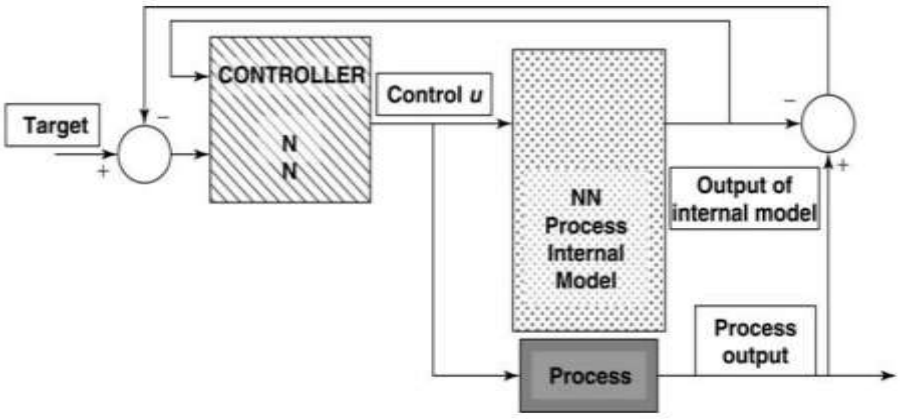



*Fuente. Dreyfus, Gérard. Neural Networks: Methodology and Applications. 2005. Springer (Ed).*

[10] Rodriguez, Victor. Garzón, Jaime. López, Jesús. Control Neuronal por Modelo Inverso de un Servosistema Usando Algoritmos de Aprendizaje Levenberg-Marquardt y Bayesiano. Disponible en web <http://arxiv.org/ftp/arxiv/papers/1111/1111.4267.pdf >

**2.1.5.3 Control basado en el modelo interno.** Este tipo de control involucra el modelo inverso y directo del sistema a controlar, es más robusto en cuanto a perturbaciones comparado con el modelo inverso simple, ya que el control se realiza con la realimentación de la salida que es la diferencia entre el modelo directo del sistema y el sistema real[11].

**Figura 11. Control por modelo interno**




*Fuente. Dreyfus, Gérard. Neural Networks: Methodology and Applications. 2005. Springer (Ed).*

**2.1.6 Redes neuronales y Matlab.** Matlab con su toolbox *Neural Network* facilita la labor de programación, debido a que posee comandos especializados para la creación y entrenamiento de redes neuronales artificiales. Se puede evidenciar el buen desempeño de controladores neuronales implementados en Matlab en [12] y [13]

**2.1.6.1 Crear una red neuronal.** Para la creación de la red neuronal Matlab posee un comando llamado *newff*, este crea una red *feed-forward backprogation*[14]. A esta función se le asignan cuatro parámetros de entrada, el primero es *P*, un vector formado por los valores máximos y mínimos de cada uno de los elementos

[11] Bermeo, Leonardo. Plataforma para la Implementación de Neuro-Controladores por Modelo Inverso de la Planta. 2004. Universidad del Valle. Cali, Colombia.

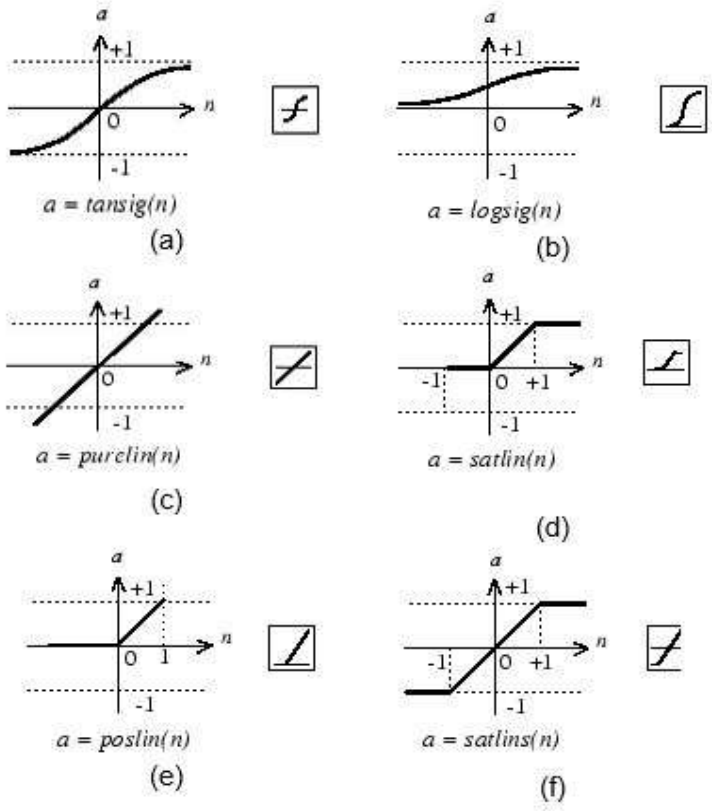
[12] Olivares, Victor; Urrea, Claudio; Cordoba, Félisa. Diseño e Implementación de un Simulador para el control de Posición de un Sistema a Grúa, Empleando Matlab-Simulink. 2014. Universidad de Santiago de Chile. Santiago de Chile, Chile. Disponible en web < http://www.researchgate.net/publication/270577126_Diseo_e_Implementacin_de_un_Simulador_para_el_Control_de_Posicin_de_un_Sistema_Gra_Empleando_MatLab-Simulink >.

[13] Escobar, Luisa; Montoya, Oscar; Giraldo, Didier. Control Global del Péndulo Furuta Empleando Redes Neuronales Artificiales y Realimentacion de Variables de Estado. Junio, 2013. Universidad tecnológica de Pereira. Pereira, Colombia. Disponible en web < http://www.scielo.org.co/scielo.php?pid=S0123-77992013000100005&script=sci_arttext>.

[14] Ayuda de Matlab®. Disponible en web < http://www.mathworks.com/help/>

del vector de entrada. El segundo parámetro es un arreglo que contiene el número de neuronas de cada capa, luego se asignan las funciones de activación de las neuronas de cada capa y el último parámetro es el nombre de la función de entrenamiento que se usará.
Las funciones de activación pueden ser, *tansig, logsig, purelin, satlin, poslin y satlins*. Ver *Figura 12*.

**Figura 12. Funciones de activación. (a)Función sigmoidal hiperbólica. (b)Función sigmoidal logarítmica. (c)Función lineal. (d)Función lineal saturada. (e)Función lineal positiva. (f)Función lineal simétrica saturada.**




*Fuente. Ayuda de Matlab. Disponible en web < http://www.mathworks.com/help/nnet/index.html>*

Las funciones de entrenamiento pueden ser: *trainlm, trainbfg, trainrp, traingd*, entre otras. Todas estas funciones actualizan los valores de los pesos y *bias* de acuerdo con el método o algoritmo que usen.

**2.1.6.2 Entrenar una red neuronal.** Una vez creada la red, se realiza el entrenamiento usando el comando *train*. Este comando toma como argumentos de entrada la red creada, las entradas y las salidas de la red, como argumentos de

salida se tienen: *net*, es una nueva red, *tr* hace referencia al registro de entrenamiento (épocas y desempeño), *Y* es la salida de la red y *E* el error.

## 2.2 SISTEMAS MULTIVARIABLES

En muchos casos, los sistemas MIMO (*Multiple-input Multiple-output*) presentan interacciones entre más de una variable manipulada y controlada, lo cual puede generar inconvenientes al momento de realizar el modelamiento y el control de un sistema. Por esta razón en sistemas multivariables es importante encontrar el grado de interacción que existe entre las variables de entrada y salida del proceso, para así determinar la forma de emparejarlas de tal manera que se disminuya al máximo la interacción que existe entre ellas. Esto permite que sistemas multivariable se reduzcan a *n* sistemas univariables, facilitando el proceso de modelamiento y control ya que estos se pueden realizar de manera independiente.

### Interacción en sistemas multivariables

Para determinar si hay o no interacción entre los lazos existen algunos métodos, como el de matriz ganancias relativas (RGA) propuesto por Bristol en 1996[15], el cual consiste en calcular las ganancias en lazo abierto y las ganancias relativas en régimen permanente para cada par de variables de entrada y salida del sistema.

### Ganancias en lazo abierto:

Las ganancias en lazo abierto para sistemas 2x2 son las siguientes

$$K_{11} = \frac{\Delta c1}{\Delta m1}\Big|_{m2} \qquad K_{12} = \frac{\Delta c1}{\Delta m2}\Big|_{m1}$$
$$K_{21} = \frac{\Delta c2}{\Delta m1}\Big|_{m2} \qquad K_{22} = \frac{\Delta c2}{\Delta m2}\Big|_{m1}$$

*Ecuaciones 6.*

[15] Bristol, E. H., On a New Measure of Interaction for Multivariable Process Control. IEEE Transaction on automatic control. January, 1996.

Donde $K_{ij}$ es la ganancia relativa que relaciona las variables controladas ($c_i$) con las variables manipuladas ($m_j$), realizando un cambio en cada variable manipulada mientras las otras permanecen constantes.

**Ganancias relativas, medida de interacción:**

La medida de interacción propuesta por Bristol, es simplemente la relación entre la ganancia en lazo abierto sobre la ganancia en lazo cerrado. Esta medida es adimensional ya que está definida como la relación de dos ganancias con las mismas variables controladas y manipuladas[16].

$$\mu_{ij} = \frac{K_{ij}}{K'_{ij}} \qquad \textit{Ecuación 7.}$$

Donde $\mu_{ij}$ es la ganancia relativa para el par $c_i - m_j$.

Para un sistema 2x2 las ganancias relativas son:

$$\mu_{11} = \frac{K_{11}}{K'_{11}} \qquad \mu_{12} = \frac{K_{12}}{K'_{12}}$$
$$\mu_{21} = \frac{K_{21}}{K'_{21}} \qquad \mu_{22} = \frac{K_{22}}{K'_{22}} \qquad \textit{Ecuaciones 8.}$$

Donde $\mu_{ij}$ es la ganancia relativa, $K_{ij}$ es la ganancia en lazo abierto y $K'_{ij}$ es la ganancia en lazo cerrado.

Las ganancias relativas también pueden ser calculadas a partir de las ganancias en lazo abierto.

$$\mu_{11} = \frac{K_{11}K_{22}}{K_{11}K_{22} - K_{12}K_{21}} \qquad \mu_{12} = \frac{-K_{12}K_{21}}{K_{11}K_{22} - K_{12}K_{21}} \qquad \textit{Ecuaciones 9.}$$

[16] Smith, Carlos y Corripio, Armando. Principles and Practice of Automatic Process Control, John Wiley & Sons, New York, 1997.

$$\mu_{21} = \frac{-K_{12}K_{21}}{K_{11}K_{22} - K_{12}K_{21}} \qquad \mu_{22} = \frac{K_{11}K_{22}}{K_{11}K_{22} - K_{12}K_{21}}$$

Cuando ya se tienen todos los valores de ganancias, se forma la matriz de ganancias relativas (RGA) que se muestra a continuación.

**Figura 13. Matriz de ganancias relativas**

| | $m_1$ | $m_2$ |
|---|---|---|
| $c_1$ | $\mu_{11}$ | $\mu_{12}$ |
| $c_2$ | $\mu_{21}$ | $\mu_{22}$ |

Los elementos de esta matriz deben cumplir con las siguientes propiedades:

La suma de los elementos de cada columna debe ser 1.

La suma de los elementos de cada fila debe ser 1.

Todos los elementos que forman la matriz son adimensionales.

Para realizar el análisis de la matriz de ganancias relativas se deben tener en cuenta las siguientes consideraciones:

Si $U_{ij} = 1$ significa que no hay interacción entre los lazos, puesto que la ganancia de lazo abierto es la misma que la ganancia en lazo cerrado.

Si $U_{ij} = 0$ indica que $m_i$ no debe ser controlada por $c_j$.

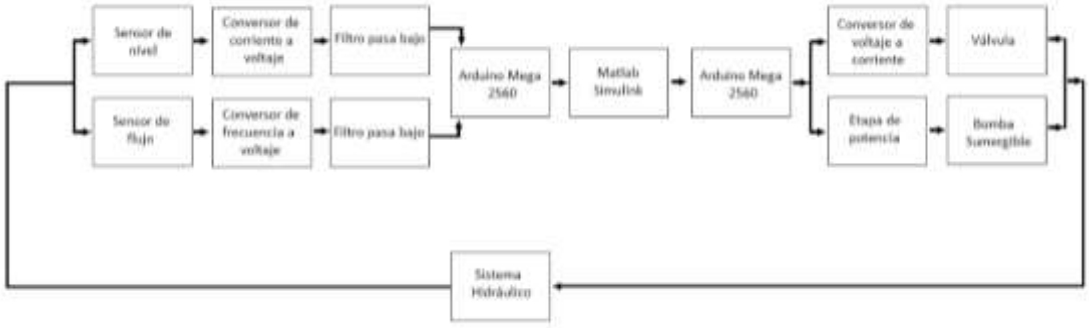
Si $U_{ij} = \infty$ no es posible el control en el lazo, ya que $c_j$ se ve muy poco o nada afectada por $m_i$.

Con base en las consideraciones anteriores se puede concluir que los pares de variables controladas-manipuladas que tengan la ganancia relativa con el valor más cercano a uno son las que deben ser emparejadas.

# 3. RECURSOS DE HARDWARE

El diseño del hardware usado para la alimentación y el acondicionamiento de señales ha sido tomado del proyecto “Diseño e Implementación de un Controlador para un Sistema Multivariable de Nivel y Flujo Utilizando Microcontrolador Especial para Lógica Difusa”[17] con la diferencia de que no se usa un Microcontrolador, puesto que el control se realiza directamente desde el software Matlab® y la comunicación y adquisición de datos se hace a través de la tarjeta de desarrollo Arduino Mega, además de que se incorpora un fijador de nivel de voltaje DC en la etapa de conversión de frecuencia a voltaje.

**Figura 14. Diagrama de bloques del sistema**




## 3.1 CIRCUITOS DE ALIMENTACIÓN

### Fuente dual de +/- 12V

Para la construcción de esta fuente se hizo uso de un transformador de 12+12 VAC a 2A y de los reguladores LM7812 y LM7912 para el voltaje positivo y negativo respectivamente. Esta fuente es utilizada para la polarización de los amplificadores LM358. La alimentación positiva se usa para polarizar el conversor de frecuencia a voltaje, LM2907.

[17] Ramirez G. Viviana, Calvache G. Oscar. “Diseño e Implementación de un Controlador Multivariable de Nivel y Flujo Utilizando Microcontroladores Especiales para Lógica Difusa”. Neiva, Colombia. 2013.

### Fuente de 24V

Se usa la fuente de referencia SP-320-24[18], que dispone de una potencia máxima de 312 W y entrega 24V a 13A. Esta es usada para la alimentación de la válvula de control y de perturbación.

### Alimentación de 5V

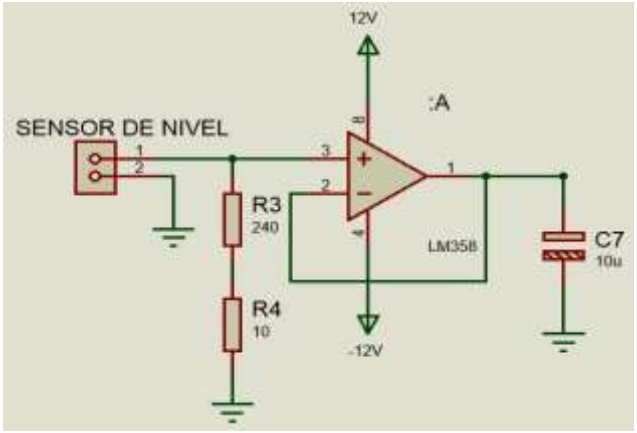
Se toman dos alimentaciones de 5V, una directamente desde el Arduino y la otra a través de la fuente de alimentación de 24V usando el regulador de 5V LM7805, las cuales se usan en la parte de la potencia de la bomba y la válvula.

## 3.2 CIRCUITOS DE ACONDICIONAMIENTO DE SEÑALES

### Conversor de Corriente a Voltaje.

Es el encargado de convertir la señal de corriente de 4mA a 20mA que entrega el sensor de nivel en una señal de voltaje de 0V a 5V, para ser leída por la entrada analógica A0 del Arduino Mega 2560. Este conversor se implementa con un amplificador operacional LM358 como seguidor.

**Figura 15. Circuito conversor de corriente a voltaje**




Para el cálculo de la resistencia se tiene en cuenta que el valor de voltaje máximo soportado por el Arduino es de 5V y la corriente máxima entregada por el sensor es de 20 mA.

[18] 320W Sigle Output With PFC Function SP 320 Series. Mean Well. Disponible en web <http://www.meanwell.com/search/SP-320/SP-320-spec.pdf>

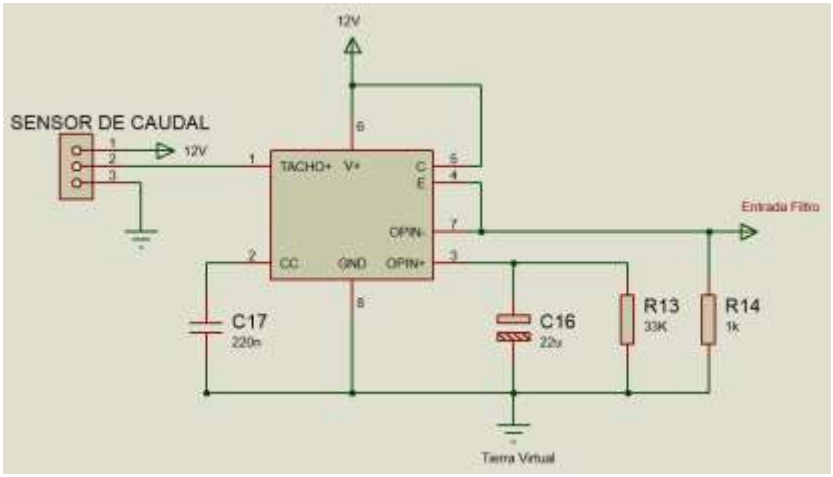
$$R = \frac{V}{I} = \frac{5v}{20mA} = 250\Omega$$ *Ecuación 10.*

El valor de la resistencia fue implementado con R3 y R4 (*Figura 15.*)

## Conversor de Frecuencia a Voltaje.

Toma la señal de salida del sensor de flujo y se convierte en una señal de voltaje entre 0V y 5V que será leída por la entrada analógica A1 del Arduino Mega 2560. El integrado LM2907 es el encargado de esta labor, la configuración de este es tomada de[19].

**Figura 16. Circuito conversor de frecuencia a voltaje**




Con base en la información suministrada por la hoja de datos del integrado se calculan los valores de las resistencias R13, R14 y capacitores C16, C17.

$$R13 \geq \frac{V_{max}}{I_{max}} = \frac{5V}{180\mu A} = 27.7K\Omega$$ *Ecuación 11.*

Con esta condición, se calcula el valor de R13, asumiendo que C17 tiene un valor de 0.22µF, de la siguiente manera

[19] Ramirez G. Viviana, Calvache G. Oscar. "Diseño e Implementación de un Controlador Multivariable de Nivel y Flujo Utilizando Microcontroladores Especiales para Lógica Difusa". Neiva, Colombia. 2013.

$$R13 = \frac{V_{max}}{C1 * V_{cc} * F_{max}} = \frac{5V}{0.22\mu F * 12V * 56Hz} = 33.82K\Omega$$

*Ecuación 12.*

$$R13 = 33K\Omega\ (v.comercial)$$

$$C16 = \frac{V_{cc}}{2} * \frac{C17}{V_{riz}} * \left(1 - \frac{V_{max}}{R_{13} * I_{max}}\right)$$

*Ecuación 13.*

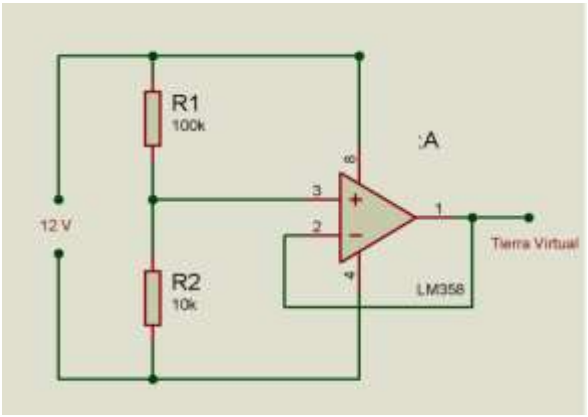
$$C16 = \frac{12V}{2} * \frac{0.22\mu F}{0.009V} * \left(1 - \frac{5V}{33K\Omega * 180\mu A}\right)$$

$$C16 = 23.2\mu A \approx 22\mu A$$

Para R14 se asume un valor que no afecte considerablemente el voltaje de salida, se le asigna el valor de 1KΩ.

Debido a que este integrado detecta solo los cruces por cero y la señal entregada por el sensor no cumple con esta condición, se hizo necesario fijar un nivel DC más alto que el de la tierra común, a continuación se muestra el circuito usado.

**Figura 17. Fijador de nivel de voltaje**



**Filtros pasa bajo.**

Son implementados para eliminar el ruido proveniente de los actuadores, la frecuencia de corte es de 0.2 Hz, lo cual permite que sólo pase la componente DC de cada señal.

El filtro fue diseñado con el software *Filter Wiz Pro v4* y con las siguientes especificaciones, orden del filtro: 4, frecuencia de corte: 0.2Hz.

**Conversor de Voltaje a Corriente.**

El integrado XTR110KP[20] es el dispositivo que permite realizar esta conversión, acepta entradas de 0 a 5V o 0 a 10V y se puede conectar a las salidas de 4 mA a 20 mA, 0 mA a 20 mA entre otros rangos. El rango escogido es de 0 a 5V y en la salida de 4mA a 20mA, para esto se usaron las siguientes conexiones.

**Cuadro 1. Conexión del XTR110KP**

| INPUT(V) | OUTPUT (mA) | PIN 3 | PIN 4 | PIN 5 | PIN 9 | PIN 10 |
|---|---|---|---|---|---|---|
| 0 a 5 | 4 a 20 | 10V Ref | Com | input | Com | open |

*Fuente. Precision Voltage to Current Converter-Transmitter XTR110KP. Texas Instrument, Texas, 2009. Disponible en web < http://www.ti.com/lit/ds/sbos141c/sbos141c.pdf>*

**Etapa de Potencia.**

El circuito que proporciona la potencia a la bomba se implementa basado en[21], pues el MTP3055 soporta corrientes hasta de 10 A, tiene una velocidad de trabajo buena, y el optoacoplador 4N25 provee un aislamiento entre la etapa de potencia y la etapa de control, lo cual brinda protección al Arduino y demás dispositivos electrónicos de baja potencia que se estén usando.

[20] Precision Voltage to Current Converter-Transmitter XTR110KP. Texas Instrument, Texas, 2009. Disponible en web < http://www.ti.com/lit/ds/sbos141c/sbos141c.pdf>

[21] Ramirez G. Viviana, Calvache G. Oscar. “Diseño e Implementación de un Controlador Multivariable de Nivel y Flujo Utilizando Microcontroladores Especiales para Lógica Difusa”. Neiva, Colombia. 2013.

## 3.3. SENSORES Y ACTUADORES

### Sensor de Nivel LIT25.

Este hace uso del ultrasonido para medir la variación de altura del líquido almacenado, la cual es convertida en una señal de corriente de 4 mA a 20 mA[22]. Se escoge este tipo de sensor para medir nivel debido a que su instalación se realiza fuera del líquido y al no entrar en contacto con este evita problemas de corrosión, además de que produce una medición continua y puntual.

### Sensor de Flujo Efecto Hall.

Detecta la cantidad de flujo de agua que circula por la tubería, su funcionamiento se basa en el efecto hall. La señal de salida está dada en frecuencia[23]. Se usa este tipo de sensor ya que su flexibilidad y durabilidad hace que sea idóneo para esta tarea, lo cual constituye una solución flexible a un bajo costo.

### Válvula.

Las válvulas usadas son de la marca DANFOSS referencia EV260B 15B, su funcionamiento se basa en regular progresivamente la corriente de la bobina para ajustar la válvula en cualquier posición entre completamente cerrada y completamente abierta[24].

Son usadas dos válvulas, una para el tanque de perturbación, la cual maneja un sistema de control ON/OFF y la otra para el tanque principal, esta opera de manera proporcional, la señal de control es suministrada por la salida PWM del Arduino.

Se realizaron algunas pruebas variando el voltaje aplicado a la válvula para observar su comportamiento. A continuación se muestran los resultados obtenidos.

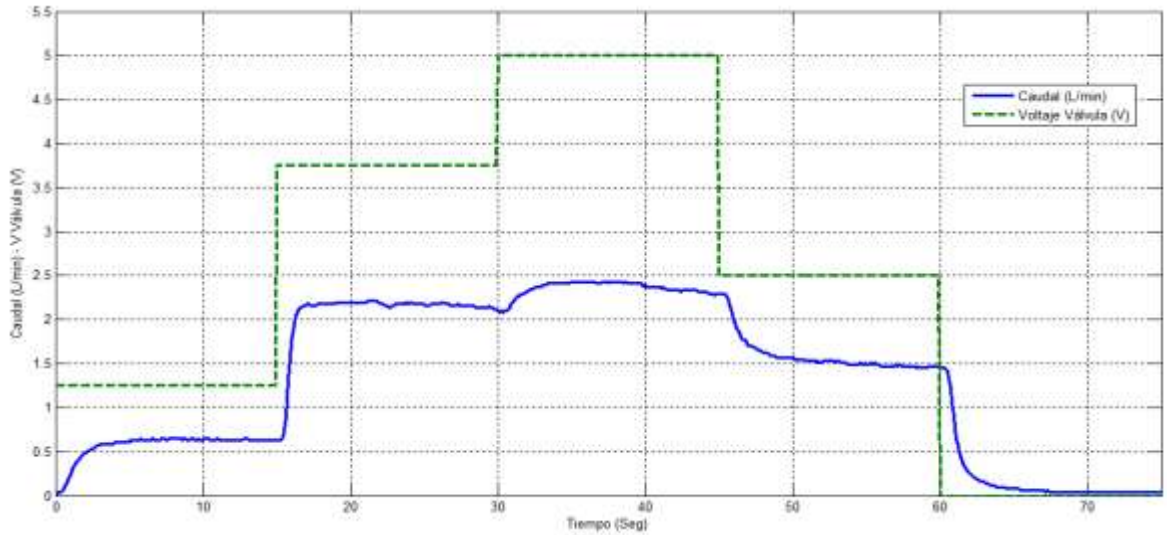
---

[22] LITE25 Level Indicating Transmitter Manual Series 3.5.1. Greyline Intrements Inc. Disponible en <http://www.greyline.com/pdf/LIT25%20Ultrasonic%20Manual%20Series%203.5.1.pdf>

[23] Water Flow Sensor HZ21WA. Disponible en <http://www.microelectronicos.com/datasheets/UCTS0058.pdf>

[24] Folleto Técnico Válvulas Solenoides Proporcionales Servoaccionadas De 2 Vías Tipo EV260B. Danfoss, 2014. Disponible en <http://www.ra.danfoss.com/TechnicalInfo/Literature/Manuals/04/IC.PD.200.O4.05.pdf>

**Figura 18. Comportamiento de la válvula**




## Bomba sumergible.

La bomba usada para suministrar el caudal de entrada al tanque principal es de la marca RULE 1100, esta se encuentra ubicada en el tanque de reserva. De igual forma que con la válvula, la señal de control es suministrada por una salida PWM del Arduino con voltajes de 0 a 5V, que aplicados a la etapa de potencia logra elevarlos a voltajes entre 0 a 24 V, lo cual le proporciona la suficiente potencia para su correcto funcionamiento. El cambio de voltaje aplicado a la bomba proporciona un cambio en el caudal de entrada, lo cual se observa en las pruebas realizadas que se consignan en la *Cuadro 2*.

**Cuadro 2. Comportamiento de la bomba**

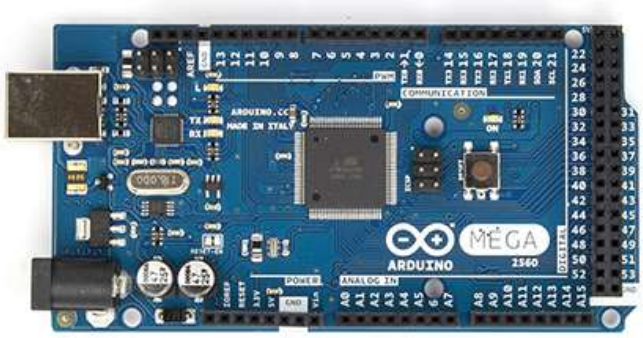
| PWM Bomba (V) | Caudal de entrada (L/min) |
|---|---|
| 1 | 1.27 |
| 1.4 | 3.67 |
| 1.5 | 4.94 |
| 2 | 9.89 |
| 3 | 14.27 |
| 4 | 16.25 |
| 5 | 19.8 |

## 3.4 TARJETA DE ADQUISICIÓN DE DATOS

### Arduino Mega 2560.

Es usada como tarjeta de adquisición de datos debido a que cumple con todos los requerimiento necesarios para este propósito, pues cuenta con 16 entradas analógicas con resolución de 10 bits, 15 salidas PWM con resolución de 8 bits y velocidad de trabajo de 16MHz. Además de presentar un bajo costo en comparación con otras tarjetas.

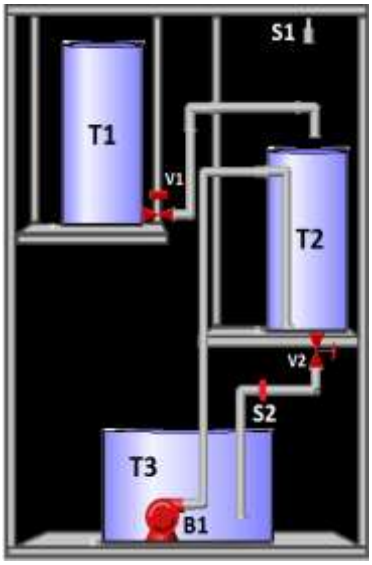
**Figura 19. Arduino Mega 2560**




*Fuente: http://arduino.cc/en/Main/ArduinoBoardMega2560*

# 4. DESCRIPCIÓN DEL PROCESO

**Figura 20. Sistema hidráulico**




Fuente. Ramirez G. Viviana, Calvache G. Oscar. “Diseño e Implementación de un Controlador Multivariable de Nivel y Flujo Utilizando Microcontroladores Especiales para Lógica Difusa”. Neiva, Colombia. 2013.

**Figura 21. Sistema hidráulico implementado**

En la *Figura 20* se muestra el esquema de la planta, la cual consta de tres tanques; tanque de perturbación de nivel T1, tanque principal T2, tanque de reserva T3, dos electroválvulas proporcionales; electroválvula de perturbación V1, electroválvula de control V2 y una bomba sumergible B1, un sensor de nivel S1 ubicado en la parte superior de T2 y un sensor de caudal S2 ubicado en la tubería inmediatamente después de V2.

Se pretende controlar el nivel y el caudal de salida de T2 accionando B1 y V2 que proporcionan el flujo de entrada y salida de líquido de T2 respectivamente, a través de V1 se ingresa un flujo de líquido proveniente de T1 con el cual se busca perturbar el proceso de control del nivel.

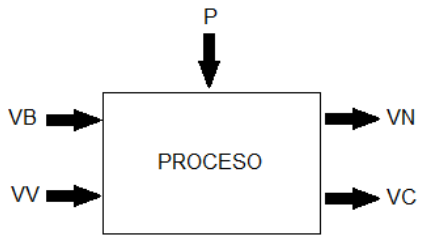
## 4.1 ENTRADAS Y SALIDAS DEL PROCESO

Se consideran como entradas del proceso el voltaje aplicado a la bomba sumergible B1 (VB) y el voltaje aplicado a la electroválvula V2 (VV).

Se consideran como salidas del proceso el voltaje entregado por el sensor de nivel (VN) y el voltaje entregado por el sensor de caudal (VC)

Se considera como perturbación (P).

**Figura 22 Diagrama del proceso**




## 4.2 EVALUACIÓN DE LA INTERACCIÓN DE LAS VARIABLES

Para determinar las ganancias en lazo abierto y las ganancias relativas en régimen permanente del sistema, se realiza una prueba experimental que consiste en aplicar un escalón constante durante todo el tiempo de la prueba a una variable de entrada, mientras que la otra variable es sometida a un cambio de escalón en el tiempo $t_0$ y así observar como es el comportamiento de las variables de salida, ver *Figura 23* y *Figura 24*.

**Figura 23. Respuesta del proceso ante un cambio de escalón en VB**

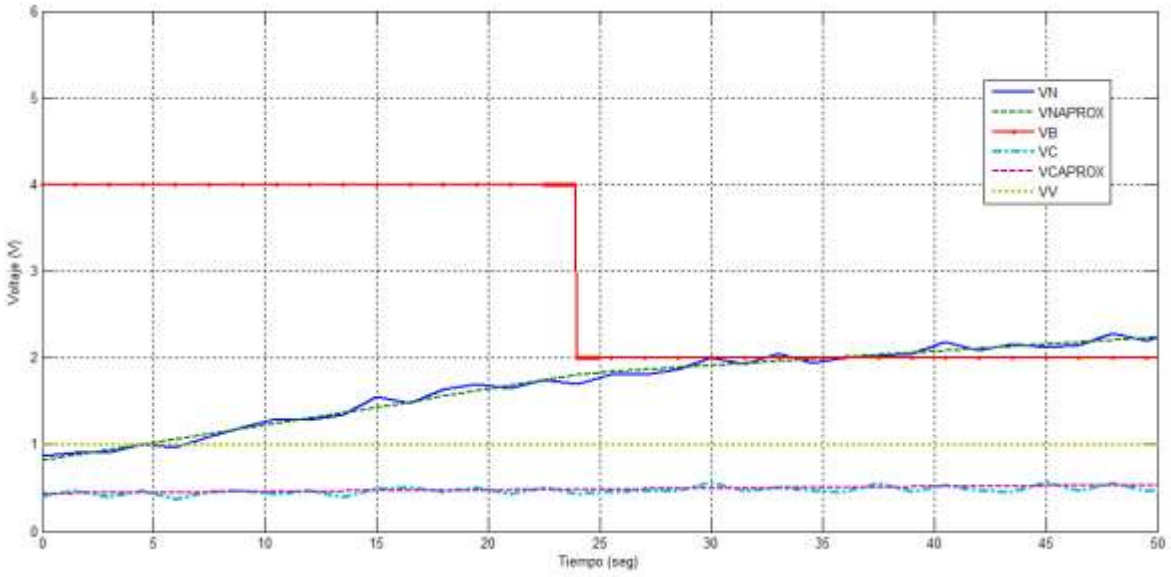
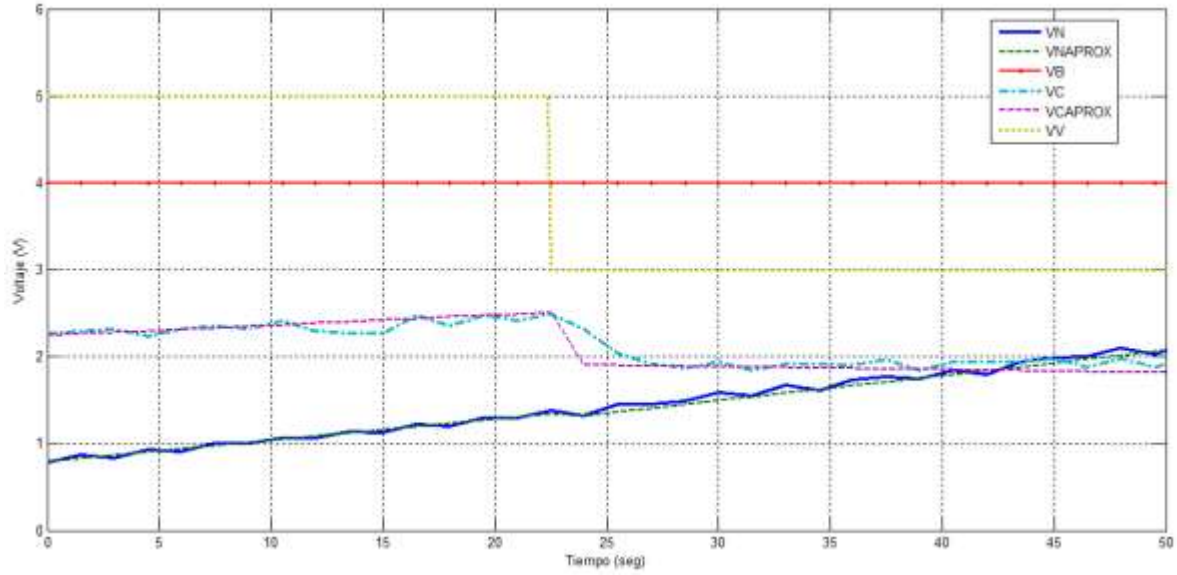



**Figura 24. Respuesta del proceso ante un cambio en el escalón en VV**




VNAPROX y VCAPROX corresponden a la regresión lineal de VN y VC respectivamente.

La *Figura 23* muestra que ante el cambio en el escalón de VB se experimenta una disminución en la pendiente de VN, pero no se evidencia ninguna variación en VC, en la *Figura 24* donde ante el cambio en el escalón de VV se observa una variación en VC y una variación despreciable en la pendiente de VN

Obtenida esta información, se calculan las ganancias en lazo abierto y las ganancias relativas para cada variable, de acuerdo a las *Ecuaciones 6* y 9. Los resultados se registran en la *Cuadro 3.*

$$K_{12} = \frac{\Delta c1}{\Delta m2}\Big|_{m1} = \frac{0.0288 - 0.0248}{3 - 5} = -0.002 \approx 0 \qquad K_{11} = \frac{\Delta c1}{\Delta m1}\Big|_{m2} = \frac{0.016 - 0.409}{2 - 4} = 0.1965$$

$$K_{22} = \frac{\Delta c2}{\Delta m2}\Big|_{m1} = \frac{1.9 - 2.3}{3 - 5} = 0.2 \qquad K_{21} = \frac{\Delta c2}{\Delta m1}\Big|_{m2} = \frac{0}{2 - 4} = 0$$

*Ecuaciones 14*.

$$\mu_{11} = \frac{K_{11}K_{22}}{K_{11}K_{22} - K_{12}K_{21}} = \frac{(0.1965)(0.2)}{(0.0393) - (0)} = 1$$

$$\mu_{12} = \frac{-K_{12}K_{21}}{K_{11}K_{22} - K_{12}K_{21}} = \frac{-(0)(0)}{(0.0393) - (0)} = 0$$

$$\mu_{21} = \frac{-K_{12}K_{21}}{K_{11}K_{22} - K_{12}K_{21}} = \frac{(0)(0)}{(0.0393) - (0)} = 0$$

$$\mu_{22} = \frac{K_{11}K_{22}}{K_{11}K_{22} - K_{12}K_{21}} = \frac{(0.1965)(0.2)}{(0.0393) - (0)} = 1$$

*Ecuaciones 15.*

**Cuadro 3. Ganancias en lazo abierto y ganancias relativas**

| Ganancias de Lazo Abierto | | Ganancias Relativas | |
|---|---|---|---|
| $K_{11}$ | 0.1965 | $\mu_{11}$ | 1 |
| $K_{12}$ | 0 | $\mu_{12}$ | 0 |
| $K_{21}$ | 0 | $\mu_{21}$ | 0 |
| $K_{22}$ | 0.2 | $\mu_{22}$ | 1 |

De lo anterior se obtiene la matriz de ganancias relativas, *Cuadro 4* donde se evidencia que el sistema puede ser tratado como dos sistemas SISO desacoplados y se establece que el Nivel debe ser controlado por VB y el caudal por VV dado al alto grado de interacción que se presenta entre las variables de entrada y salida mencionadas.

**Cuadro 4. Matriz de ganancias relativas**

| | VB | VV |
|---|---|---|
| Nivel | 1 | 0 |
| Caudal | 0 | 1 |

# 5. DISEÑO DE LOS CONTROLADORES NEURONALES

Para realizar el diseño de los controladores neuronales de cada uno de los dos subsistemas del proceso, es necesario seguir los siguientes pasos:

- Adquisición de datos.
- Identificación de los modelos matemáticos del sistema.
- Creación de los controladores por modelo inverso de cada subsistema.
- Simulación de los controladores neuronales.
- Implementación de los controladores neuronales en tiempo real.

## 5.1 ADQUISICIÓN DE DATOS

En esta primera etapa es necesario recolectar los datos que describen correctamente el comportamiento real del sistema, pues de esto depende el desarrollo satisfactorio de las etapas siguientes y por consiguiente un control óptimo para el sistema.

Para poder hacer la adquisición del nivel (cm) y caudal (L/min) se deben realizar las siguientes conversiones:

**Nivel**: El sensor de nivel entrega valores en corriente entre 4mA y 20mA y está calibrado para que mida alturas entre 0cm y 50.8cm respectivamente. El conversor de corriente a voltaje entrega valores entre 0.98V para 0cm y 4.93V para 50.8cm, para facilitar los cálculos se restan los 0.98V, de modo que ahora el rango esta entre 0V y 3.95V. Realizando una regla de tres se obtiene la ecuación que relaciona el voltaje entregado al Arduino y el nivel:

$$nivel = 12.86 * (V - 0.98)$$ *Ecuación 16.*

**Caudal:** La salida del sensor de caudal esta descrito por[25]:

$$F_{sensor} = 4.1 * Q$$ *Ecuación 17.*

[25] Water flow sensor HZ21WA. Disponible en web: <http://www.microelectronicos.com/datasheets/UCTS0058.pdf>

Ahora la frecuencia máxima del conversor de frecuencia a voltaje es de 56Hz y el voltaje es de 4.9V, realizando una regla de tres se tiene:

$$F_{sensor} = \frac{56Hz}{4.9V} * V$$ *Ecuación 18.*

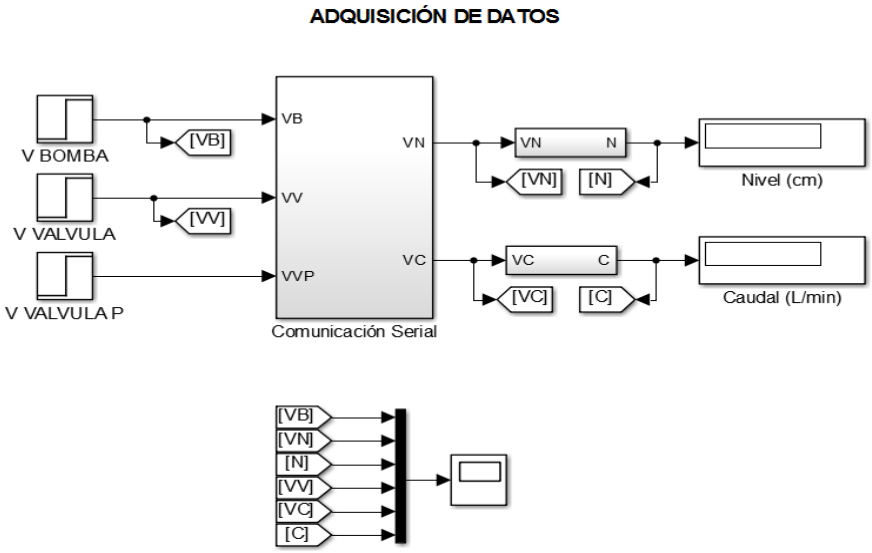
Finalmente se despeja Q de la Ecuación 17. y se reemplaza $F_{sensor}$ en esta misma.

$$Q = 2.414 * (V - 1.09)$$ *Ecuación 19.*

Para hacer la adquisición de datos se ha creado un archivo en Simulink *Figura 25,* en donde incorpora los bloques *serial sent* y *serial receive* los cuales realizan la comunicación entre el arduino y el Simulink. Debido a que la transmisión en serie requiere un formato *uint8* y que las operaciones realizadas en Simulink requieren formato *double* se hizo necesario usar los correspondientes conversores con el bloque *Data Type Conversion*.

También fue necesario incorporar *subsystems* que realizan las conversiones pertinentes, elementos de visualización y recolección de datos como *displays* y *scopes* y los bloques generadores de señales como *steps* y *uniform random numbers*.

**Figura 25. Adquisición de datos**



El programa almacena los siguientes datos:

VB: Voltaje aplicado a la bomba, primera variable manipulada.

VV: Voltaje aplicado a la válvula, segunda variable manipulada.

VN: Voltaje entregado por el sensor de nivel, primera variable controlada.

VC: Voltaje entregado por el sensor de caudal, segunda variable controlada.

N: Nivel entregado por el sensor de nivel (cm)

C: Caudal entregado por el sensor de caudal (L/min)

Para lograr la comunicación entre la tarjeta Arduino y los bloques *Serial Send* y *Serial Receive* de Simulink, se programó un algoritmo de lectura y escritura de datos en el Arduino, Ver *ANEXO B,* en el cual se leen los voltajes que entrega el sensor de nivel y el sensor de caudal por las entradas analógicas A0 y A1 respectivamente, para transmitirlos a la interfaz de Simulink a través del bloque *Serial Receive* y se escribe la señal de control para los actuadores (bomba, válvula de control y válvula de perturbación) por las salidas PWM 9, 10 y 11 respectivamente, transmitida desde el Simulink a través del bloque *Serial Send.*

Se decide usar la modulación por ancho de pulso (PWM) para proporcionar los voltajes de control a los actuadores, ya que mejora el rendimiento del sistema al controlar la cantidad de potencia entregada a la carga sin que haya pérdidas considerables de energía ni calentamiento de los artefactos, además permite una máxima variación del ciclo útil del 0% al 100%.

## 5.2 IDENTIFICACIÓN DE LA PLANTA

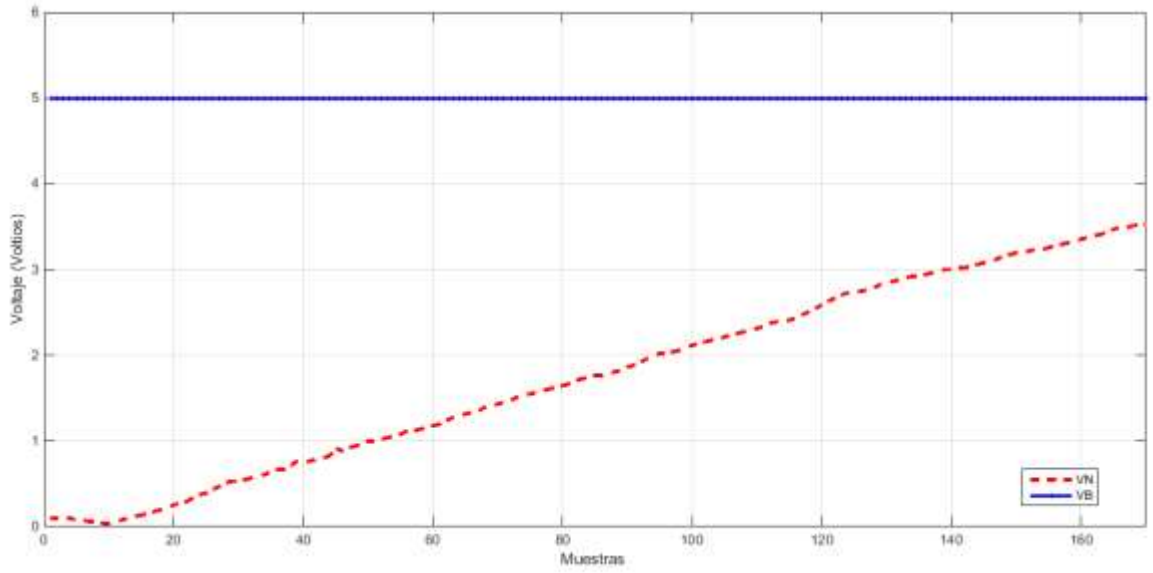
El modelamiento de los dos subsistemas de la planta se basa en pruebas experimentales del proceso.

**Determinación de Las Funciones De Transferencia.** Para encontrar las funciones de transferencia del sistema se hizo uso del *System Identification Toolbox*, puesto que es la herramienta especializada con la que cuenta Matlab

para esta tarea, con esta es posible obtener un modelo con solo los datos de entrada y salida del proceso como se muestra en [26].

**Función de transferencia para el nivel.** Se tomaron los datos en tiempo real del sistema en lazo abierto, donde se considera como entrada el voltaje de la bomba y como salida el voltaje de nivel, tal como se muestra en la *Figura 26* dejando el voltaje de la válvula en cero y por consiguiente el caudal en cero.

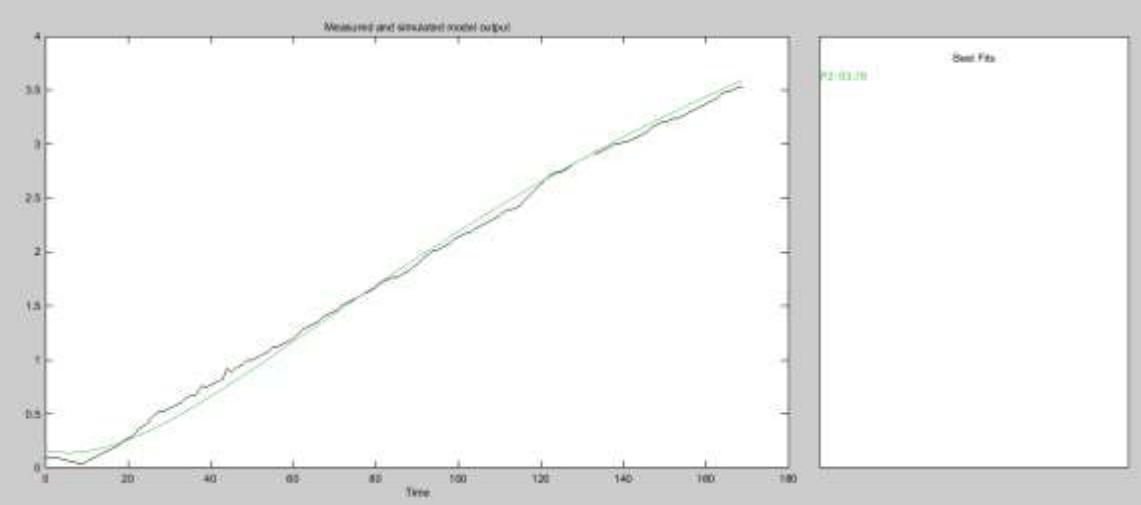
**Figura 26. Voltaje de la bomba - voltaje de nivel**




Con la herramienta *IDENT* se obtiene el modelo del sistema con una aproximación de 93.78%. Ver *Figura 27*.

[26] Osorio, Carlos. Extracción de Modelos Dinámicos Directamente de Datos Experimentales usando Identificación de Sistemas. Marzo, 2015. Mathworks. Disponible en web < http://www.mathworks.com/videos/extracting-dynamic-models-from-experimental-data-using-system-identification-spanish-100499.html >

**Figura 27. Identificación para VB-VN**

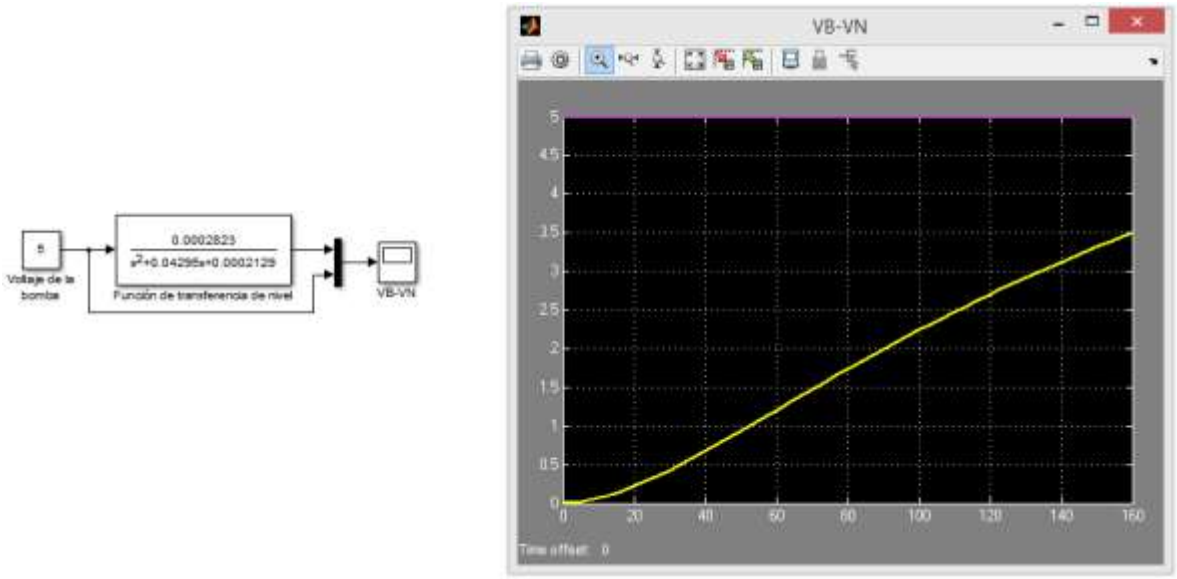



El cual arroja la siguiente función de transferencia, donde se observa que tiene dos polos reales, no tiene zeros ni retardo:

$$G_{11}(s) = \frac{0.0002823}{s^2 + 0.04295s + 0.0002129}$$ *Ecuación 20.*

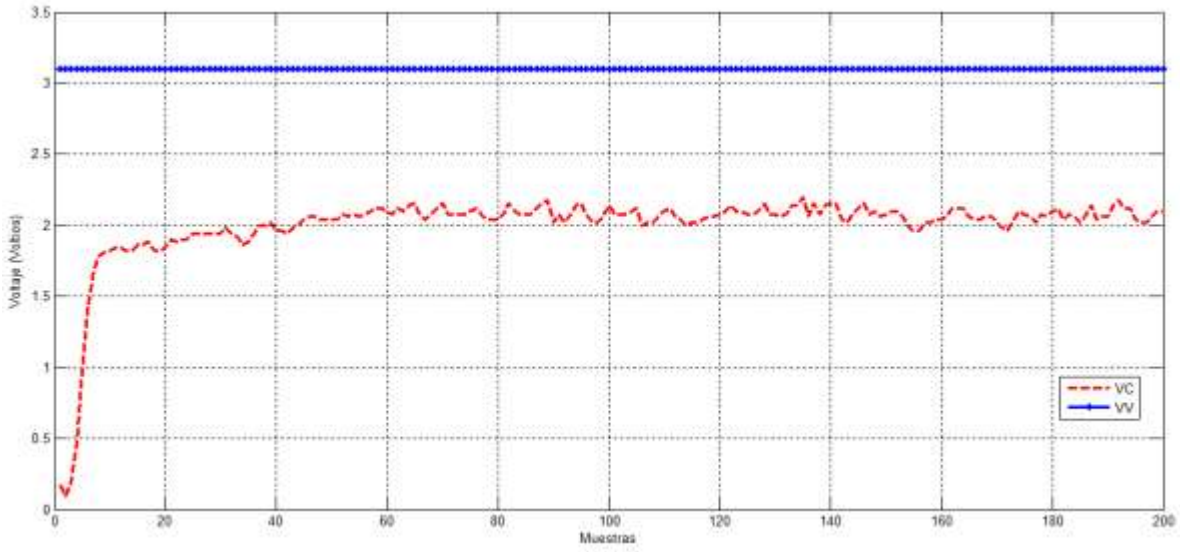
La *Figura 28* muestra la validación realizada en Simulink. Se puede ver que la respuesta del modelo, *Figura 28* es muy similar a la respuesta real, *Figura 26* teniendo en cuenta que a las dos se le aplica la misma señal de entrada de 5V, el valor de nivel máximo alcanzado en la *Figura 26* es de 3.7 y el de la *Figura 26* es de 3.5.

**Figura 28. Validación en Simulink del voltaje de la bomba con respecto al voltaje de nivel**



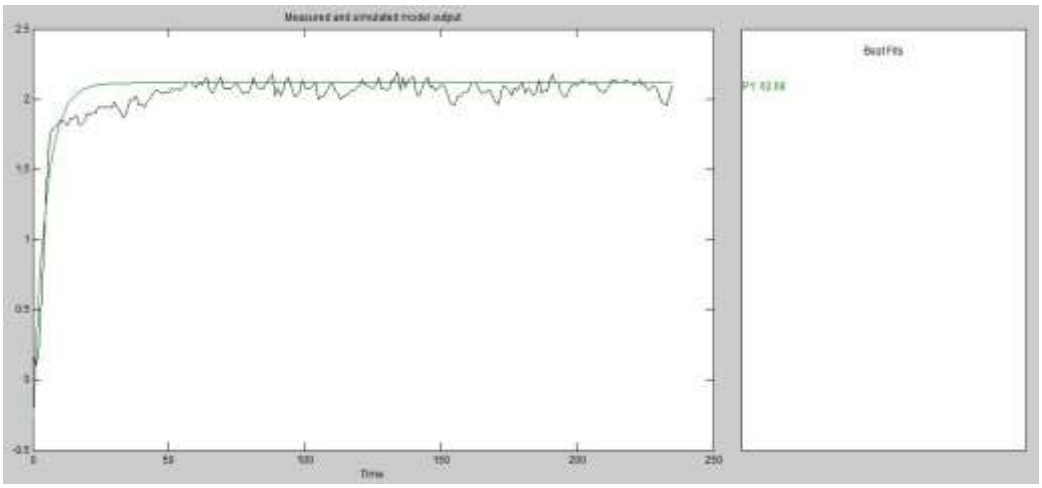

**Función De Transferencia Para El Caudal.** Se tomaron los datos en tiempo real del sistema en lazo abierto, donde se considera como entrada el voltaje de la válvula y como salida el voltaje de caudal, tal como se muestra en la *Figura 29* Dejando el voltaje de la bomba en un valor estable para que se mantenga un nivel en el tanque.

**Figura 29. Voltaje de la válvula - voltaje de caudal**



Con la Herramienta IDENT se obtiene el modelo del sistema con una aproximación de 62.68%. Ver *Figura 30*.

**Figura 30. Identificación para VV-VC**

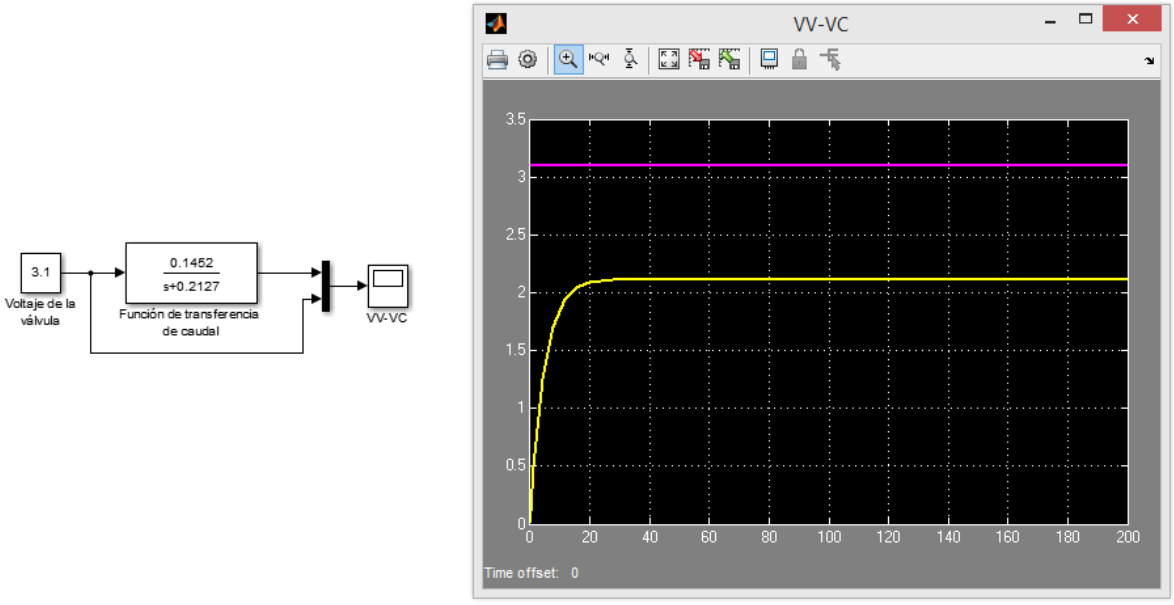


El cual arroja la siguiente función de transferencia, esta posee un solo polo real y no tiene zeros ni retardo.

$$G_{22}(s) = \frac{0.1452}{s + 0.2127}$$ *Ecuación 21.*

La *Figura 31* muestra la validación realizada en Simulink. Se puede ver que la respuesta del modelo, *Figura 31* es muy similar a la respuesta real, *Figura 29* teniendo en cuenta que a las dos se le aplica la misma señal de entrada de 3.1V, el valor de caudal máximo alcanzado en la *Figura 29* es de aproximadamente 2.1 al igual que en la *Figura 31*.

**Figura 31. Validación en Simulink del voltaje de la válvula con respecto al voltaje de caudal**



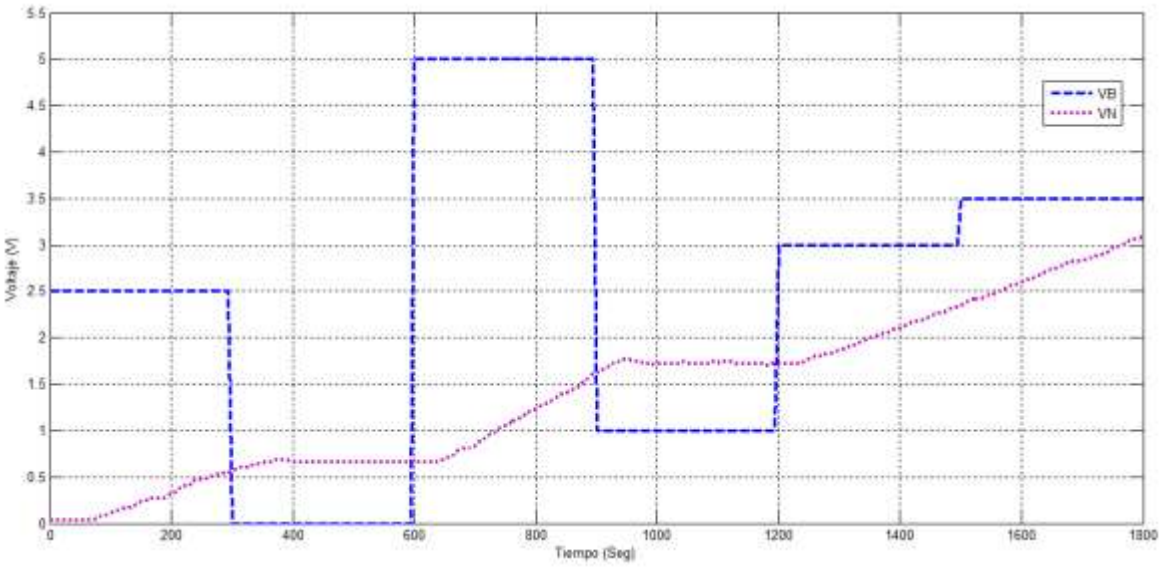

## 5.3 CONTROLADORES NEURONALES POR MODELO INVERSO

En esta etapa se diseñan dos controladores por modelo inverso, uno para cada variable a controlar.

**5.3.1 Controlador para la variable nivel.** El sistema de control para la variable nivel, toma como variable manipulada el voltaje aplicado a la bomba VB y como variable controlada el voltaje de nivel VN. Para el nivel se desarrolla un control por modelo interno, ya que las entradas de la red son la salida real de la planta y el error entre la salida del modelo directo y la salida real de la planta. Con la ayuda del Neural Network toolbox de Matlab se desarrolla el algoritmo, ver *ANEXO D*, que procede de la siguiente manera:

a. Carga los valores del VB y VN.

**Figura 32. VB-VN para el entrenamiento**

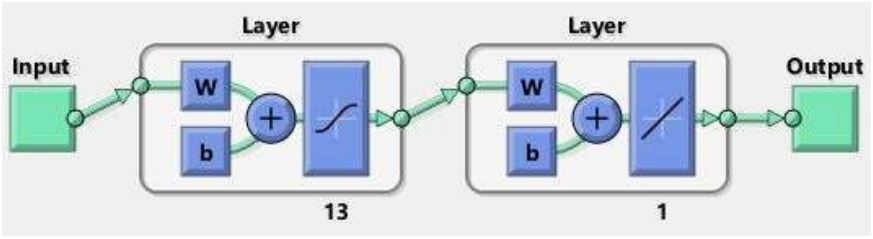



b. Genera los patrones de entrenamiento usando el comando *convec*, con este comando se crea una matriz con todas las posibles combinaciones de los valores que toma el sistema.

c. Calcula el error de nivel por medio de un ciclo *for*, entre la salida simulada de la planta, la cual se obtiene aplicando el comando *sim* al modelo del nivel, y los datos de salida real del proceso cargados al inicio (VN).

d. Crea la red neuronal con los datos del error de voltaje nivel y el voltaje de nivel (VN) usando el comando *newff*, esta posee dos capas, una capa oculta con 13 neuronas y la capa de salida con 1 neurona, para la elección del número de neuronas en las capas ocultas no existe una regla, se deben hacer pruebas y elegir el número con el que tenga mejor comportamiento. La función de activación usada para la capa oculta fue sigmoidal tangencial hiperbólica (*tansig*), al ser una función no lineal permite a la red aprender relaciones lineales y no lineales entre la entrada y la salida, el rango de operación de esta función está entre -1 y 1. La función de activación utilizada para la capa de salida es *pureline*, ya que al ser lineal no modifica la salida calculada por la capa oculta y además puede tomar cualquier valor.

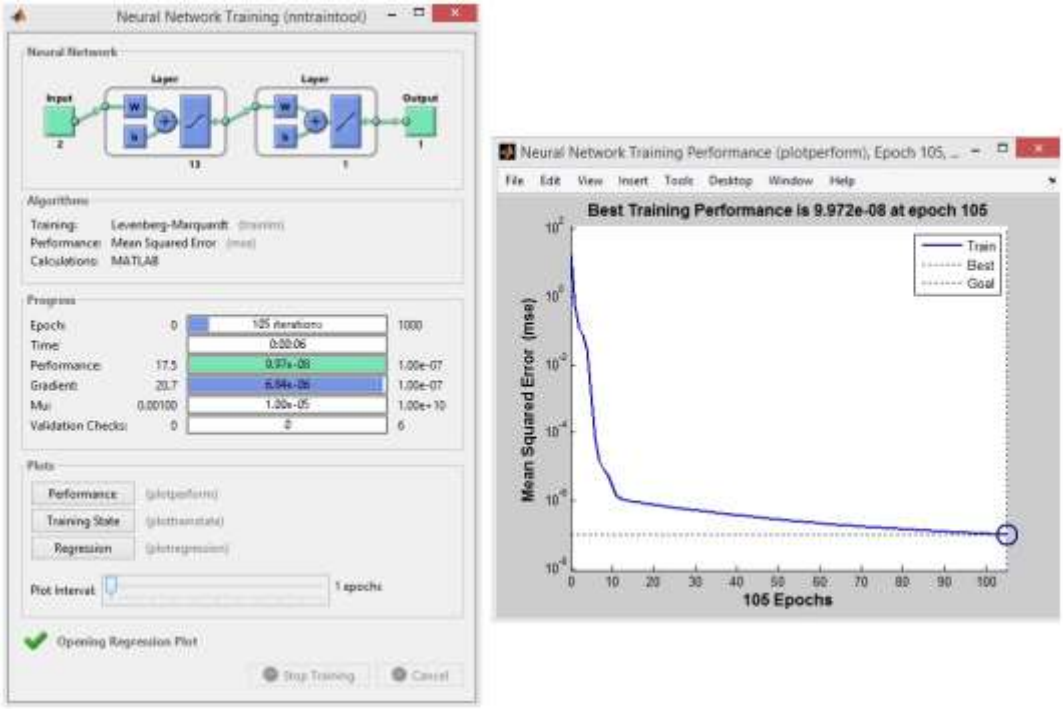
**Figura 33. Red neuronal para el nivel**




e. Realiza el entrenamiento de la red neuronal del modelo inverso del sistema usando el comando *train*. El método de entrenamiento seleccionado fue el de Levenberg-Marquardt (*trainlm*), ya que está diseñado especialmente para disminuir el error cuadrático medio, un parámetro fundamental en el diseño de controladores neuronales.

   En la *Figura 34* se puede ver que el entrenamiento es exitoso. El error cuadrático medio (MSE) indica la distancia entre la estimación de los valores de prueba del modelo y los valores de la prueba real, por lo que entre más cercano a cero sea este valor, será mucho mejor; para este caso el MSE fue de 9.972e-08 que fue logrado en 105 épocas.

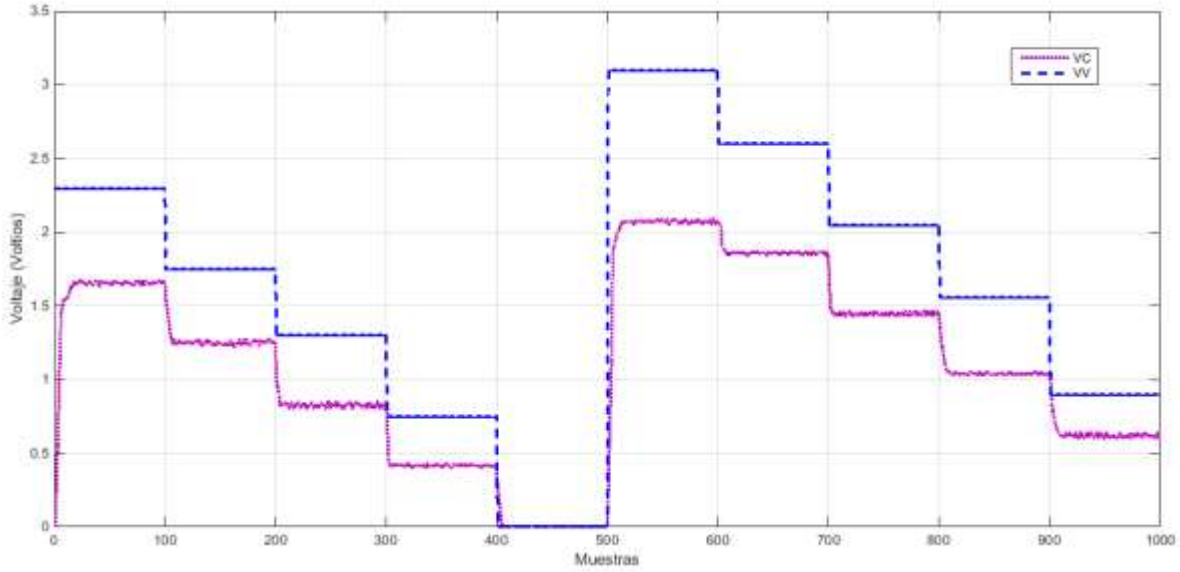
**Figura 34. Entrenamiento para la red neuronal del nivel**



**5.3.2 Controlador para la variable caudal.** El sistema de control para la variable caudal, toma como variable manipulada el voltaje aplicado a la válvula de control VV y como variable controlada el voltaje de caudal VC. Con la ayuda del Neural Network toolbox de Matlab se desarrolla el algoritmo, ver *ANEXO D*, que procede de la siguiente manera:

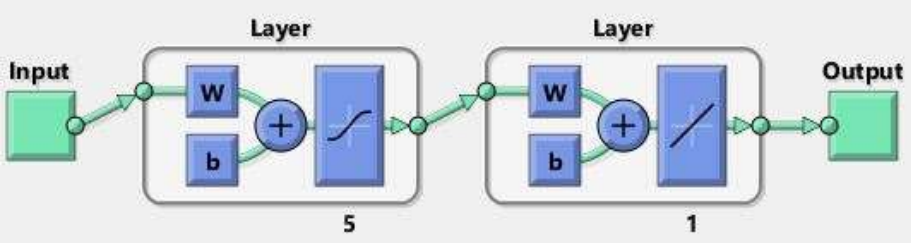
a. Carga los valores de VV y VC.

**Figura 35. VV-VC para el entrenamiento**



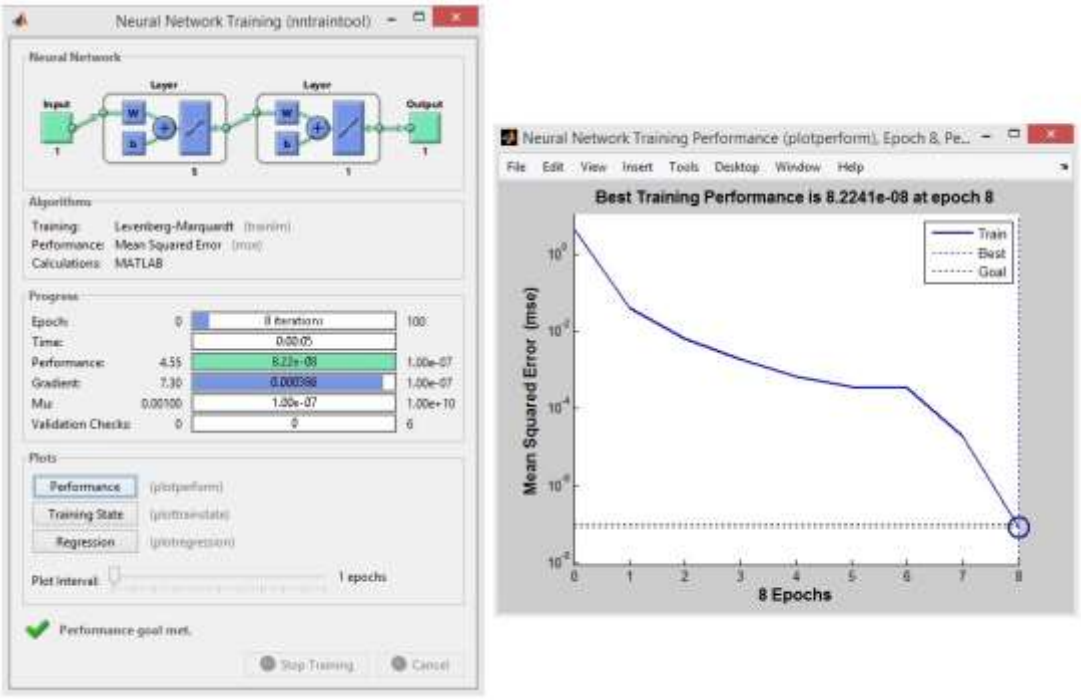

b. Crea la red neuronal del modelo inverso del sistema (Controlador). La red está formada por dos capas, una capa oculta con 5 neuronas y la capa de salida con 1 neurona. Las funciones de activación usadas son las mismas de la red neuronal para el nivel, en la capa oculta la función *tansig* y en la capa de salida *purelin*.

**Figura 36. Red Neuronal para el caudal**




c. Realiza el entrenamiento de la red neuronal del modelo inverso del sistema, el método usado al igual que en la variable nivel es el Levenberg-Marquardt (*trainlm*). En la *Figura 37* se observa que el MSE es de 8.2241e-08 por lo que se puede afirmar que el entrenamiento fue exitoso.

**Figura 37. Entrenamiento para la red neuronal del caudal**




## 5.4 SIMULACIÓN DE LOS CONTROLADORES NEURONALES

El controlador es puesto a prueba aplicándolo al modelo de la planta, tal como se muestra en la *Figura 38* Donde se generan diferentes cambios en el set-point de nivel y de caudal que abarcan el rango de operación, obteniendo la respuesta graficada en la *Figura 39* En la cual se aprecia un sobreimpulso en el nivel mucho

mayor a los presentados en el caudal, un error en estado estable y un tiempo de establecimiento, pero en general, realiza una buena labor de control para las dos variables. El análisis de este comportamiento se hace en la sección 5.6.

**Figura 38. Sistema de control basado en redes neuronales aplicado al modelo de la planta**

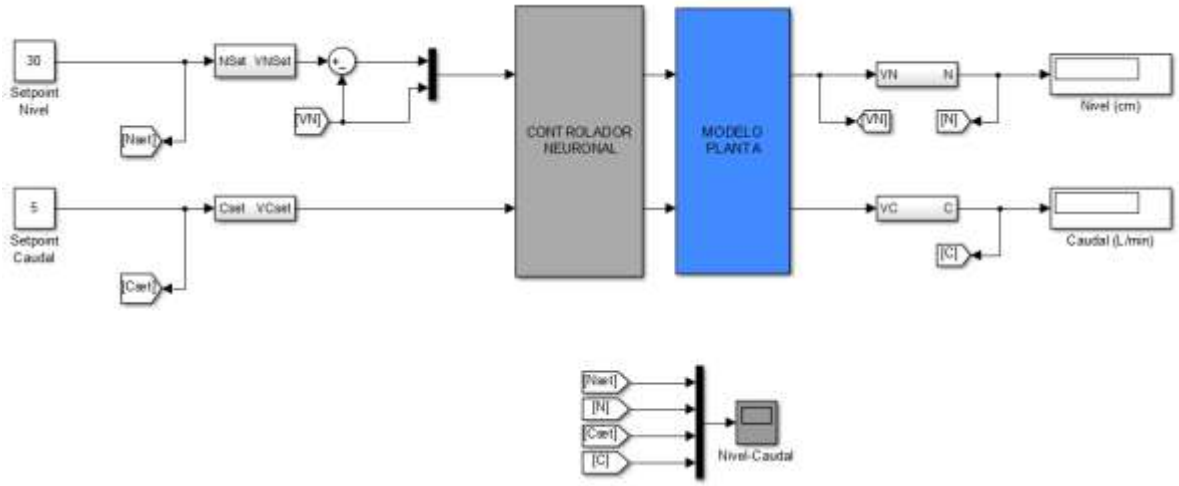
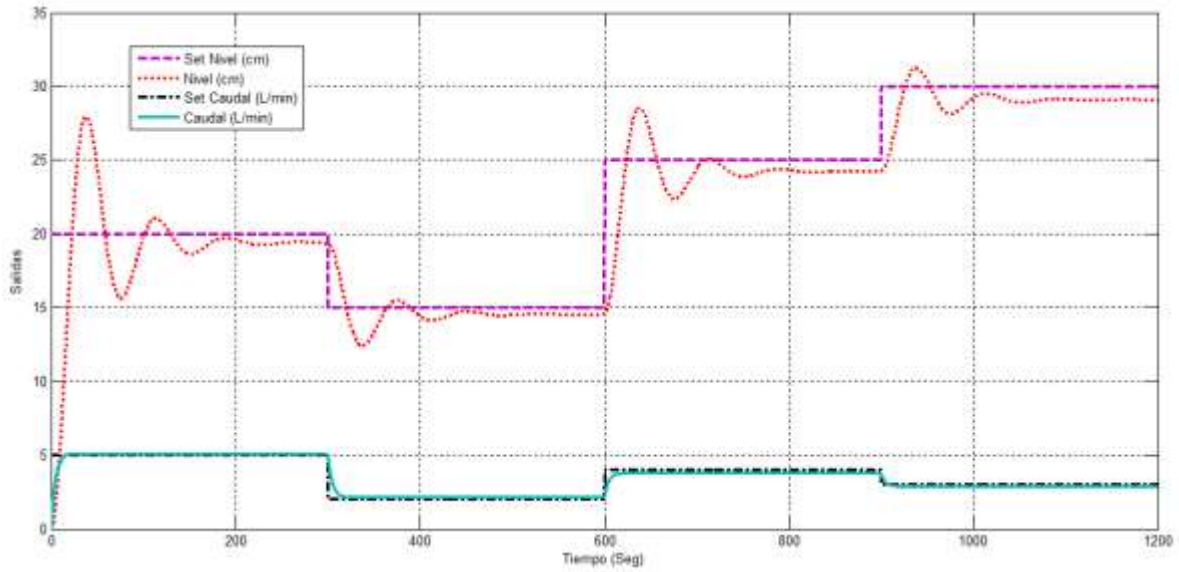


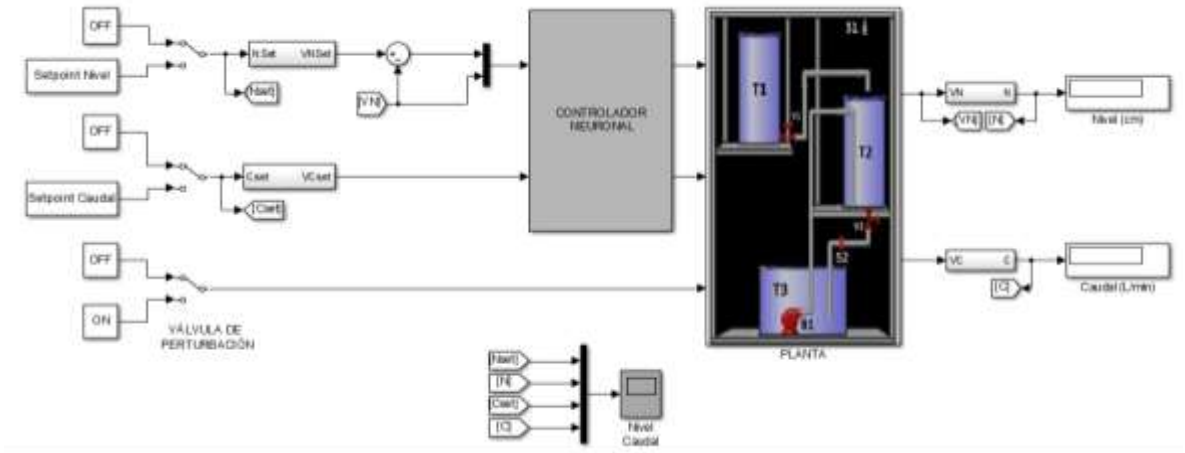

**Figura 39. Respuesta del controlador simulada para diferentes cambios en el set-point de nivel y de caudal**



## 5.5 IMPLEMENTACIÓN DE LOS CONTROLADORES NEURONALES EN TIEMPO REAL

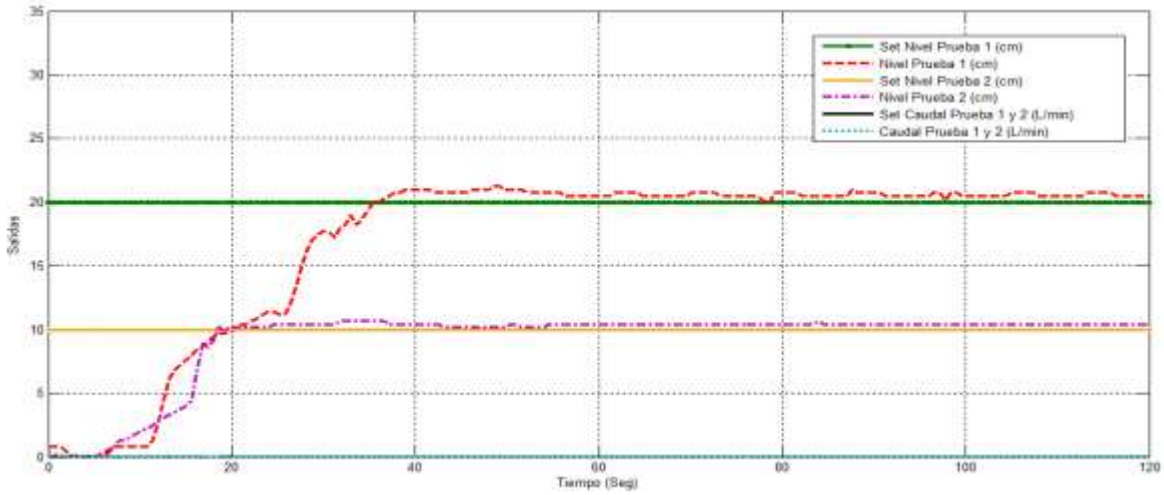
Luego de realizar las pruebas del control en simulación y observar que se obtiene una respuesta satisfactoria, se procede a su aplicación en la planta. La *Figura 40* muestra la interfaz de control en tiempo real implementada en Simulink a través de la cual se realizan las pruebas de los controladores.

**Figura 40. Sistema de control basado en redes neuronales en tiempo real**



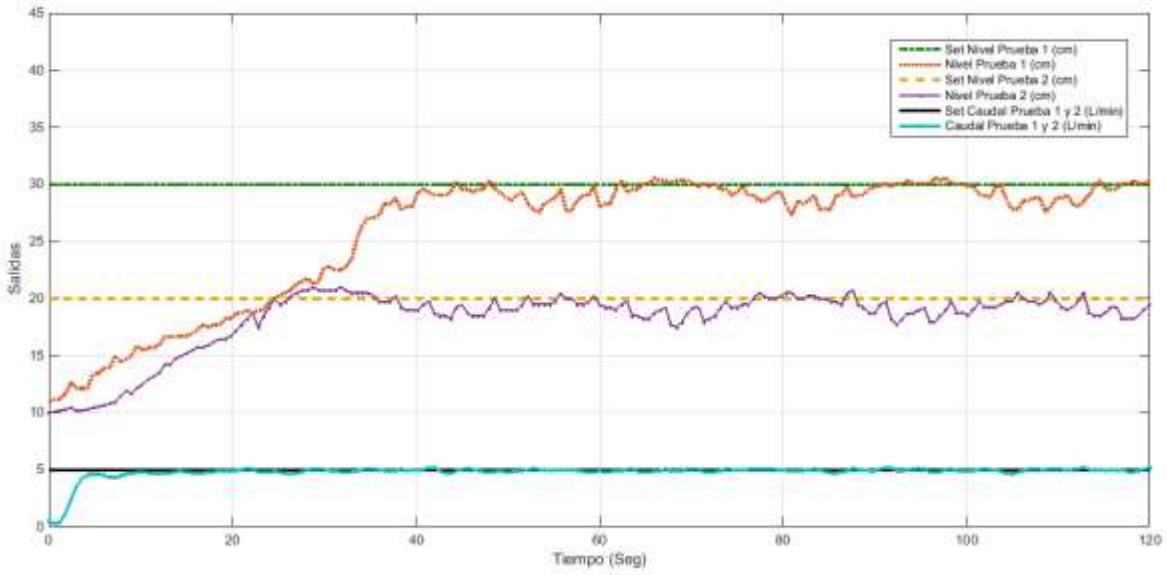

En la *Figura 41* se observan dos pruebas de la respuesta del control para diferentes valores de set-point de nivel, sin flujo de salida de líquido y con el tanque T2 completamente vacío, haciéndose evidente que cuanto mayor sea el nivel de líquido que se desea alcanzar, el tiempo de establecimiento aumenta. En la prueba 1, para un nivel de 20 cm se establece aproximadamente a los 40 segundos y en la prueba 2, para un nivel de 10 cm se establece aproximadamente en 20 segundos, la mitad del tiempo que le llevó a la prueba 1.

**Figura 41. Respuesta del controlador para diferentes set-point de nivel con Caudal de salida en cero**



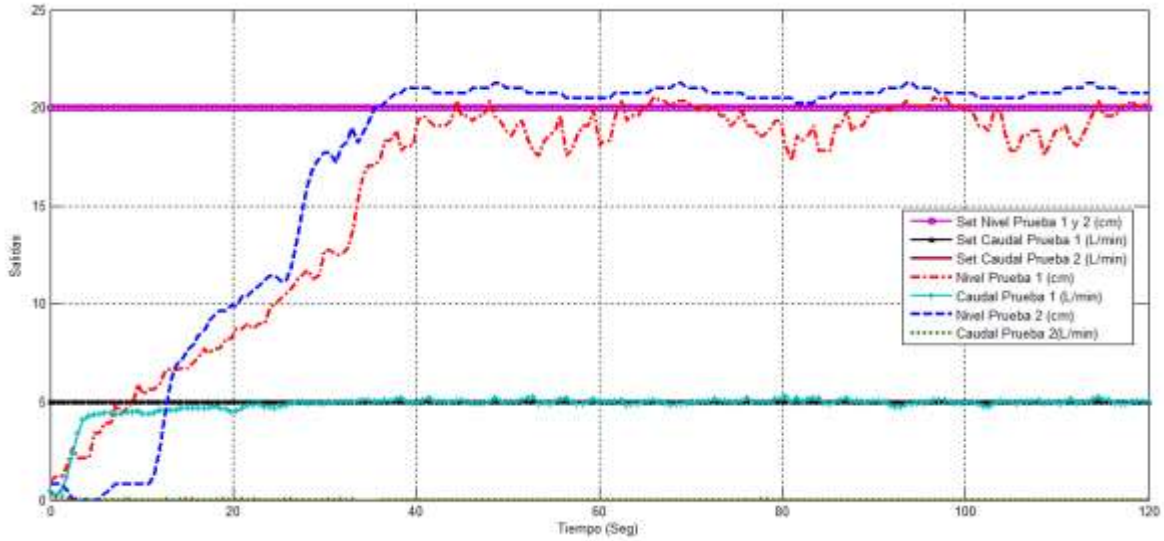

Se muestran en la *Figura 42* otras dos pruebas de la respuesta del control para diferentes valores de set-point de nivel, esta vez con un flujo de salida de líquido de 5 L/min y con un nivel inicial de líquido en T2 de 10 cm, donde se evidencian fluctuaciones en el estado estable de dicha variable, debido a que hay un flujo de líquido de salida que perturba el control. La prueba 1 muestra un aumento en el tiempo de establecimiento a 40 segundos para el valor de set-point de 30 cm y la prueba 2 tiene un tiempo de establecimiento de 25 segundos para un valor de setpoint de 20 cm y el caudal se establece en 5 segundos.

**Figura 42. Respuesta del controlador para diferentes set-point de nivel con caudal de salida en 5 L/min**



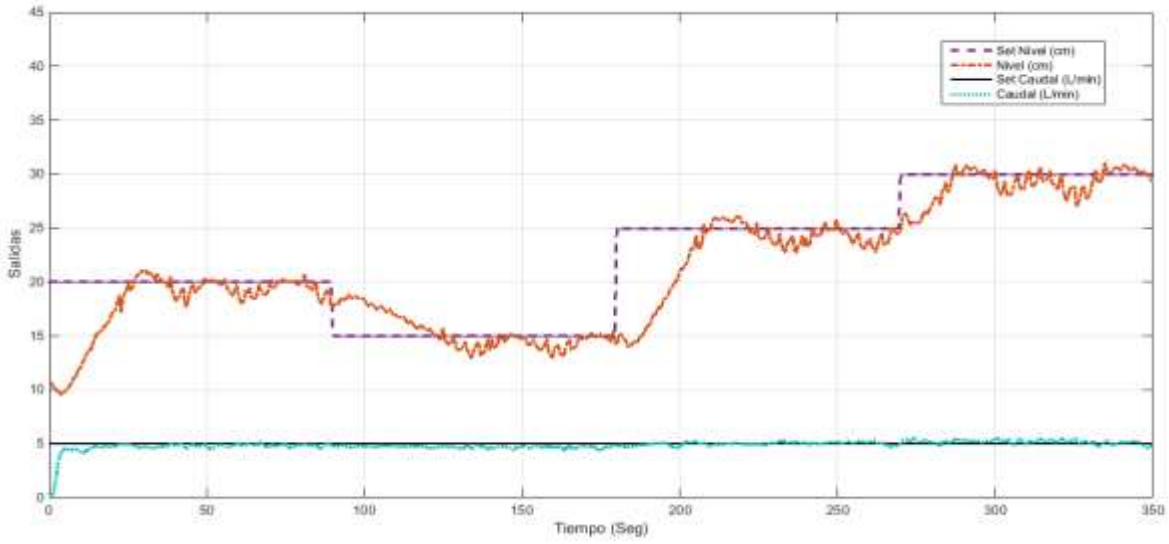

Se realizan dos pruebas donde se pretende mostrar que la afectación ocasionada por el caudal de salida de líquido en el tiempo de establecimiento del nivel no es tan considerable, como se muestra en la *Figura 43,* en la primera prueba se aplica el caudal de salida máximo que es de 5 L/min y se establece un set-point de nivel de 20 cm el cual es alcanzado a los 45 segundos, y en la segunda prueba se cierra totalmente la válvula V2 impidiendo la salida de líquido del tanque T2 y se establece el mismo set-point de 20 cm, el cual es alcanzado a los 35 segundos.

**Figura 43. Respuesta del controlador para diferentes set-point de caudal y un nivel en 20cm**



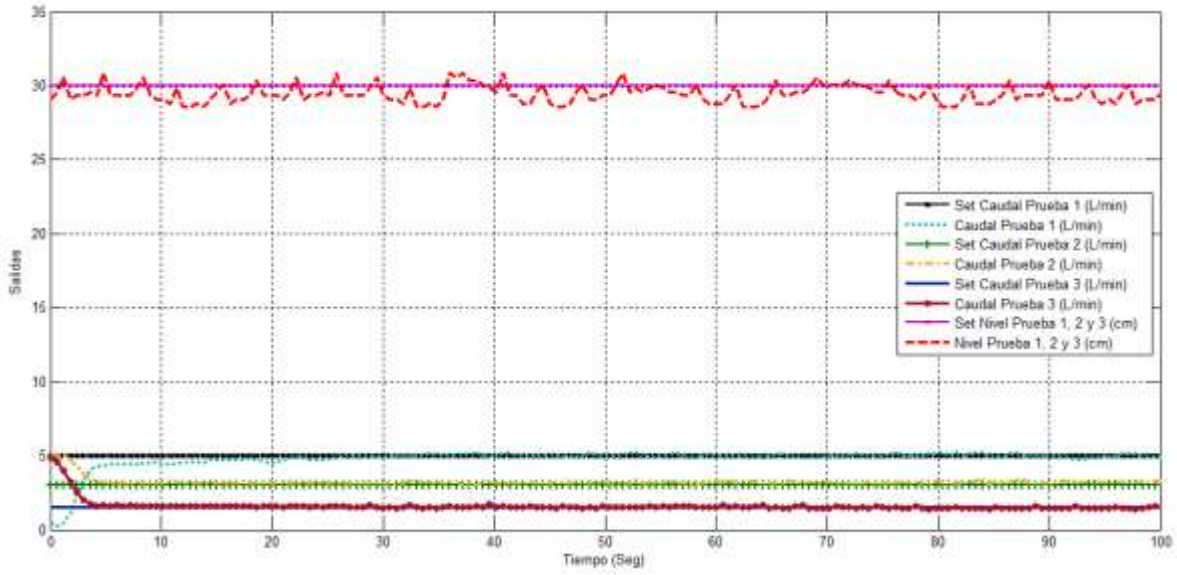

En la prueba de la *Figura 44* se muestra el comportamiento del nivel controlado para diferentes cambios en su set-point aplicando un flujo de salida constante de líquido de 5 L/min y un nivel inicial de líquido en T2 de 10 cm, observándose el mismo tiempo de establecimiento de 5 segundos para el caudal y las mismas fluctuaciones de las pruebas 1 y 2 de la *Figura 43* debido a la perturbación ocasionada por el caudal de salida.

**Figura 44. Respuesta del controlador para diferentes cambios en el set-point de nivel con caudal de salida en 5 L/min**



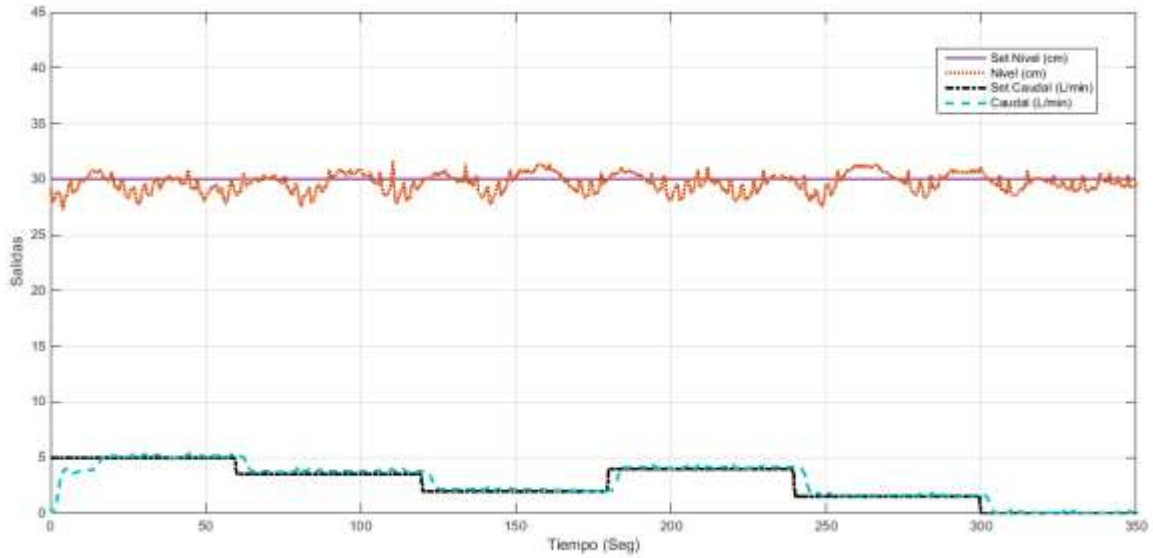
En la *Figura 45* se observan tres pruebas de la respuesta del control para diferentes valores de Set-point de caudal, con un nivel constante en el tanque T2 de 30 cm, evidenciándose que el valor del set-point de caudal que se desea alcanzar no afecta el tiempo de establecimiento ya que en todos los casos el caudal se establece aproximadamente a los 5 segundos. En la realización de las pruebas 2 y 3 se hace necesario iniciarlas en el caudal máximo que es 5 L/min, para lograr eliminar las burbujas de aire producidas por la tubería.

**Figura 45. Respuesta del controlador para diferentes set-point de caudal con un nivel constante en 30 cm**



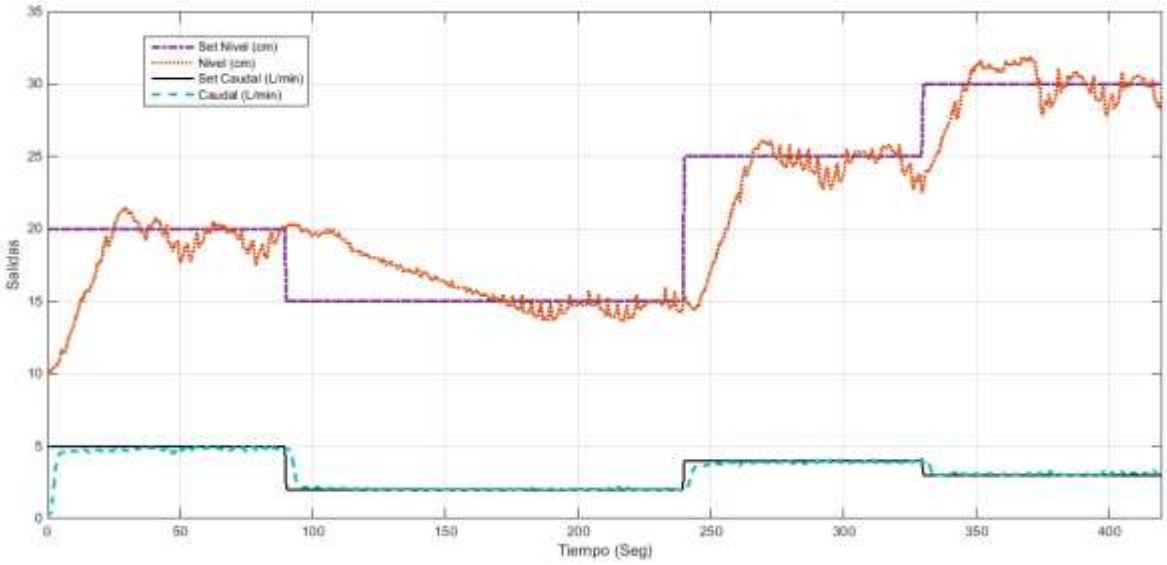

En la prueba de la *Figura 46* se muestra el comportamiento del caudal controlado para diferentes cambios en su set-point aplicando un nivel constante de líquido de 30 cm. En esta prueba se logra reafirmar lo observado en las pruebas 1, 2 y 3 de la *Figura 45* puesto que en cada cambio en el escalón del set-point, el caudal se establece aproximadamente a los 5 segundos de haber ocurrido dicho cambio; excepto en el primer intervalo de tiempo donde el caudal no logra alcanzar el set-point en los 5 segundos como se esperaría, debido a las burbujas que se presentaron en la tubería, lo cual torna al fluido turbulento y solo hasta los 17 segundos alcanza un flujo laminar.

**Figura 46. Respuesta del controlador para diferentes cambios en el set-point de caudal con un nivel constante en 30 cm**




Para la prueba de la *Figura 47* se aplican cambios tanto en el set-point de nivel como en el de caudal, una particularidad observada en esta prueba es que en el intervalo de tiempo de 90 a 240 segundos al nivel le toma un mayor tiempo en alcanzar su siguiente set point debido a que este cambia de un valor mayor a uno menor y que el flujo de salida de líquido no es el máximo.

**Figura 47. Respuesta del controlador para diferentes cambios en el set-point de nivel y de caudal**

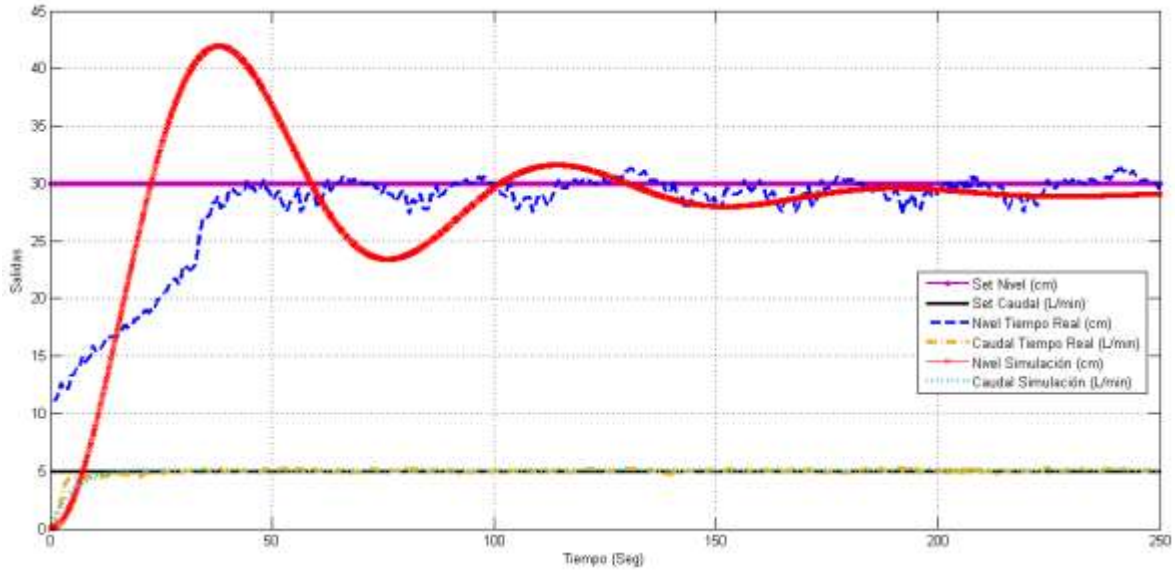


## 5.6 VALIDACIÓN DE LA RESPUESTA DEL CONTROL

Para realizar la validación de los controladores en simulación y en tiempo real se aplica un set-point de nivel de 30 cm y un set-point de caudal de 5 L/min como se muestra en la *Figura 48* donde se evidencia que la respuesta entre el caudal simulado y el caudal en tiempo real son muy aproximadas con una diferencia en el tiempo de establecimiento de 5 segundos puesto que, en tiempo real tarda 5 segundos en alcanzar su valor de estado estable y en simulación tarda 10 segundos en lograrlo.

Para la variable nivel, la respuesta en tiempo real y simulada difieren en su comportamiento en estado transitorio, dado que la primera no presenta sobreimpulso y tiene un tiempo de establecimiento de 45 segundos, a diferencia de la respuesta simulada, la cual presenta un sobreimpulso de 39.15% y un tiempo de establecimiento de 130 segundos; esto se debe a que el modelo matemático más aproximado que genera el IDENT es el de un sistema subamortiguado de segundo orden y como tal, su respuesta ante un escalón va a ser la propia de este tipo de sistemas, por lo cual no describe de manera precisa las características dinámicas de la bomba. Pero se observa que el comportamiento en estado estacionario es muy aproximado.

**Figura 48. Comparación del desempeño del control neuronal simulado y en tiempo real**



# 6. COMPARACIÓN DEL DESEMPEÑO DEL CONTROL NEURONAL Y EL CONTROL DIFUSO

Con base en las pruebas que se realizaron y que se muestran en el capítulo anterior se evalúa el desempeño del control teniendo en cuenta tres parámetros:

El sobreimpulso, es el valor pico máximo que se obtiene de la curva de respuesta, medido desde el valor deseado. En este control no existe sobreimpulso para ninguna de las dos variables controladas, en el nivel se presentan algunas oscilaciones que no son consideradas sobreimpulso ya que son causadas por la perturbación que se presenta cuando hay caudal de salida.

El error en estado estacionario, es la diferencia entre el valor deseado y el valor en estado estacionario de la curva de respuesta, este último se obtiene calculando la regresión lineal de la curva a partir del momento en el que se estabiliza. Para el control implementado se observa que el porcentaje de error es muy pequeño, siendo el valor máximo en las pruebas realizadas de 0.01%.

El tiempo de establecimiento se define como el tiempo mínimo en el que la curva de respuesta alcanza y se mantiene en un rango preestablecido. En este control, el tiempo de establecimiento para la variable nivel varía de acuerdo al valor inicial que tenga y al set-point que se desea alcanzar, por el contrario para la variable caudal el tiempo de establecimiento se mantiene casi igual sin importar las condiciones iniciales y su set-point.

Los resultados obtenidos se consignan en la *Cuadro 5.*

**Cuadro 5. Resultados del controlador neuronal**

| Figura | Variable | Valor inicial | Set-Point | Tiempo de Establecimiento (Seg.) | Error en Estado Estacionario (%) | Sobre Impulso (%) |
|---|---|---|---|---|---|---|
| ***Figura 42*** | **Nivel (cm)** | 10 | 30 | 40 | 0.01 | 0 |
| | **Caudal (L/min)** | 0 | 5 | 6 | 0.0002 | 0 |
| ***Figura 47*** | **Nivel (cm)** | 10 | 20 | 24 | 0.005 | 0 |
| | **Caudal (L/min)** | 0 | 5 | 6 | 0.002 | 0 |
| | **Nivel (cm)** | 20 | 15 | 30 | 0.005 | 0 |
| | **Caudal (L/min)** | 5 | 2 | 5 | 0.003 | 0 |
| | **Nivel (cm)** | 15 | 25 | 15 | 0.0025 | 0 |
| | **Caudal (L/min)** | 2 | 4 | 6 | 0.001 | 0 |
| | **Nivel (cm)** | 25 | 30 | 16 | 0.005 | 0 |
| | **Caudal (L/min)** | 4 | 3 | 6 | 0.001 | 0 |

Para realizar la comparación del rendimiento entre el control neuronal implementado y el control difuso desarrollado en el proyecto "Diseño e Implementación de un Controlador para un Sistema Multivariable de Nivel y Flujo utilizando Microcontrolador Especial para Lógica Difusa" [27] es necesario remitirse a la Tabla 5 de dicho proyecto. De la cual se obtiene que el error en estado estacionario disminuye notablemente, dado que el porcentaje de error máximo es de 0.01% y 0.003% por redes neuronales y de 3% y 1.75% por control difuso para nivel y caudal respectivamente. En cuanto al sobreimpulso, aunque en el control difuso es mínimo, el control por redes neuronales lo elimina en las dos variables controladas.

Para analizar el tiempo de establecimiento se tienen en cuenta la Figura 54 y Figura 55 del proyecto de control difuso[27], de las cuales se extraen los datos consignados en la *Cuadro 6* donde se observa que el tiempo de establecimiento también mejora considerablemente para las dos variables.

**Cuadro 6. Comparación del tiempo de establecimiento**

| | Control | Variable | Valor Inicial | Set-Point | Tiempo de Establecimiento (Seg.) |
|---|---|---|---|---|---|
| **Fig.42** | **RNA** | Nivel (cm) | 10 | 30 | 40 |
| **Tabla5[27]** | **Fuzzy** | | 0 | 20 | 70 |
| **Fig.47** | **RNA** | Nivel (cm) | 20 | 15 | 30 |
| **Fig.54[27]** | **Fuzzy** | | 18 | 12 | 100 |
| **Fig.47** | **RNA** | Caudal (L/min) | 2 | 4 | 6 |
| **Fig.55[27]** | **Fuzzy** | | 2 | 4 | 55 |

[27] Ramirez G. Viviana, Calvache G. Oscar. "Diseño e Implementación de un Controlador Multivariable de Nivel y Flujo Utilizando Microcontroladores Especiales para Lógica Difusa". Neiva, Colombia. 2013.

# 7. CONCLUSIONES

De acuerdo a la matriz de ganancias obtenida al aplicar el método de Bristol, no existe interacción alguna entre los lazos VB-VN y VV-VC, por lo que el sistema hidráulico puede considerarse como dos sistemas SISO independientes. Por tanto, el uso de técnicas de desacople no son necesarias.

La técnica del modelo inverso permite desarrollar controladores neuronales con características particulares de acuerdo al tipo de aplicación requerida. Para el caso de la variable caudal, al presentar una dinámica de complejidad moderada, la técnica del modelo inverso simple es suficiente para establecer una relación directa entre la entrada y la salida, sin requerir una realimentación o un procesamiento del estado actual de la variable controlada. Cuando el comportamiento de la variable a controlar es más complejo y además está expuesto a perturbaciones, como en el caso del nivel, es preciso utilizar una variación del modelo inverso conocida como modelo interno, que permite examinar las condiciones actuales de la variable controlada, evaluar el error y tomar las acciones correctivas necesarias para alcanzar el punto de establecimiento deseado.

Las simulaciones del controlador son consistentes con el rendimiento que se presenta durante las pruebas en tiempo real. Si bien existen algunas diferencias en el régimen transitorio de la respuesta de la variable nivel, estas se atribuyen a que el modelo de segundo orden no representa en su totalidad las características dinámicas del actuador implicado, en este caso, la bomba. Esta aproximación produce un sobreimpulso característico de este tipo de sistemas, no obstante, el tiempo de establecimiento es muy similar en los dos casos.

El control basado en redes neuronales es superior al realizado mediante lógica difusa de acuerdo a los criterios evaluados, dado que se evidencia un mejoramiento del error en estado estacionario y del tiempo de establecimiento, a la vez que elimina por completo el sobreimpulso de las dos variables controladas del proceso.

Matlab es un software que provee todas las herramientas necesarias para desarrollar el modelamiento, el monitoreo, la visualización y el diseño del controlador basado en redes neuronales para el sistema hidráulico, a través de la interfaz de Simulink y los diferentes toolbox especializados para cada una de estas labores como el System Identification Toolbox y Neural Network toolbox.

# 8. RECOMENDACIONES

Realizar mantenimiento y limpieza periódicamente a los actuadores y a los tanques de almacenamiento, para que el sistema funcione en óptimas condiciones.

No obstaculizar el espacio entre el tanque de control y el sensor de Nivel ultrasónico con ningún objeto ajeno al sistema, dado que esto podría ocasionar lecturas erróneas del sensor y dañar algún componente de la placa de acondicionamiento de señales.

Al realizar las conexiones del sistema, es recomendable conectar primero la tarjeta de desarrollo Arduino al computador antes de energizar el circuito de acondicionamiento de señales, de igual manera, cuando se procese a apagar el sistema, se debe desenergizar el circuito antes de desconectar el Arduino del computador, esto con el fin de evitar posibles daños en la tarjeta.

En algunas ocasiones, el caudal de salida del tanque de control presenta flujos de líquido turbulento debido a las burbujas de aire en la tubería, lo que representa una limitación para el control del caudal ya que este se realiza para un flujo de salida laminar. Se recomienda, para lograr un caudal laminar, abrir la válvula V2 en su totalidad, es decir, aplicando un set-point del máximo caudal, en este caso 5 L/min y esperar a que las burbujas de aire desaparezcan y salga un caudal limpio, ya habiendo obtenido el tipo de caudal deseado se procede a realizar el control sin ningún inconveniente cambiando el set-point al valor deseado.

Como trabajos futuros se propone realizar la identificación del sistema por otros métodos y aplicar las técnicas de control clásico para realizar comparaciones del desempeño entre los métodos de control clásico y los métodos de control moderno.

## BIBLIOGRAFÍA


Ayuda de Matlab®. Disponible en web < http://www.mathworks.com/help/nnet/index.html>

Bermeo, Leonardo. Plataforma para la Implementación de Neuro-Controladores por Modelo Inverso de la Planta. 2004. Universidad del Valle. Cali, Colombia.

Bristol, E. H., On a New Measure of Interaction for Multivariable Process Control. IEEE Transaction on automatic control. January, 1996.

Dreyfus, Gérard. Neural Networks: Methodology and Applications. 2005. Springer (Ed).

Escobar, Luisa; Montoya, Oscar; Giraldo, Didier. Control Global del Péndulo Furuta Empleando Redes Neuronales Artificiales y Realimentacion de Variables de Estado. Junio, 2013. Universidad tecnológica de Pereira. Pereira, Colombia. Disponible en web < http://www.scielo.org.co/scielo.php?pid=S0123-77992013000100005&script=sci_arttext>.

Folleto Técnico Válvulas Solenoides Proporcionales Servoaccionadas De 2 Vías Tipo EV260B. Danfoss, 2014. Disponible en <http://www.ra.danfoss.com/TechnicalInfo/Literature/Manuals/04/IC.PD.200.O4.05.pdf>

Haykin, Simon. Neural Network: A Comprehensive Foundation. 1998. Person Prentice Hall (Ed).

LITE25 Level Indicating Transmitter Manual Series 3.5.1. Greyline Intrements Inc. Disponible en <http://www.greyline.com/pdf/LIT25%20Ultrasonic%20Manual%20Series%203.5.1.pdf>

Norgaard, M. (2000) Neural Networks for Modelling and Control of Dynamic Systems. Springer – Verlag, London.

Olivares, Victor; Urrea, Claudio; Cordoba, Félisa. Diseño e Implementación de un Simulador para el control de Posición de un Sistema a Grúa, Empleando Matlab-Simulink. 2014. Universidad de Santiago de Chile, Santiago de Chile. Disponible en web < http://www.researchgate.net/publication/270577126_Diseo_e_Implementacin_de_un_Simulador_para_el_Control_de_Posicin_de_un_Sistema_Gra_Empleando_MatLab-Simulink >.

Osorio, Carlos. Extracción de Modelos Dinámicos Directamente de Datos Experimentales usando Identificación de Sistemas. Marzo, 2015. Mathworks. Disponible en web < http://www.mathworks.com/videos/extracting-dynamic-models-from-experimental-data-using-system-identification-spanish-100499.html >

Parker. Optimal algorithms for adaptive networks: Second order backpropagation, second order direct propagation and second order Hebbian learning. 1995 En: D.B. Parker IEEE 1st Int. Conf. on Neural Networks, vol.2, pp.593 600, San Diego, CA.

Pino, Raúl; Gómez, Alberto; De Abajo, Nicolás. Introducción a la Inteligencia Artificial: Sistemas Experto, Redes Neuronales Artificiales y Computación Evolutiva. Oviedo, España. 2001. Servicios de Publicaciones Universidad de Oviedo (Ed).

Ponce, Cruz Pedro. Inteligencia Artificial con Aplicaciones a la Ingeniería. 2010. Alfaomega (Ed).

Precision Voltage to Current Converter-Transmitter XTR110KP. Texas Instrument, Texas, 2009. Disponible en web < http://www.ti.com/lit/ds/sbos141c/sbos141c.pdf>

Ramirez G. Viviana, Calvache G. Oscar. “Diseño e Implementación de un Controlador Multivariable de Nivel y Flujo Utilizando Microcontroladores Especiales para Lógica Difusa”. Neiva, Colombia. 2013.

Rodriguez, Victor. Garzón, Jaime. López, Jesús. Control Neuronal por Modelo Inverso de un Servosistema Usando Algoritmos de Aprendizaje Levenberg-Marquardt y Bayesiano. Disponible en web <http://arxiv.org/ftp/arxiv/papers/1111/1111.4267.pdf >

Rojas, Raul. Neural Network: A Systematic Introduction. 1996. Springer (Ed).

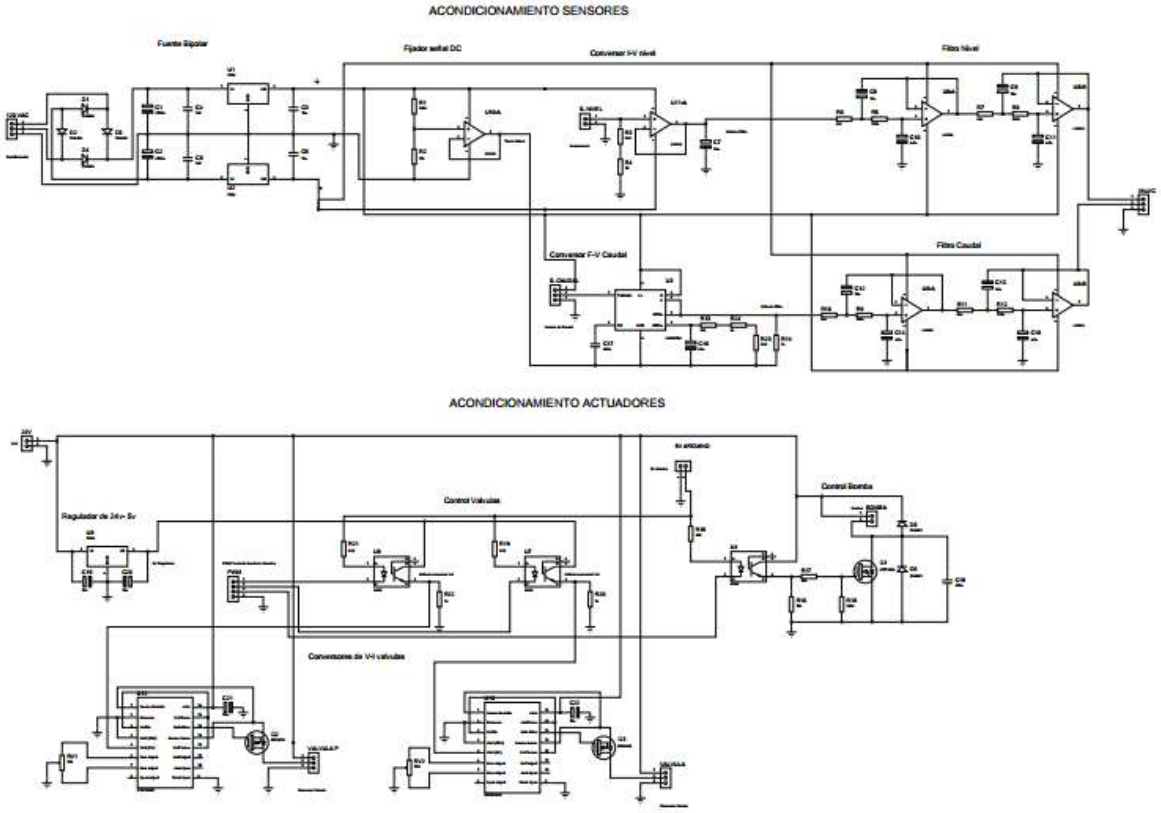
Smith, Carlos y Corripio, Armando. Principles and Practice of Automatic Process Control, John Wiley & Sons, New York, 1997.

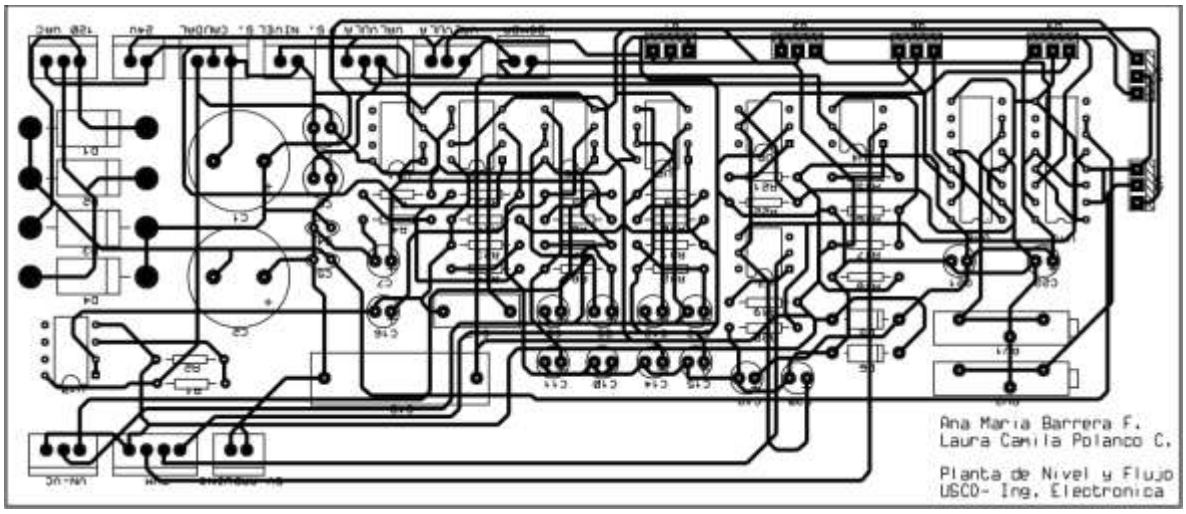
320W Sigle Output With PFC Function SP 320 Series. Mean Well. Disponible en web <http://www.meanwell.com/search/SP-320/SP-320-spec.pdf>

Water Flow Sensor HZ21WA. Disponible en <http://www.microelectronicos.com/datasheets/UCTS0058.pdf>

# ANEXOS

## ANEXO A. Circuito de acondicionamiento de señales.



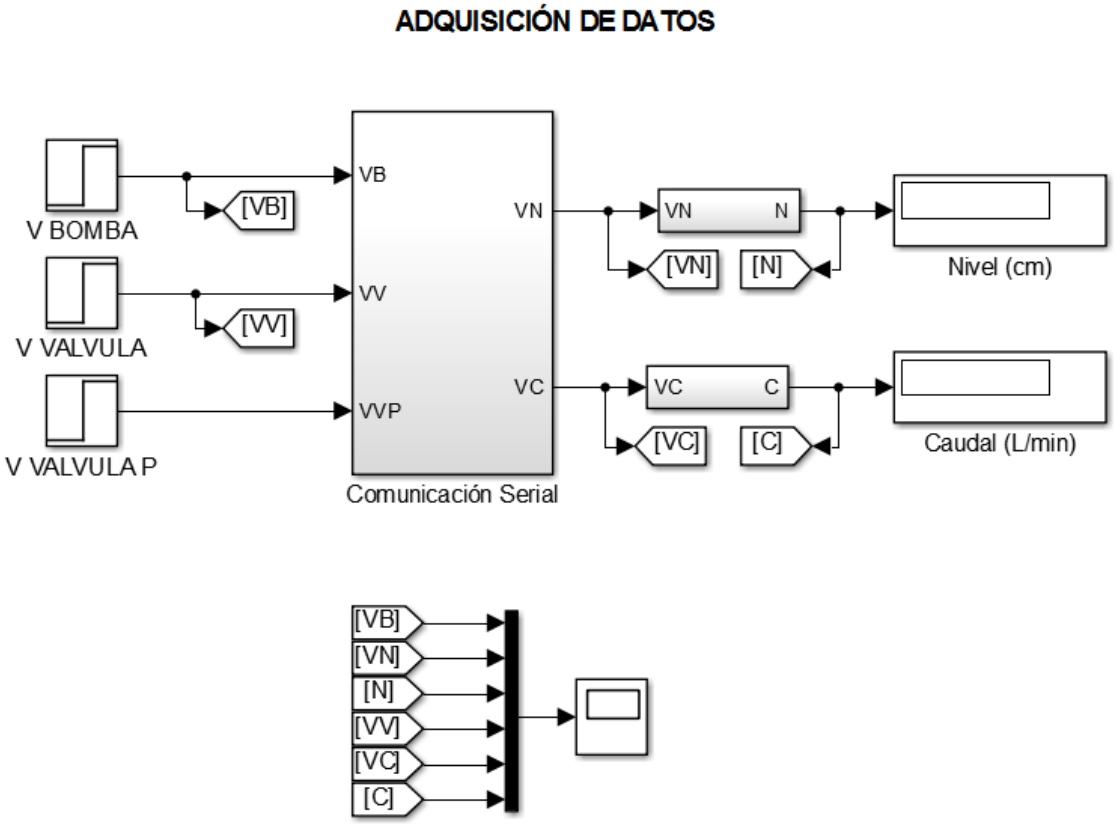

Ana Maria Barrera F.
Laura Camila Polanco C.
Planta de Nivel y Flujo
USCO- Ing. Electronica

**ANEXO B. Comunicación serial (código de Arduino).**

```
// COMUNICACION SERIAL

int out1 = 0; //Sensor de nivel
int out2 = 0; //Sensor de caudal
byte in1 = 0;
byte pinOut1 = 9; //Bomba
byte in2 = 0;
byte pinOut2 = 10; //Valvula
byte in3 = 0;
byte pinOut3 = 11; //Valvula Perturbacion

void setup() {
  // inicializar puerto serie
  Serial.begin(9600);
   //preparar output
   pinMode(pinOut1, OUTPUT);
   pinMode(pinOut2, OUTPUT);
   pinMode(pinOut3, OUTPUT);
}

void loop() {
  // leer del pin A0 (Sensor de nivel) como
  out1 = analogRead(A0);
  // escalar para obtener formato uint8
  out1 = map(out1, 0, 1023, 0, 255);
  Serial.write(out1);

  delay(20);

  // leer del pin A1 (Sensor de caudal) como
```

```
  out2 = analogRead(A1);
  // escalar para obtener formato uint8
  out2 = map(out2, 0, 1023, 0, 255);
  Serial.write(out2);

   // leer del serie si hay datos
 if(Serial.available()){
    in1 = Serial.read();
    // escribir en el pin 9 (Bomba)
    analogWrite(pinOut1, in1);

    delay(20);

    in2 = Serial.read();
    // escribir en el pin 10 (Valvula)
    analogWrite(pinOut2, in2);

    delay(20);

    in3 = Serial.read();
    // escribir en el pin 11 (Valvula Perturbacion)
    analogWrite(pinOut3, in3);
  }
  // esperar para estabilizar el conversor
  delay(20);
}
```

**ANEXO C. Interfaz de adquisición de datos en Simulink.**



## ANEXO D. Código en Matlab del controlador neuronal.

```
% CONTROLADOR DE LA VARIABLE NIVEL

%% GENERAR PATRONES

%Cargar los valores de VN y VB
nivel=VN';
vbomba=VB';

%Generacion de patrones
p=combvec(nivel,vbomba); %Patrones de entrada

%Identificacion de los patrones
for i=1:length(p)
    nivel0=p(1,i);
    vbomba=p(2,i);
    sim('modelo_nivel');
    enivel(i)=nivel.signals.values(length(nivel.signals.values))-nivel0;
%se calcual el error
    iteraciones=sprintf('patron no %d de %d',i,length(p))
end

Er(1,:)= enivel; %Error de nivel.


%% CREACIÓN DEL CONTROLADOR

innet(1,:)=p(1,:); %Entrada
innet(2,:)=Er; %Error nivel
outnet(1,:)=p(1,:) %Señal de control

%Entrenamiento de la red del controlador
net=newff(minmax(innet), [13 1], {'tansig' 'purelin'}, 'trainlm');
net.trainParam.epochs=200;
net.trainParam.goal=1e-7;
[net tr Y E]=train(net,innet,outnet);
view(net)

gensim(net)




% CONTROLADOR DE LA VARIABLE CAUDAL


% Variables de entrada y salida
out=VC'; %Variable de salida del proceso (Voltaje de Caudal)
```

```
in=VV';  %Variable de entrada del proceso (Voltaje de valvula)

%Creación de la red neuronal
net=newff(minmax(out), [5 1], {'tansig' 'purelin'}, 'trainlm');
net.trainParam.goal=1e-7;
net.trainParam.epochs = 100;
view(net)

%Entrenamiento
[net,tr,Y,E]=train(net,out,in);

gensim(net)
```

## ANEXO E. Interfaz de control en Simulink.