

Sanchez Rico Raúl

#25  
Neuronal Network  
3CM2

MLP

PRACTICA 5

**Introduccion**

El algoritmo de Mínimo Cuadrado Medio (Least Mean Square, LMS), introducido por Widrow y Hoff en 1959 , es un algoritmo adaptativo, que utiliza un método basado en gradiente de mayor inclinación. El algoritmo LMS utiliza las estimaciones del vector de gradiente a partir de los datos disponibles. LMS incorpora un procedimiento iterativo que realiza correcciones sucesivas al vector de peso en la dirección del negativo del vector de gradiente que finalmente conduce al error cuadrado medio mínimo. En comparación con otros algoritmos, el algoritmo LMS es relativamente simple; no requiere cálculo de la función de correlación ni requiere inversiones matriciales.

La regla de aprendizaje del perceptrón de Frank Rosenblatt y el algoritmo LMS de Bernard Widrow y Marcian Hoff fueron diseñados para entrenar redes de preceptrones de una sola capa. Sin embargo, las redes de una sola capa sufrieron de la desventaja de que solo eran capaces de resolver problemas de clasificación linealmente separables. Rosenblatt como Widrow estaban al tanto de estas limitaciones, y propusieron redes multicapa que pudieran superarlas, pero no fueron capaces de generalizar sus algoritmos para entrenar estas redes más poderosas.

En la decada de los 80’s el algoritmo backpropagation fue redescubierto y ampliamente publicado. Fue redescubierto independientemente por David Rumelhart, Geoffrey Hinton y Ronald Williams, David Parker, y Yann Le Cun. El algoritmo fue popularizado por su inclusión en el libro *Parallel Distributed Processing* (Procesamiento Paralelo Distribuido), el cual describió el trabajo del grupo de Procesamiento Paralelo Distribuido llevado por los psicólogos David Rumelhart y James McClelland. La publicación de este libro estimuló un torrente de investigación en el campo de las redes neuronales. El perceptrón multicapa, entrenado por el algoritmo backpropagation, es actualmente la red neuronal más ampliamente utilizada.

El algoritmo LMS, llamada backpropagationpuede ser utilizada para entrenar redes multicapa. Al igual que con la ley de aprendizaje LMS, backpropagation es un algoritmo aproximado de descenso más empinado, en el que el índice de rendimiento es el error cuadrático medio. La diferencia entre el algoritmo LMS y backpropagation es solo en la forma en que se calculan las derivadas. Para una red lineal de una sola capa, el error es una función lineal explícita de los pesos de la red, y sus derivadas con respecto a los pesos se pueden calcular fácilmente. En redes de múltiples capas con funciones de transferencia no lineales, la relación entre los pesos de la red y el error es más compleja. Para calcular las derivadas, necesitamos usar la regla de la cadena del cálculo.

Marco Teorcio

**Marco Teorico**

**Backpropagation**

Es uno de los tipos de redes más comunes. Se basa en otra red mas simple llamada perceptrón simple solo que el número de capas ocultas puede ser mayor o igual que una. Es una red unidireccional (feedforward). La arquitectura típica de esta red es la siguiente:

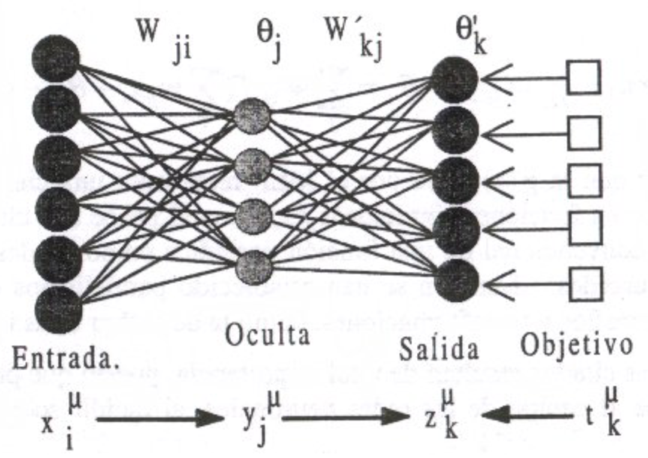


Ilustración 1

Las neuronas de la capa oculta usan como regla de propagación la suma ponderada de las entradas con los pesos sinápticos wij y sobre esa suma ponderada se aplica una función de transferencia de tipo sigmoide, que es acotada en respuesta.

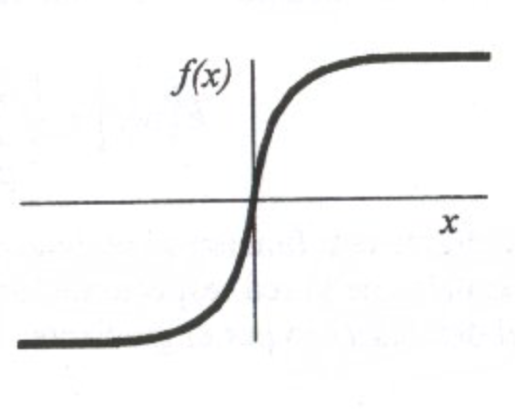


Ilustración 2

El aprendizaje que se suele usar en este tipo de redes recibe el nombre de retropropagacion del error (backpropagation). Como funcion de coste global, se usa el error cuadratico medio. Es decir, que dado un par (xk, dk) correspondiente a la entrada k de los datos de entrenamiento y salida deseada asociada se calcula la cantidad:

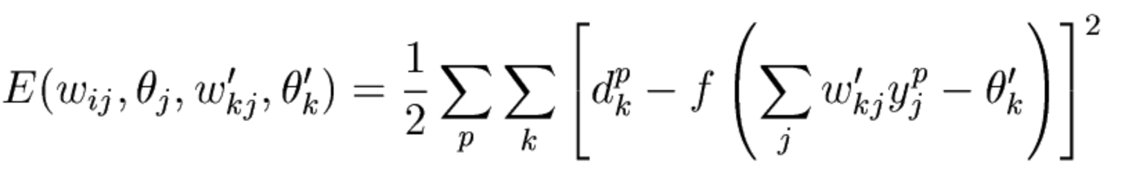


Ilustración 3

vemos que es la suma de los errores parciales debido a cada patrón (índice p), resultantes de la diferencia entre la salida deseada dp y la salida que da la red f(.) ante el vector de entrada xk. Si estas salidas son muy diferentes de las salidas deseadas, el error cuadratico medio sera grande. f es la función de activación de las neuronas de la capa de salida e y la salida que proporcionan las neuronas de la ultima capa oculta.Sobre esta función de coste global se aplica algun procedimiento de minimización. En el caso del MLP se hace mediante un descenso por gradiente. El aprendizaje por backpropagation queda como sigue:

1. Inicializar los pesos y los umbrales iniciales de cada neurona. Hay varias posibilidades de inicialización siendo las mas comunes las que introducen valores aleatorios pequeños.
2. Para cada patrón del conjunto de los datos de entrenamiento
3. Obtener la respuesta de la red ante ese patrón. Esta parte se consigue propagando la entrada hacia adelante, ya que este tipo de red es feedforward. Las salidas de una capa sirven como entrada a las neuronas de la capa siguiente, procesandolas de acuerdo a la regla de propagación y la función de activación correspondientes.
4. Calcular los errores asociados según la ecuación 3-2
5. Calcular los incrementos parciales (sumandos de los sumatorios). Estos incrementos dependen de los errores calculados en 2.b
6. Calcular el incremento total ,para todos los patrones, de los pesos y los umbrales según las expresiones en la ecuación 3-2
7. Actualizar pesos y umbrales
8. Calcular el error actual y volver al paso 2 si no es satisfactorio.

En resumen el percetrón multicapa evoluciona el perceptrón simple y para ello incorpora capas de neuronas ocultas, con esto consigue representar funciones no lineales. El perceptrón multicapa esta compuesto por por una capa de entrada, una capa de salida y n capas ocultasentremedias. Se caracteriza por tener salidas disjuntas pero relacionadas entre sí, de tal manera que la salida de una neurona es la entrada de la siguiente. En el perceptrón multicapa se pueden diferenciar una 2 fases:

* Propagación en la que se calcula el resultado de salida de la red desde los valores de entrada hacia delante.
* Aprendizaje en la que los errores obtenidos a la salida del perceptrón se van propagando hacia atrás (backpropagation) con el objetivo de modificar los pesos de las conexiones para que el valor estimado de la red se asemeje cada vez más al real, este aproximación se realiza mediante la función gradriente del error.

Su arquitectura consta de 3 partes:

* Capa de entrada: conecta la red con el exterior, cada neurona se corresponde con cada una de las variables de entrada de la red.
* Capas ocultas: son una aglomeración de capas en as que cada activación de una salida procede de la suma ponderada de las activaciones de la capa anterior conectadas, mas sus correspondientes umbrales (bias, sesgos).
* Capa de salida: conecta las capas ocultas con la salida de la red que proporciona los resultados.

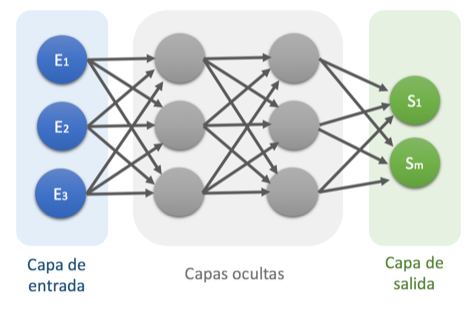


Ilustración 4

**Experimento**

Polinomio de entrada

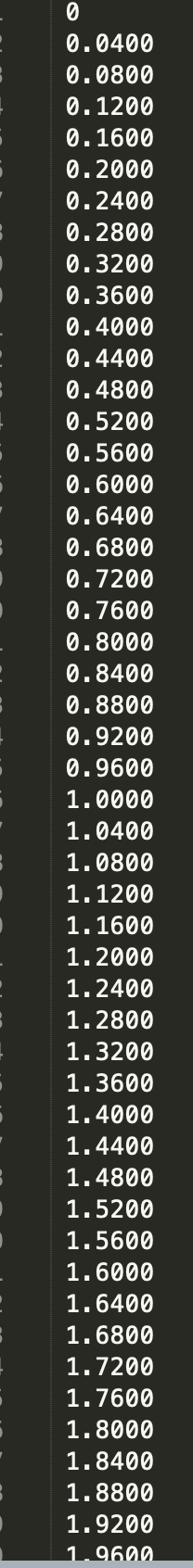
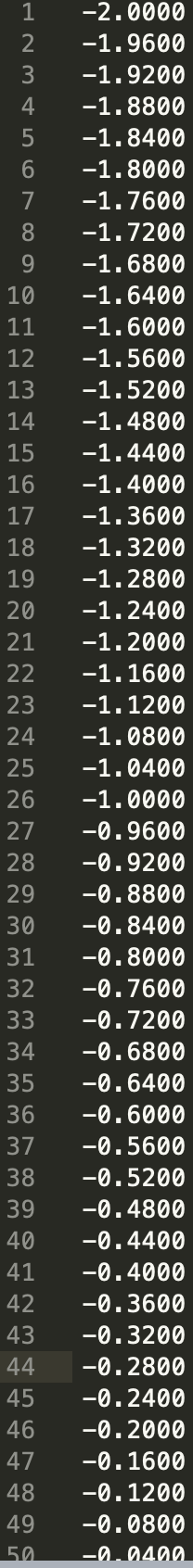


Ilustración 5 Polinomios de entrada

Polinomios Target

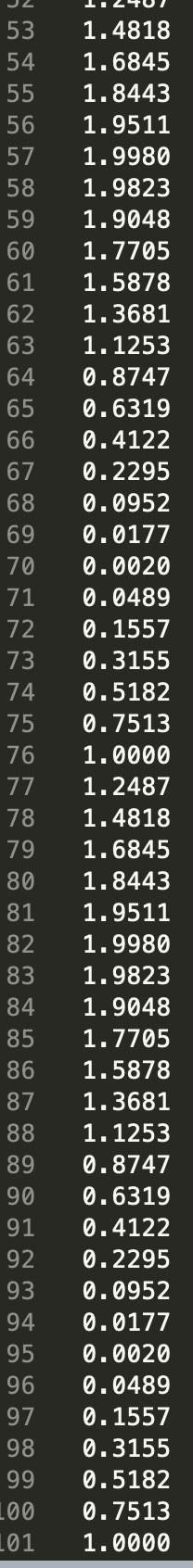
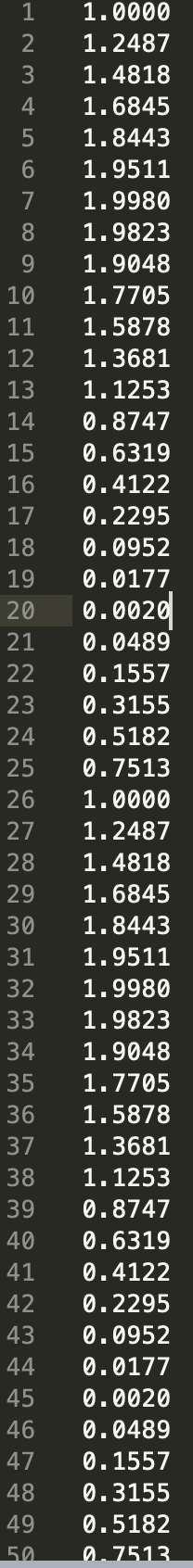


Ilustración 6 Polinomios Target

Para este ejemplo se usarán los respectivos valores de acuerdo con el orden:

* Archivo de entrada = input
* Archivo de valores deseados= target
* Arquitectura del M.L.P = 1 3 1
* Funciones de las capas del M.L.P = 2 1
* Alfa: 0.1 Itmax: 20000 Itval: 1000 Numval: 3
* Eit: 0.0000000000001
* Configuración: 2

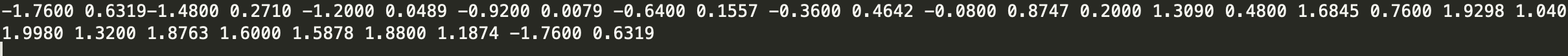


Ilustración 7 Conjunto de validacion

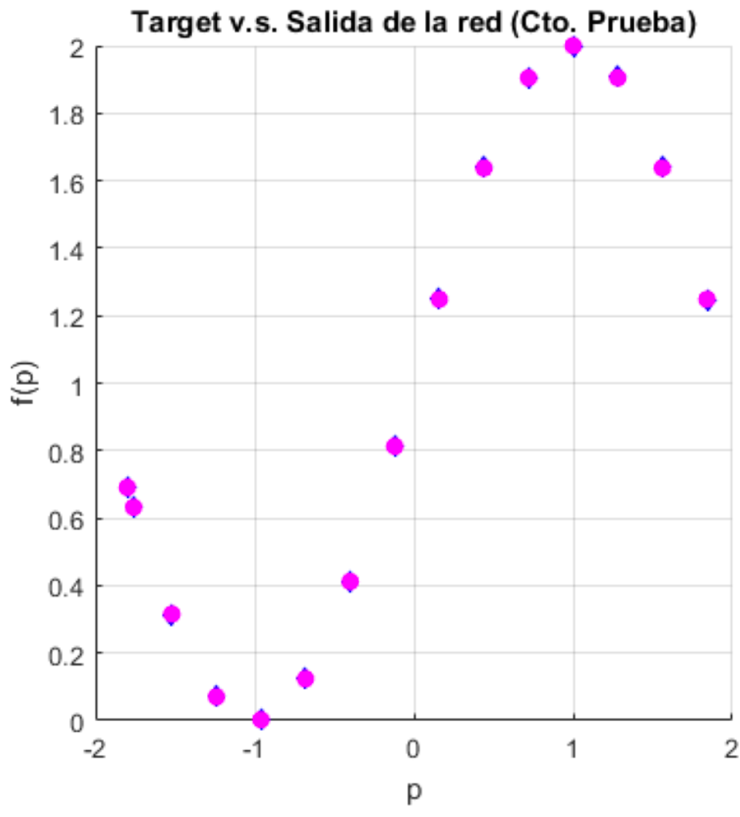


Ilustración 8 Comparación de la evaluación del conjunto de prueba (salida de la red) vs. los targets

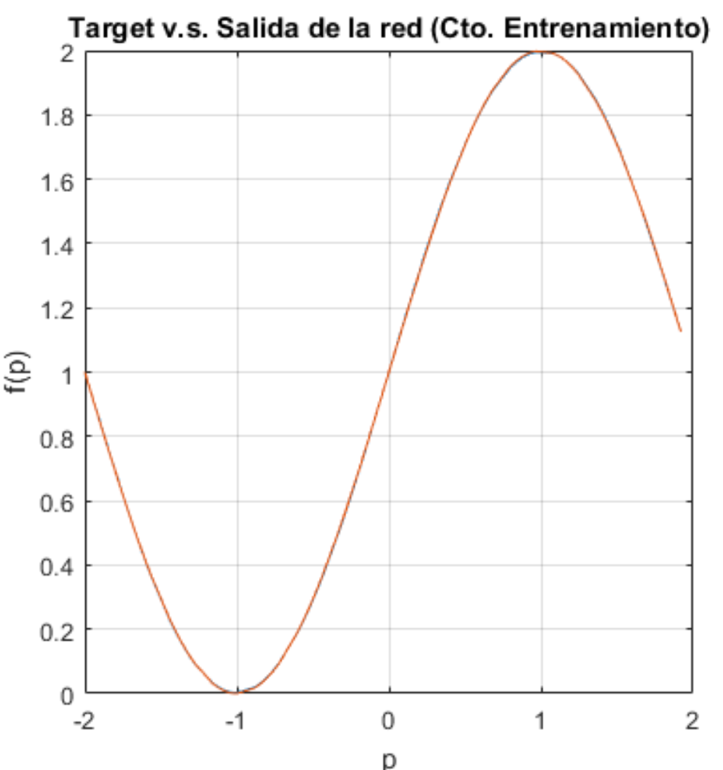


Ilustración 9 Target vs Salida de la red del conjunto de entrenamiento (Se puede apreciar que el error fue mínimo).

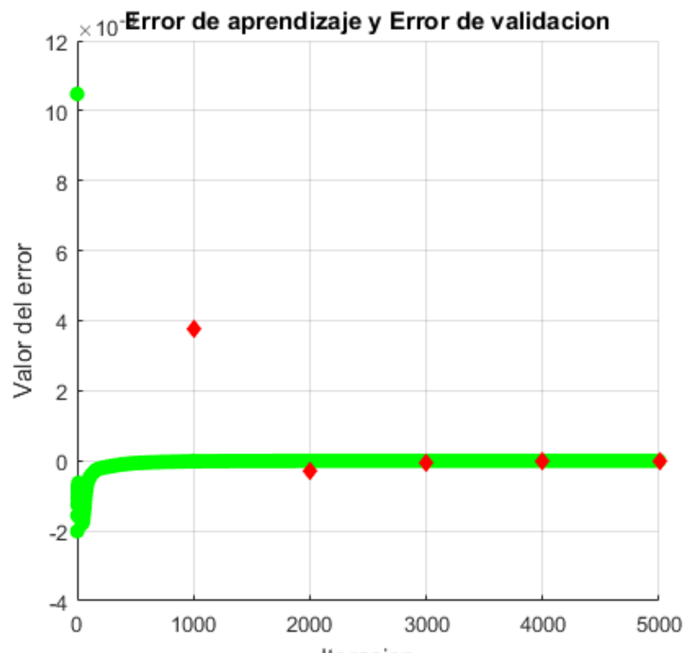


Ilustración 10 Evolución del error de aprendizaje junto con el error de validación.

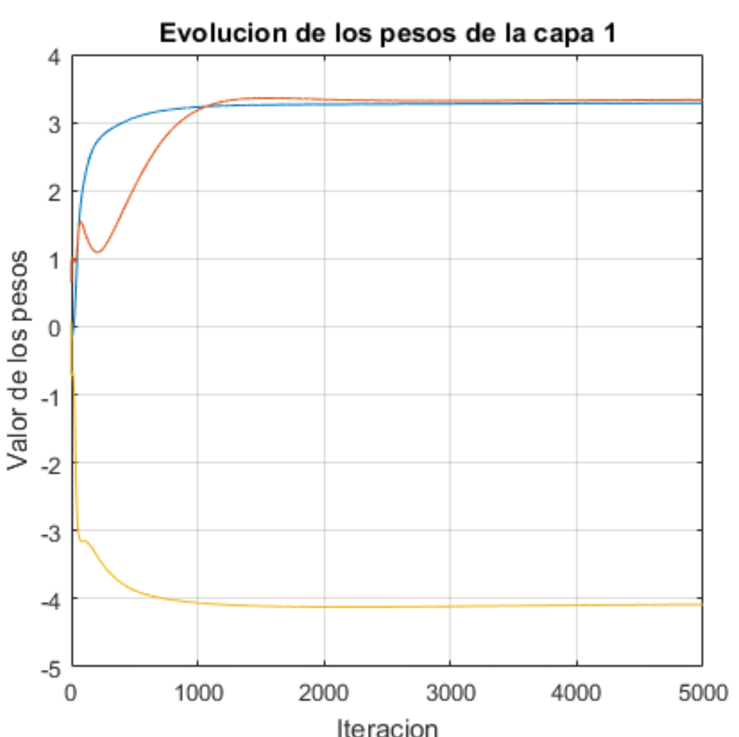


Ilustración 11 Evolución de los pesos de la capa 1.

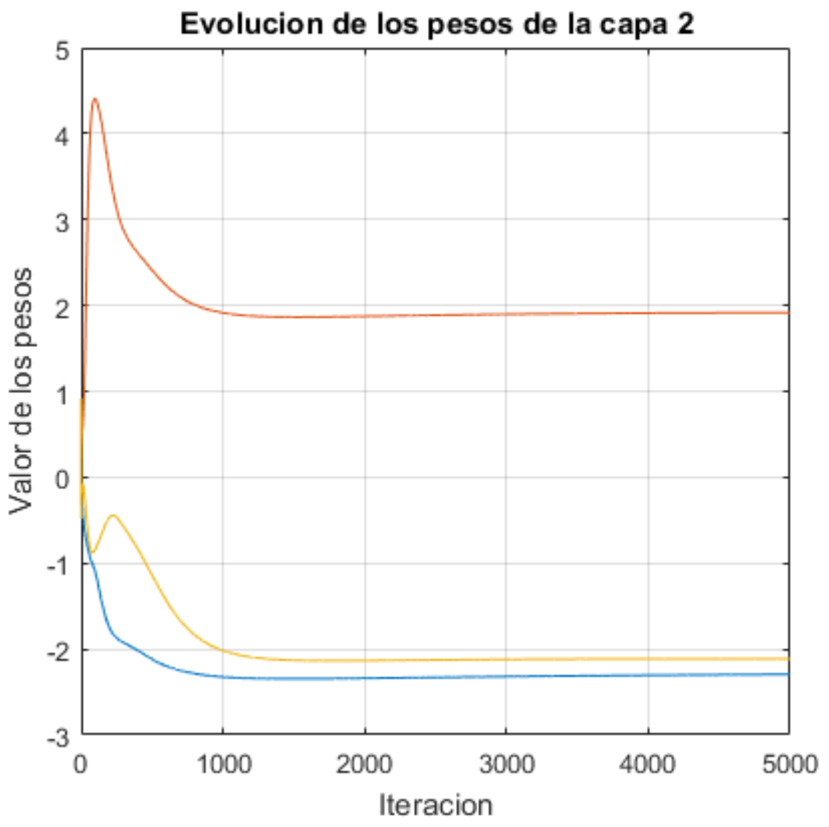


Ilustración 12 Evolución de los pesos de la capa 2.

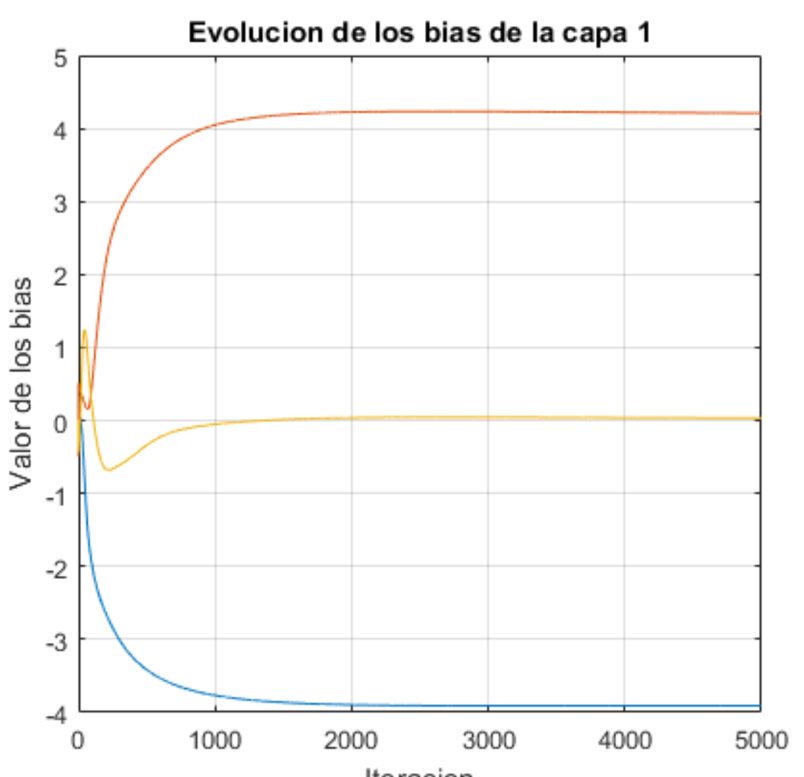


Ilustración 13 Evolución del bias de la capa 1.

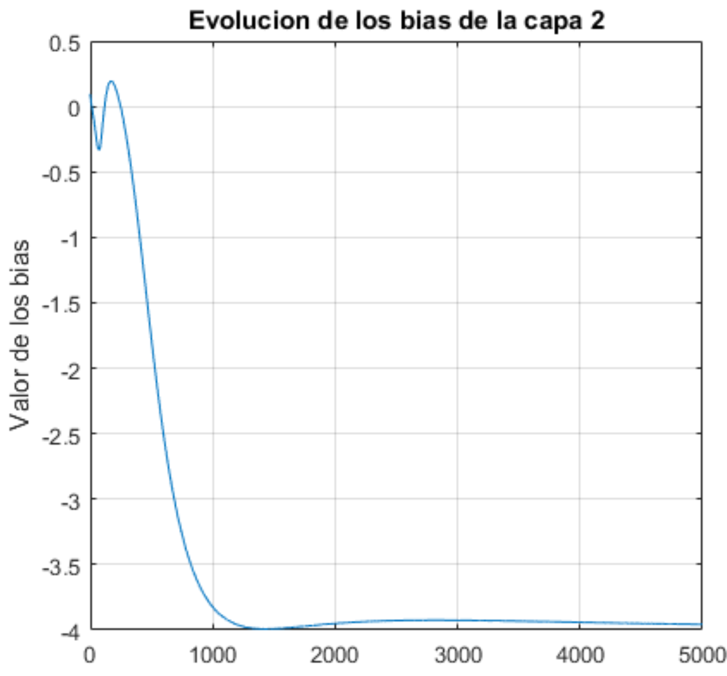


Ilustración 14 Evolución del bias de la capa 2.

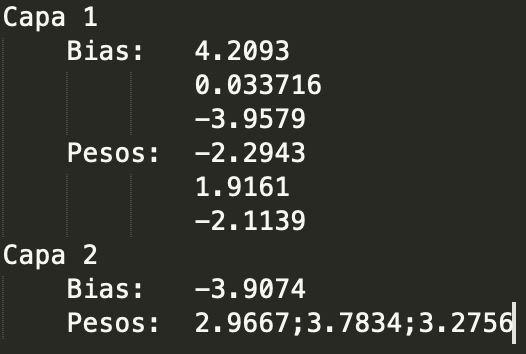


Ilustración 15 Valores pesos y bias Capas 1 y 2

**Discusión**

Siendo este el primer experimento, es la primera etapa de la practica, sin embargo ya es el producto final de codigo, como se puede ver este experiemtno demuestra la capacidad que tiene un perceptron de dos capas con 3 neuronas como se ha visto en la practica, por eso se han optenido reultados que son aceptables al comprarar la salida de la red con los valores deseados.

**Conclusion**

Aun no se puede lleagr a una conclusion concreta ya que solo se ha realziado la primera parte de la practica con datos poco grandes sin embargo ha que decir que si se ha llevado su tiempo el realizar esta practica hasta el momento. Por ende hay que recalcar que en los proximos experiemntos se llevaan las cocnlsuiones finales.

**Referencias**

M. T. Hagan, Neural Networks Design, Segunda ed.,

Diego Calvo. (2018, 8 diciembre). Perceptrón Multicapa – Red Neuronal. Recuperado 5 de junio de 2020, de https://www.diegocalvo.es/perceptron-multicapa/

El perceptron multicapa (MLP). (s. f.). Recuperado 5 de junio de 2020, de https://www.ibiblio.org/pub/linux/docs/LuCaS/Presentaciones/200304curso-glisa/redes\_neuronales/curso-glisa-redes\_neuronales-html/x105.html

LEAST MEAN SQUARE ALGORITHM. (s. f.). Recuperado 5 de junio de 2020, de http://homepages.cae.wisc.edu/~ece539/resources/tutorials/LMS.pdf