Monografía Final

Mi título

Universidad de Buenos Aires
Facultad de Ingeniería

Alumno:	Ignacio Ezequiel Cavicchioli
Padrón:	109428
Email:	icavicchioli@fi.uba.ar

20 de diciembre de 2025

Resumen

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren,

Palabras clave: Lorem ipsum dolor sit amet.

Índice

1. Introducción

Las redes neuronales en sus múltiples formas constituyen sistemas no lineales con un amplio alcance de aplicación en campos como la biología, neurociencia y, al que se avoca este trabajo, aprendizaje automático (*machine learning*, ML). En particular, nos interesa centrarnos en las aplicaciones de las redes neuronales en el campo de control automático. Esta doctrina se encarga del diseño sistemas para regular, guiar o estabilizar procesos de manera autónoma, mediante la realimentación y corrección continua de errores.

La denominada "caja de herramientas" de aquellos en el área de control está compuesta por ciertos artefactos matemáticos que permiten encarar estos problemas, como la linealización de un sistema, el control PID, realimentación de estados, *loop-shaping*, observadores, etc. Lo que no se ha tocado en las materias de control son las estrategias no lineales. En líneas generales, todos los sistemas reales exhiben cierto grado de no linearidad (citar CONTROL SYSTEM DESIGN Goodwin Graebe Salgado, p551, cap19), lo que implica que las estrategias de control lineales son válidas siempre que las no linealidades sean despreciables. Análogamente, las herramientas de identificación de sistemas basadas en la linealización de un sistema fallaran en modelar las no linealidades de estos (Como PCA vs los *autoencoders*).

Este trabajo va a trabajar sobre el uso de redes neuronales en la doctrina de control automático, como identificacores de sistemas, controladores

VER SI QUERËS METER ALGO DEL SOM QUE SALIÖ MAL Y DE OBSERVADORES Y DE GAIN SCHEDULING

2. Sistema elegido

Que tamaños Porque lo bueno lo malo la matemática -¿espacio de estados no lineal y linealizado, transferencia (lineal). el simulink

El sistema elegido está compuesto por 2 tanques de agua de dimensiones diferentes. Tanto la tubería que une los tanques como la que sale tienen cierto coeficiente hidráulico asociado a su geometría. Los valores se eligieron para que el sistema tenga sentido físico aunque no es un requisito. Se podría pensar que el sistema actúa como una cisterna amortiguadora de fluctuaciones en el caudal seguido de un reservorio que ajusta el caudal de salida.

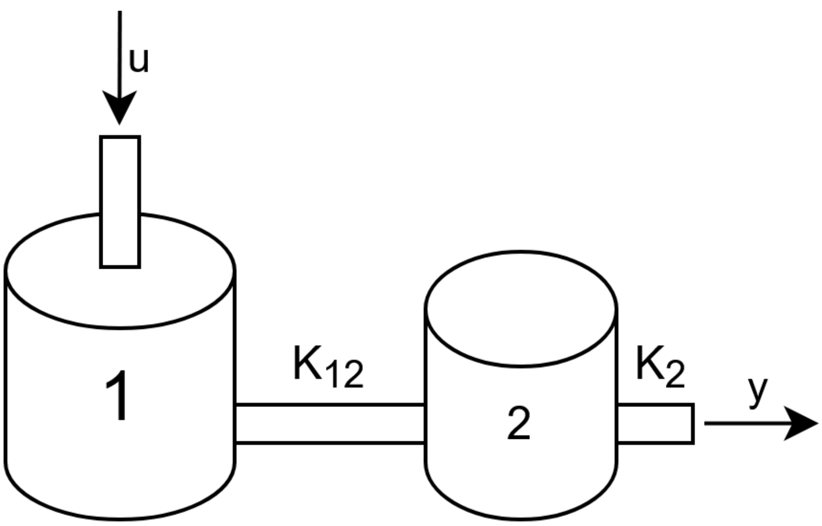

Figura 1: Sistema elegido

La figura 1 muestra el sistema recién descripto, agregando los caudales de entrada y salida u

e y. Ambos caudales son muestreados a 1 Hz, que debería ser más que suficiente para este tipo de dinámicas lentas.

Las razones por la cuales se eligió este sistema son:

- Simplicidad: Es un sistema de 2 estados, simple de modelar, linealizar, simular y controlar. Las redes quese prueben deberían poder con este problema.
- No linealidad: Como se va a ver en el inciso matemático, el sistema no es lineal, que sería un requisito si se está intentando evaluar la capacidad de copiar no linealidades de las redes neuronales.
- Realismo: Se prefirió elegir un sistema que sea fácil de entender pero real, no una abstracción de un sistema más complejo.

2.1. Modelo matemático

El modelado de este sistema hace uso de varias leyes físicas de la hidráulica. Primero se plantea que el líquido es incompresible, por lo que el volumen solo varía si los caudales de entrada y salida no son iguales.

$$\frac{dV}{dt} = \sum q_{in} - \sum q_{out} \tag{1}$$

Además, el volumen es función del área del tanque y su superficie (en este caso que el área es independiente del nivel de agua). Podemos derivar respecto del tiempo.

$$V(t) = A \cdot h(t) \xrightarrow{d/dt} \frac{dV}{dt} = A \cdot \frac{dh(t)}{dt} \tag{2}$$

Luego se aplica la ley de caudal, que relaciona el caudal entre 2 puntos con la diferencia de nivel entre ellos mismos.

$$q = k\sqrt{2g}\sqrt{\Delta h} \tag{3}$$

El caudal entre los tanques resulta:

$$q_{12}(t) = k_{12}\sqrt{2g}\sqrt{h_1(t) - h_2(t)} \tag{4}$$

El caudal de salida del segundo tanque, que también es la salida y, es:

$$q_2(t) = y(t) = k_2\sqrt{2g}\sqrt{h_2(t)} \tag{5}$$

Ahora, se plantea un balance en cada tanque igualando (1) y (2).

$$A_1 \cdot \frac{dh_1}{dt} = u(t) - q_{12}(t) \tag{6}$$

$$A_2 \cdot \frac{dh_2}{dt} = q_{12}(t) - q_2(t) = q_{12}(t) - y(t) \tag{7}$$

sustituyendo con (4) :

$$A_1 \cdot \frac{dh_1}{dt} = u(t) - k_{12}\sqrt{2g}\sqrt{h_1(t) - h_2(t)} \tag{8}$$

$$A_2 \cdot \frac{dh_2}{dt} = k_{12}\sqrt{2g}\sqrt{h_1(t) - h_2(t)} - k_2\sqrt{2g}\sqrt{h_2(t)} \tag{9}$$

Con todo esto se plantea el espacio de estados **no lineal** y continuo con la forma de a continuación:

$$\frac{d}{dt}\begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} (u - k_{12}\sqrt{2g}\sqrt{h_1 - h_2}) \cdot \frac{1}{A_1} \\ (k_{12}\sqrt{2g}\sqrt{h_1 - h_2} - k_2\sqrt{2g}\sqrt{h_2}) \cdot \frac{1}{A_2} \end{bmatrix} \tag{10}$$

$$y = k_2\sqrt{2g}\sqrt{h_2} \tag{11}$$

Las expresiones (10) y (11) se van a linealizar en torno al punto de equilibrio (x_e, u_e) y las constantes físicas indicadas abajo.

- $g = 9{,}81$
- $A_1 = 0{,}342$, $A_2 = 0{,}126$
- $k_{12} = 5 \times 10^{-4}$, $k_2 = 1 \times 10^{-3}$
- $h_{1s} = 1{,}019$, $h_{2s} = 0{,}204$
- $u_s = 0{,}002$

Las variables de estado elegidas se redefinen como variaciones en torno a ese mismo estado, por lo que de ahora en más las alturas h_1 y h_2 no son las mismas que en la planta no lineal. El proceso arranca planteando la linealización en sí, que se ve en la ecuación (12). La expresión (13) se cumple por definición del punto (x_e, u_e). En (14) y (15) se obtiene la matriz A del espacio de estados lineal. En (16) se obtiene la matriz B, y en (18), la C.

Las expresiones (17) y (18) constituyen el espacio de estados lineal para la dinámica de los tanques.

$$\overset{\circ}{X} = \overset{\circ}{\begin{bmatrix} h_1 \\ h_2 \end{bmatrix}} = f(x_e, u_e) + \frac{df}{dx}|_{(x_e,u_e)}(x - x_e) + \frac{df}{du}|_{(x_e,u_e)}(u - u_e) \tag{12}$$

$$f(x_e, u_e) = 0 \tag{13}$$

$$\frac{df}{dx}|_{(x_e,u_e)} = \begin{bmatrix} \frac{-k_{12}\sqrt{2g}}{A_1 \cdot 2\sqrt{h_1 - h_2}} & \frac{k_{12}\sqrt{2g}}{A_1 \cdot 2\sqrt{h_1 - h_2}} \\ \frac{k_{12}\sqrt{2g}}{A_2 \cdot 2\sqrt{h_1 - h_2}} & \frac{-k_{12}\sqrt{2g}}{A_2 \cdot 2\sqrt{h_1 - h_2}} - \frac{k_2\sqrt{2g}}{A_2 \cdot 2\sqrt{h_2}} \end{bmatrix} |_{(x_e,u_e)} \tag{14}$$

$$\frac{df}{dx}|_{(x_e,u_e)} = \simeq \begin{bmatrix} -0{,}00359 & 0{,}000359 \\ 0{,}00976 & -0{,}04878 \end{bmatrix} \tag{15}$$

$$\frac{df}{du}|_{(x_e,u_e)} = \begin{bmatrix} \frac{1}{A_1} \\ 0 \end{bmatrix} |_{(x_e,u_e)} = \begin{bmatrix} 2{,}92 \\ 0 \end{bmatrix} \tag{16}$$

$$\overset{\circ}{\begin{bmatrix} h_1 \\ h_2 \end{bmatrix}} = \begin{bmatrix} -0{,}00359 & 0{,}000359 \\ 0{,}00976 & -0{,}04878 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} + \begin{bmatrix} 2{,}92 \\ 0 \end{bmatrix} u \tag{17}$$

$$y = \begin{bmatrix} 0 & 0{,}0049 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} \tag{18}$$

Además de usar este modelo matemático, se estimó un espacio de estados de misma dimensión a partir del *dataset* de 14400 muestras, para comparar con las redes.

2.2. Simulaciones

Para esta monografía se decidió hacer uso de *scripts MATLAB* y el entorno de *Simulink* debido a la versatilidad que trae en lo que es simulación de sistemas, además que es la herramienta usada en las materias de control automático. La figura 2 muestra el sistema no lineal armado. En pocas palabras, usa las constantes definidas en el *workbench*, la entrada u y los estados integrados para calcular el siguiente estado de la planta.

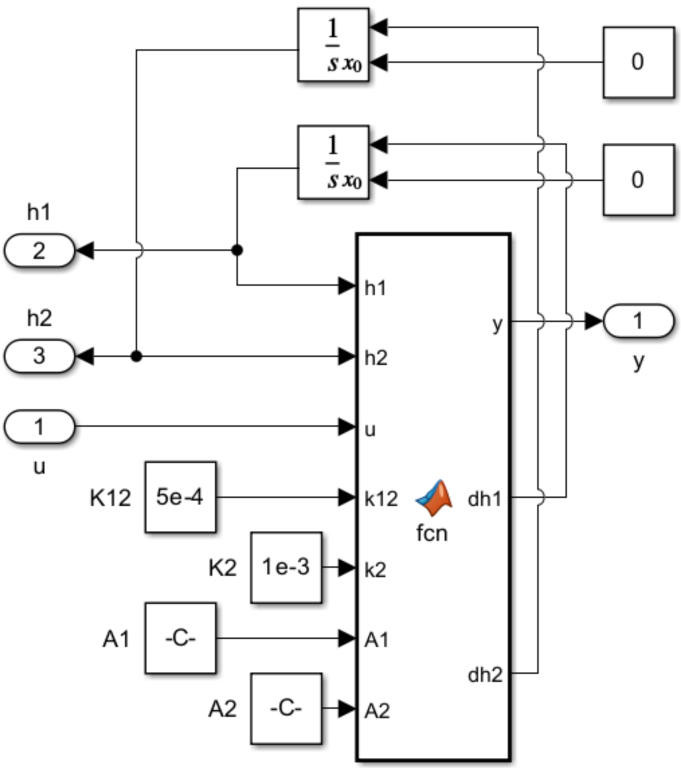

Figura 2: Planta no lineal armada en *simulink*

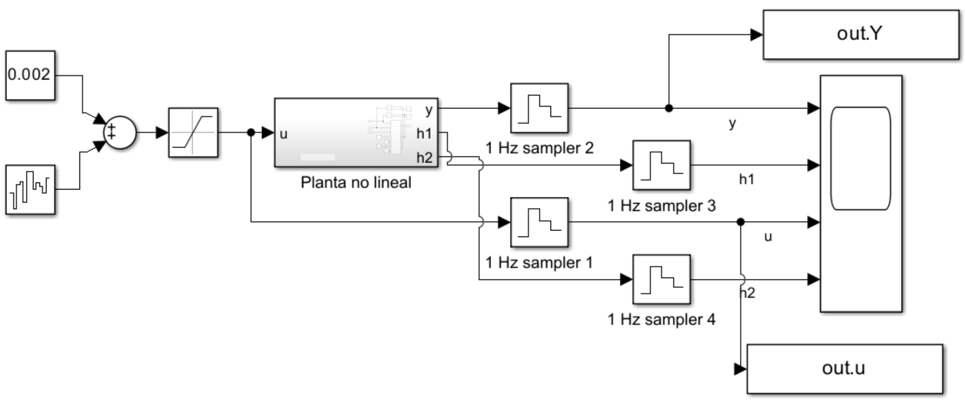

Figura 3: Muestreo de planta no lineal

La figura 3 muestra el sistema de *simulink* que emula el muestreo de la planta. De entrada u se usó una secuencia de escalones de amplitudes aleatorias (ruido blanco gaussiano) alrededor del 10 % del punto operativo. Esto se hizo para poder tener una buena variedad de respuestas al escalón, que se espera ayude a que la red aprenda la dinámica.

Como el muestreo se realizó a 1 Hz, se decidió generar un *dataset* de 4 horas (14400 muestras), y otro de 1 hora (3600 muestras). La intención es observar cuanto empeora la *performance* de la red con menos muestras de entrenamiento.

3. Identificación de sistemas

La primer experiencia de esta monografía consiste en entrenar una red neuronal en base a los datos de la planta elegida, con la finalidad de obtener un sistema no lineal que la emule.

3.1. Suposiciones y decisiones

Con el objetivo de asegurar resultados conmensurables, los experimentos se armaron y realizaron siguiendo los siguientes preceptos:

1. La única métrica de comparación elegida es el RMSE (ec. (**??**)). Se calculará sobre las secuencias de salida de todas las redes entrenadas.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{x}-x)^2} \tag{19}$$

2. El RMSE se calculará sobre la secuencia de datos de salida pero sin incluir el transitorio inicial. Esto es porque el transitorio inicial no está bien representado en los datos de entrenamiento.

3. Todos los modelos deberán poder usarse de forma auto-regresiva, es decir, realimentados con sus propias salidas. Esto implica que el entrenamientos se va a tener que realizar hasta que se observe convergencia en la simulación.

4. La cantidad de *epochs* y la cantidad de intentos de entrenamiento no fueron fijadas, sino que se ajustaron empíricamente, asegurando el cumplimiento del ítem 2.

5. Los datos se separaron en *train* y *test* en una proporción de 85 % a /15 %. La separación se realizó de forma secuencial, es decir, se tomo el primer 85 % de las muestras para entrenar y el resto para evaluar, manteniendo cierta semblanza al orden temporal. Se podría argumentar que las muestras de evaluación no necesariamente se van a distribuir idénticamente que las de entrenamiento, pero se priorizó variedad de muestras, y el hecho de que los escalones se generaron de forma aleatoria debería ayudar.

6. Se aplicó una normalización en media y varianza a los datos previo a entrenar. Se espera que esto mejore la *performance* (a entradas más espaciadas, pesos más separados) y la convergencia en el *fitting*.

Con esto dicho, se procede a los ensayos.

3.2. MLP de 1 capa oculta

3.2.1. Introducción

El primer *approach* pensado es usar un MLP de 1 capa oculta y entrenarlo en base a los datos de la planta en una configuración auto-regresiva como la de la figura 4. Dado lo visto en la materia, se espera que el MLP logre captar las no-linealidades de la planta con una capacidad que aumenta con la cantidad de neuronas en su capa oculta. En palabras más simples, se espera que los modelos más chicos tengan peor desempeño que los grandes.

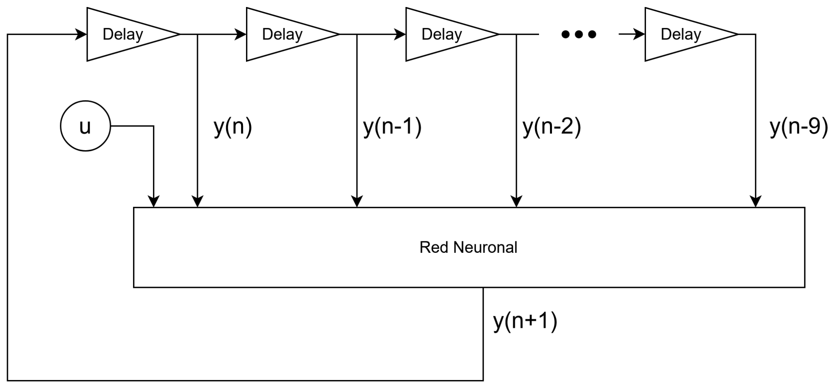

Figura 4: Modelo de red

El vector de datos de entrada $\mathbf{x}(n)$ elegido consta de 10 muestras previas de y y la referencia actual u.La salida esperada de la red es la siguiente muestra, efectivamente entrenando un predictor del siguiente estado.

$$\mathbf{x}(n) = \big[y(n),\ y(n-1),\ \ldots,\ y(n-9),\ u(n)\big]$$

$$\hat{y}(n+1) = f\big(y(n),\ y(n-1),\ \ldots,\ y(n-9),\ u(n)\big)$$

La razón por la cual se eligieron 10 muestras previas de y es porque se considera que, siendo un proceso lento, la red se beneficiaria de poder ver múltiples estados previos, notando como varía la salida en el tiempo. Si se usaran pocas muestras previas, el vector de entrada sería un apilamiento de números casi idénticos para los escalones de entrada pequeños.

3.2.2. Entrenamiento

Como se adelantó previamente, se decidió crear con 2 *datasets* así que los experimentos van a verse duplicados. Para esta parte se decidió entrenar redes de 20, 10, 5 y 2 preceptrones en la capa oculta.

Algunos puntos interesantes a destacar del entrenamiento son:

- Se entrenó por una cantidad fija de *epochs*, permitiendo interrumpir cuando el gradiente era menor a $1e-9$ (que nunca sucedió).
- El proceso de entrenamiento fue manejado por la función *train*[1] del *Deep learning toolbox* de *MATLAB*.
- En general, los modelos pequeños requirieron de más intentos de entrenamiento para llegar a un sistema que cumpliera con los requisitos planteados. El parámetro modificado fue cantidad de *epochs*.
- Hubo casos en los que los sistemas obtenidos tenían un estado estacionario oscilante, invariante respecto la acción de control. Esto fue considerado un espurio llamativo, aunque lamentablemente no se guardaron imágenes.
- El modelo lineal estimado se obtuvo usando la herramienta de identificación de sistemas propia de *MATLAB*.

[1] Documentación oficial

3.2.3. Resultados

Las figuras en esta sección muestran las señales de interés, incluyendo la entrada u, la salida y real, la y del sistema linealizado y la salida de cada red entrenada. La respuesta real (fig. 6) y las respuestas de las redes neuronales son casi indistinguibles, salvo por los transitorios iniciales. Sin embargo, lo mismo sucede para las respuestas obtenidas de los sistemas lineales.

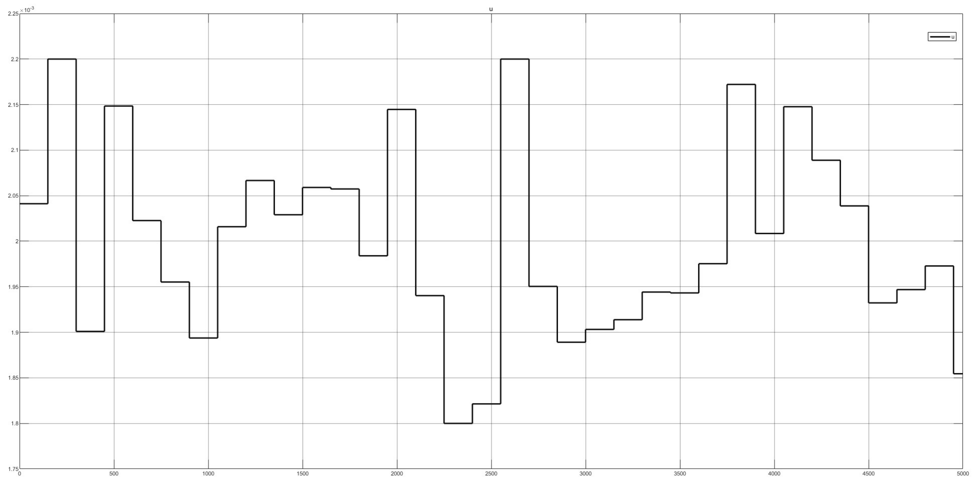

Figura 5: Entrada u

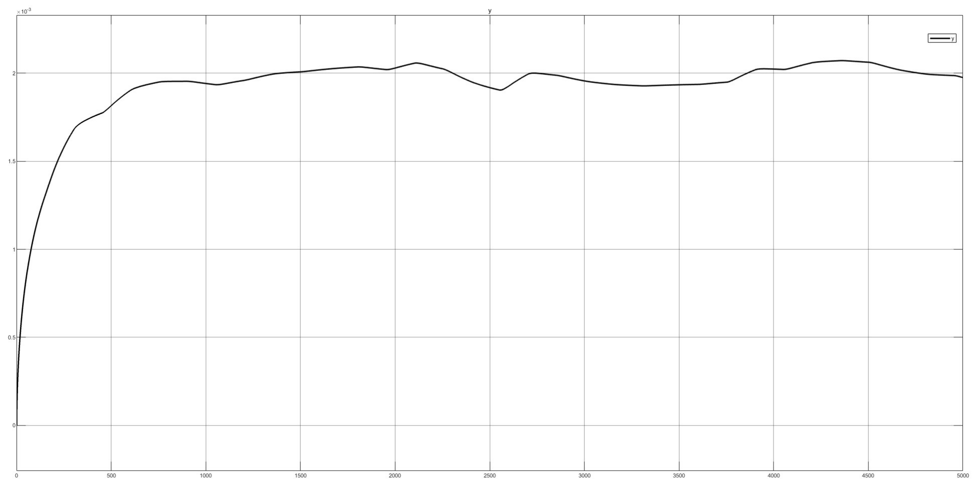

Figura 6: Salida y de la planta no lineal

A continuación se presenta un cuadro con los RMSE de todas las respuestas, calculado usando de referencia la salida del sistema no lineal original. Las secuencias fueron tomadas a partir de la muestra número 2000, evitando los transitorios, tal como se dijo que se iba a hacer.

3.2.4. Análisis de resultados

A partir de los resultados expuestos en el cuadro 1, se notan tendencias en la *performance* de los varios modelos analizados. Primero, los modelos entrenados en base a más muestras (14400) tienen RMSE inferiores a los entrenados en base al set de 3600 muestras - que es consistente con las hipótesis antes mencionadas de la capacidad de captar no-linealidades.

En segundo lugar, el sistema lineal obtenido matemáticamente (apodado “real” en el cuadro), logró mejores ajustes que dos redes por un orden de magnitud. Esto puede deberse a que el modelo lineal es muy bueno o que las redes resultaron *sub-par*. Dicho esto, es inferior

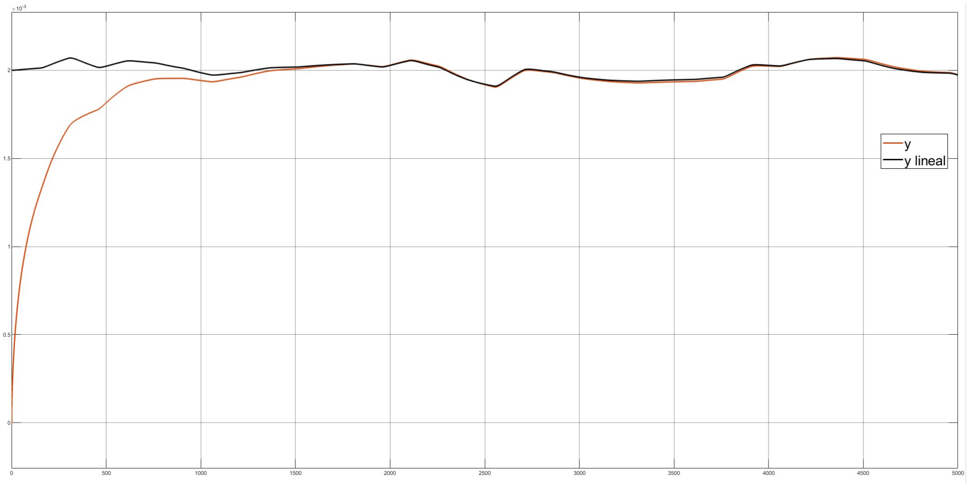

Figura 7: Comparación entre salida no lineal y lineal

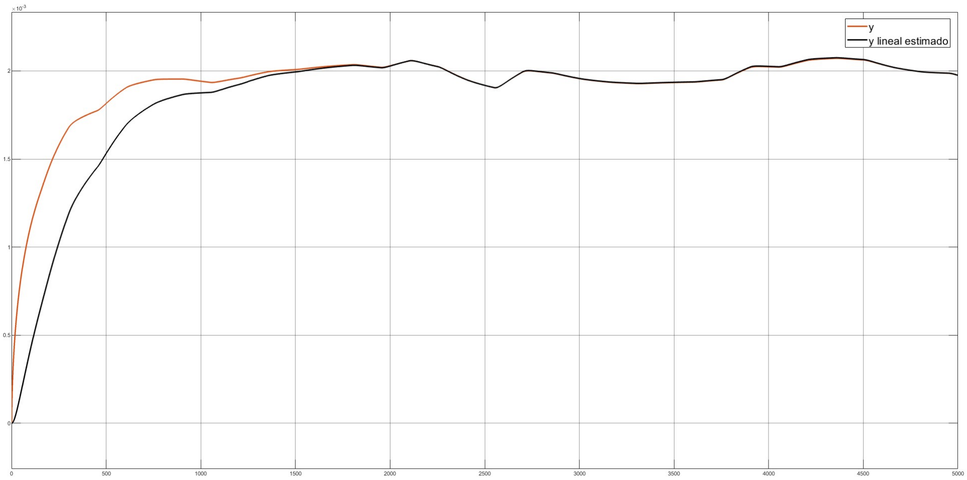

Figura 8: Comparación entre salida no lineal y lineal del modelo estimado

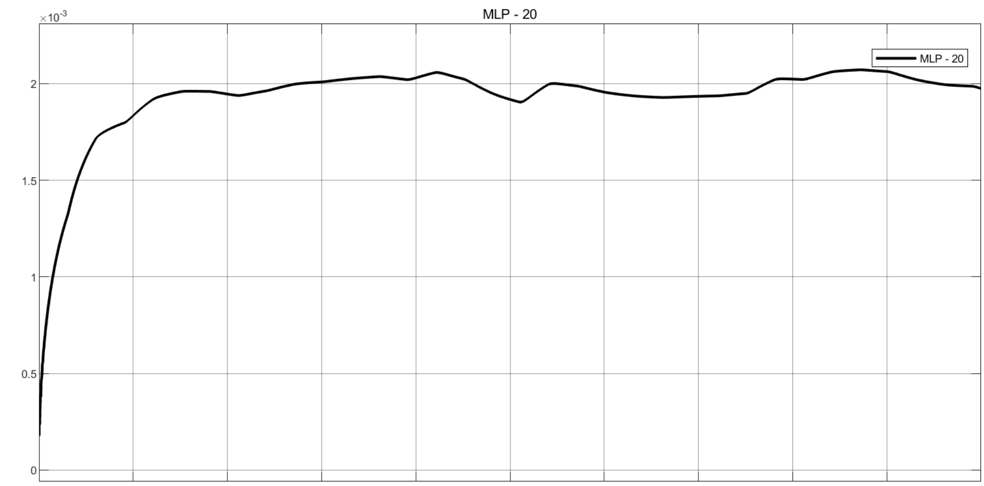

Figura 9: Salida del MLP de 20 neuronas en capa oculta - *dataset* de 14400 muestras

que el resto de las redes, incluyendo 2 entrenadas con el *dataset* pequeño, dando a entender que, incluso con pocas muestras se puede lograr un modelo no lineal mejor que uno simple derivado matemáticamente. Esto representa un punto a favor de la identificación del tipo "caja

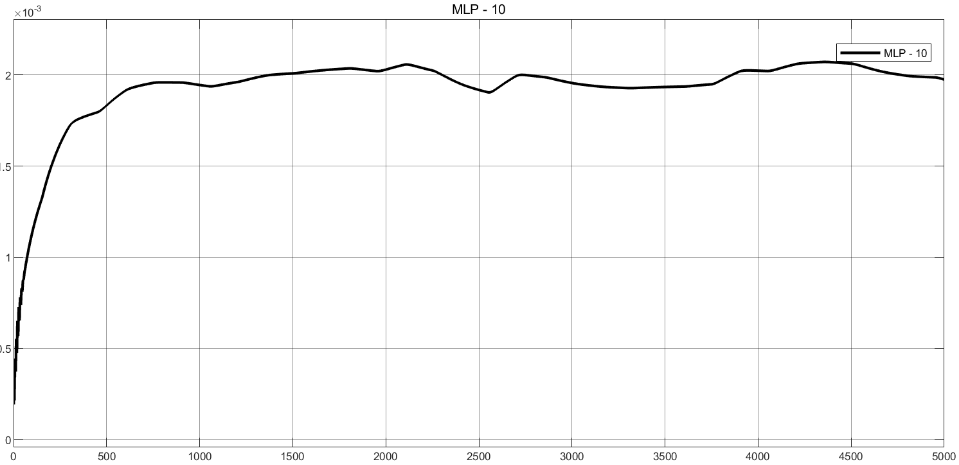

Figura 10: Salida del MLP de 10 neuronas en capa oculta - *dataset* de 14400 muestras

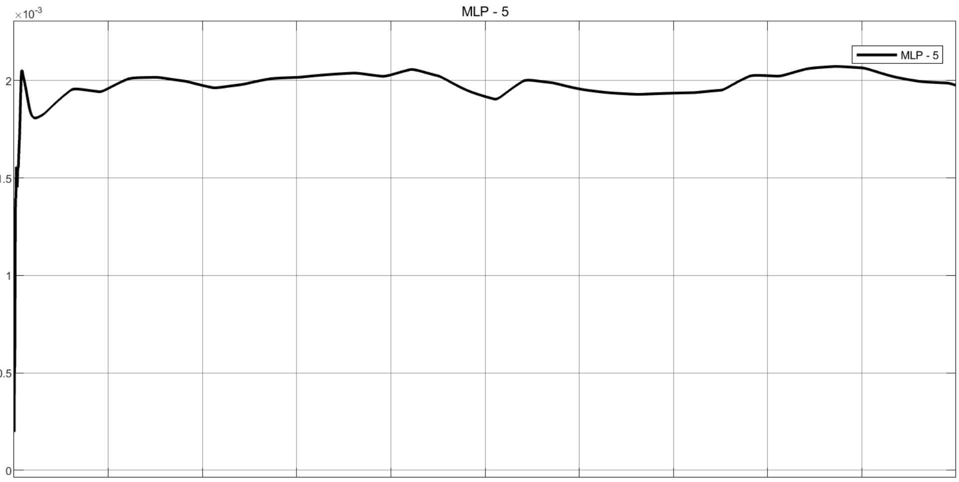

Figura 11: Salida del MLP de 5 neuronas en capa oculta - *dataset* de 14400 muestras

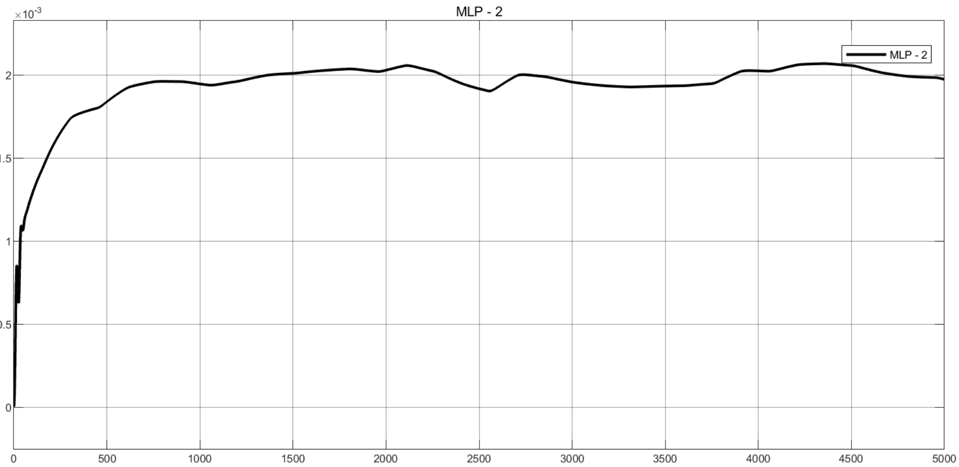

Figura 12: Salida del MLP de 2 neuronas en capa oculta - *dataset* de 14400 muestras

negra/gris".

El "podio" de los resultados está compuesto de 3 redes neuronales y el modelo lineal estimado, todos entrenadas con el *dataset* grande . La mejor red supera al modelo lineal por 7,6 veces, y no sorprende que sea la red más grande. Se considera que, al disponer de más muestras,

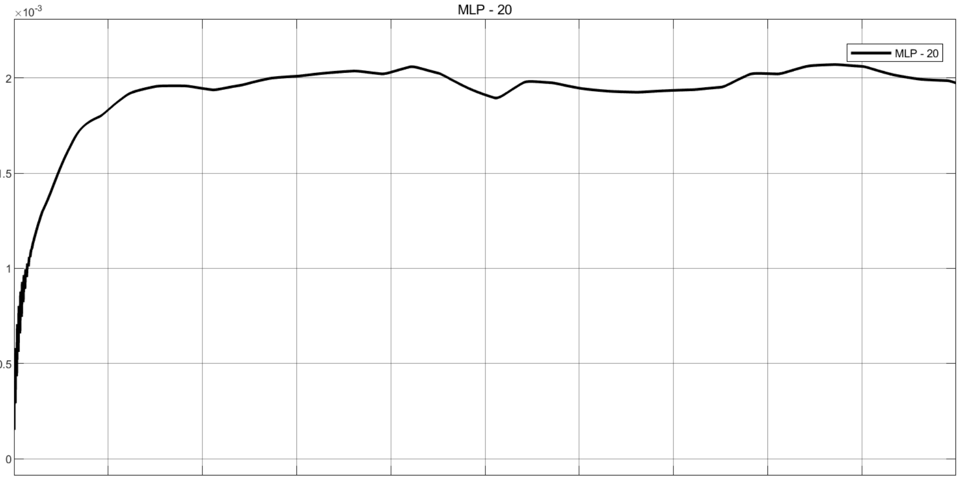

Figura 13: Salida del MLP de 20 neuronas en capa oculta - *dataset* de 3600 muestras

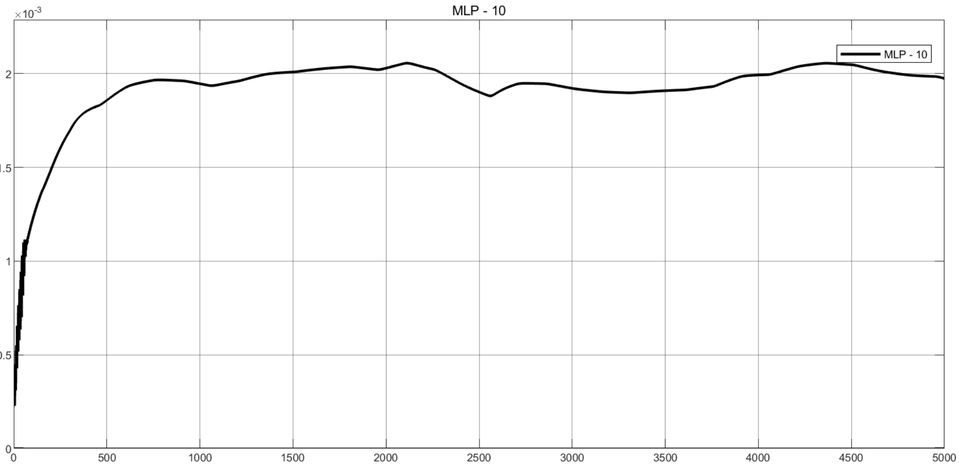

Figura 14: Salida del MLP de 10 neuronas en capa oculta - *dataset* de 3600 muestras

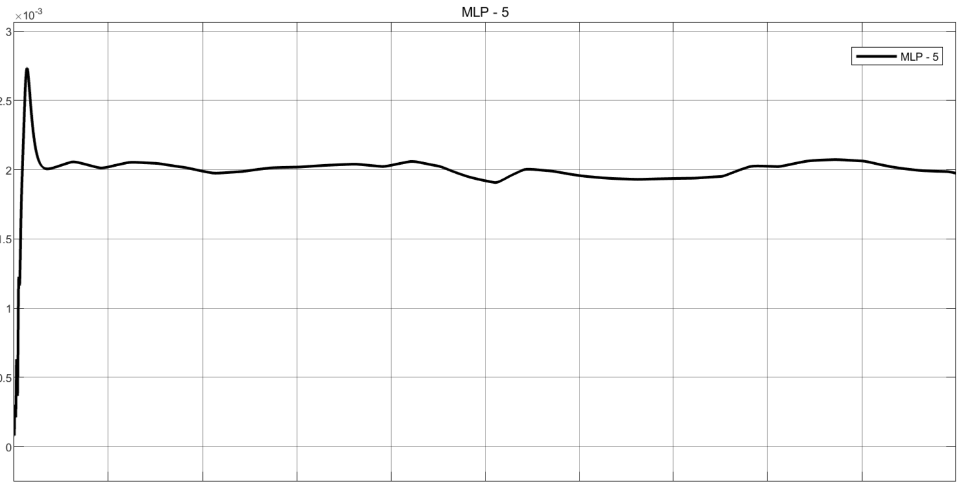

Figura 15: Salida del MLP de 5 neuronas en capa oculta - *dataset* de 3600 muestras

las redes lograron captar las no linealidades del modelo con mayor precisión.

Los resultados confirman que la cantidad de muestras disponibles permite el entrenamiento de modelos más complejos, que llegan a precisiones superiores que sus contrapartes lineales.

Ahora bien, se debe resaltar que ninguno de los modelos obtenidos es inherentemente malo,

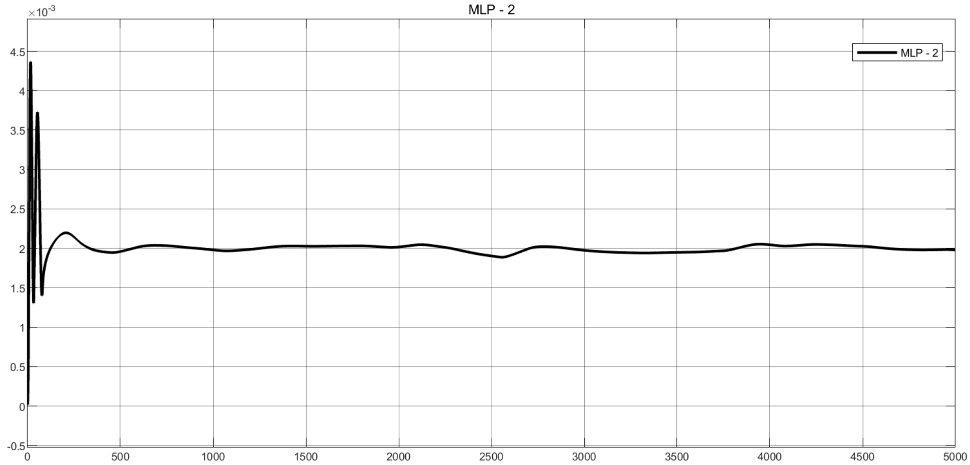

Figura 16: Salida del MLP de 2 neuronas en capa oculta - *dataset* de 3600 muestras

Modelo	RMSE
MLP 20–14400	$2{,}36 \times 10^{-7}$
MLP 10–14400	$3{,}24 \times 10^{-7}$
MLP 5–14400	$1{,}06 \times 10^{-6}$
Modelo lineal (estimado)	$1{,}80 \times 10^{-6}$
MLP 5–3600	$1{,}95 \times 10^{-6}$
MLP 2–14400	$2{,}01 \times 10^{-6}$
MLP 20–3600	$6{,}71 \times 10^{-6}$
Modelo lineal (real)	$6{,}83 \times 10^{-6}$
MLP 2 –3600	$1{,}88 \times 10^{-5}$
MLP 10–3600	$2{,}46 \times 10^{-5}$

Cuadro 1: Comparación del error cuadrático medio (RMSE) entre los distintos modelos evaluados

sino que el tipo de modelo a usar queda determinado por los requerimientos de la aplicación. Los experimentos recién detallados solo indican que una forma de lograr respuestas más similares a la real es por el uso de sistemas no lineales en la identificación.

3.3. Sistema lineal asistido

3.3.1. Introducción

El segundo enfoque ideado surge del anterior. Si se puede lograr un buen desempeño con un modelo lineal, debería ser posible entrenar una red neuronal que estime las salidas no lineales a partir de las lineales.

La figura 17 muestra el sistema armado. Se mantiene el mismo vector de entrada del experimento previo, con la diferencia que ahora las salida previas vienen del sistema lineal y no se predice la siguiente salida, sino que corrige la actual.

Inicialmente, se espera que este sistema logre mejor precisión que si solo se usara el lineal, pero no se puede saber si va a ser mejor que las otras redes. Además, se espera que sea más predecible, dado que una gran parte de la dinámica es fundamentalmente lineal.

3.3.2. Entrenamiento

Graf de performance de la red!

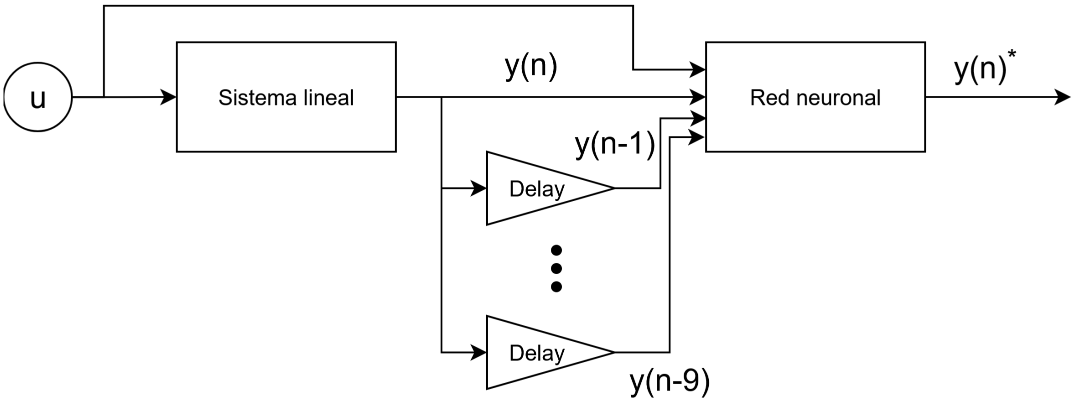

Figura 17: Modelo de red

Se decidió entrenar 3 MLPs de 2, 5, 10 y 20 neuronas en la capa oculta. Se desea observar que complejidad de red se requiere para capturar el mapeo de la salida lineal a la no lineal.

3.3.3. Resultados

3.3.4. Análisis de resultados

4. Control de sistemas

5. Misceláneos

6. Conclusiones

7. Referencias

- Material de la materia "Control Automático".
- Control System Design. Autores: Graham C. Goodwin, Stefan F. Graebe, Mario E. Salgado. Año: 2000
- Neural network (machine learning)
- Machine learning
- Neural network
- Introduction to neural network control systems