UNIVERSIDAD NACIONAL DE INGENIERIA

Practica Calificada 2

Retro propagación de errores

Lazaro Camasca Edson Nicks

Inteligencia Artificial

2019

1. Objetivo

Para desarrollar este laboratorio, tenemos por objetivo variar algunos parámetros de las redes neuronales, analizar, ver el efecto que producen y la importancia de estos:

Los parámetros a analizar son:

- Cantidad de neuronas intermedias
- Eta o radio de aprendizaje.
- Función de activación (sigmoidea gaussiana).
- Utilizar Bias (si o no)
- Tipo de entrenamiento (Batch o Patrón)
- Analizar la convergencia, si es más rápido o más lento.

Por último, concluir quienes tiene una buena aproximación, en la reducción de la función de coste del error J.

Todo el análisis se realizará sobre algunos scripts de Matlab, que son los siguientes:

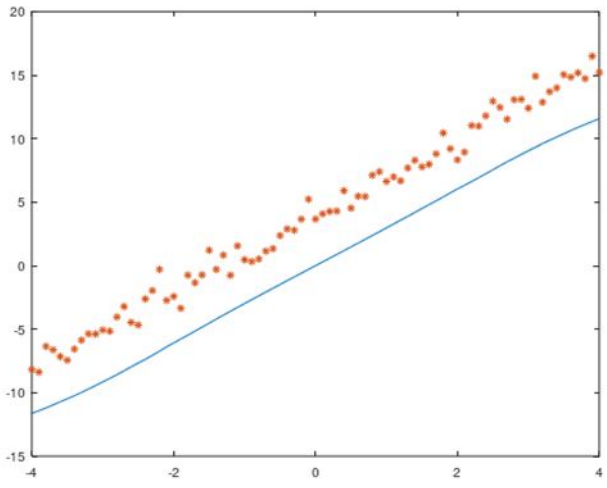
- NeuronaLinealBatch.m
- NeuronaDosEntradas.m
- NeuronaDosEntradasDosIntermedias.m

Bias

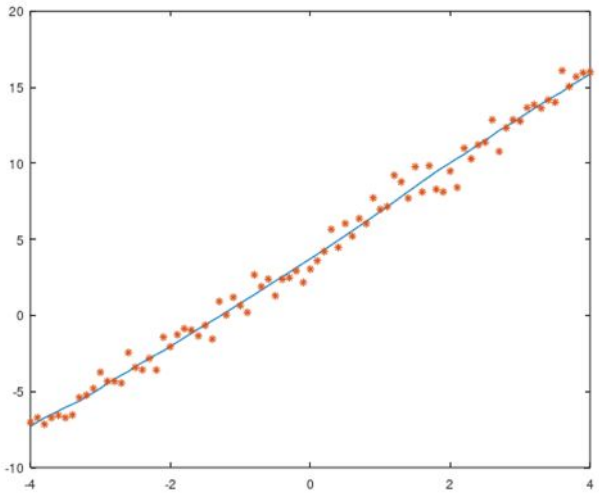
Es variable que desplaza a la función dentro de la neurona a lo largo del eje “y”, ayudando a que no exista entradas 0, cuando las entradas son 0 entonces la salida será 0, para ello se utiliza Bias para tener como entrada mínima su valor.

2. Neurona Lineal Batch

Para ver la importancia del Bias, utilizamos una eta = 0.05, una función de activación sigmoidea.

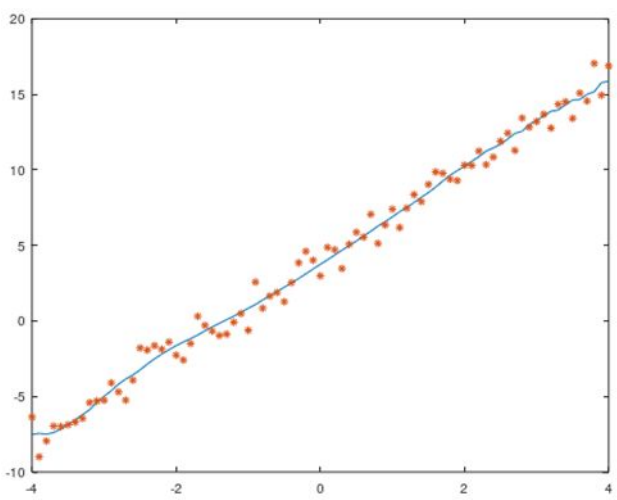


Sin Bias

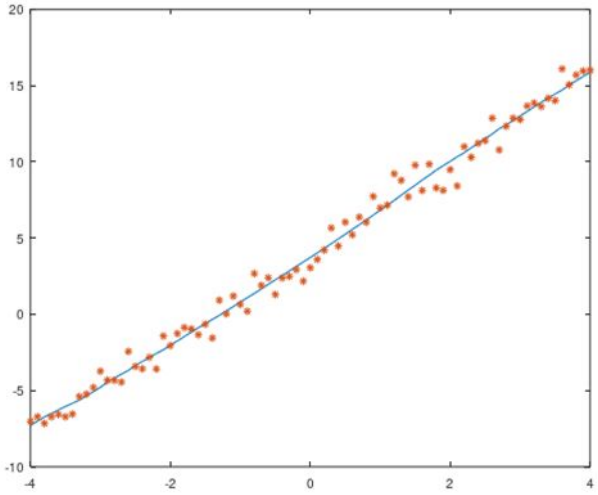


Con Bias

Se puede notar que el Bias nos ayuda muchísimo para una buena aproximación. Ahora cambiamos la función de activación por una gaussiana.



Gaussiana



Sigmoidea 2

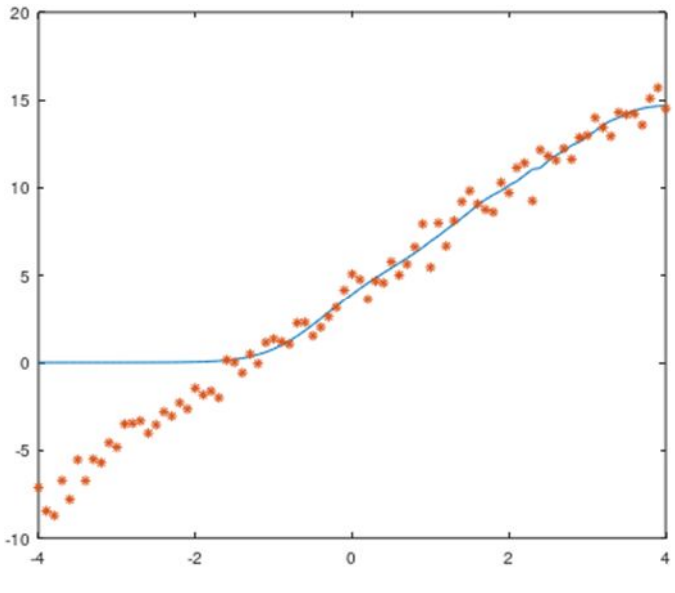
La mejor linea de tendencia la tiene una sigmoidea. Cambiamos el modo de entrenamiento de Batch a Patron.

```
%----------- Retropropagacion
er = out - yb(k,1);
error(k,1) = er;
%dndm = (1 - n.*n)/2;
dndm = -2.0*(n.*m);

% (error retro) x (salida de l
dJdw = 0*dJdw + er.*n;
dJdv = 0*dJdv + er.*in*(w.*dnd

% Actualiza las intensidades d
w = w - eta*dJdw/N;  % Patron
v = v - eta*dJdv/N;  % Patron

  end
%w = w - eta*dJdw/N;
%v = v - eta*dJdv/N;
%Batch porque la funcion de coste
JJ = 0.5*sum(error.*error)
J(iter,1) = JJ;
end
```



Entrenamiento Patrón

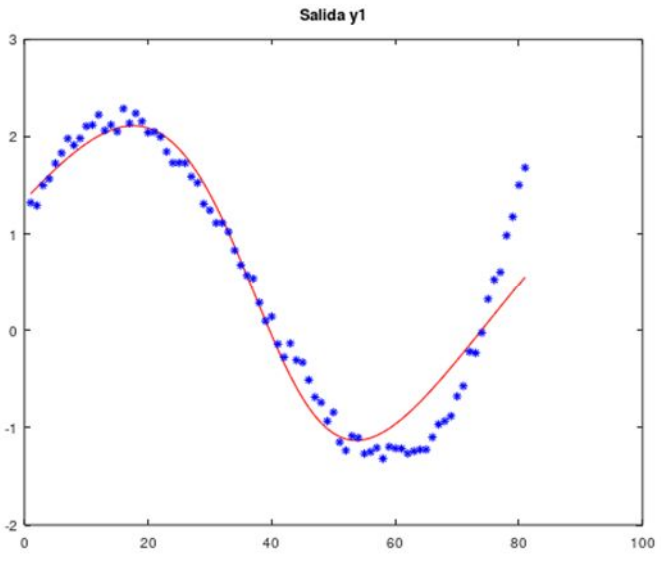
La actualizacion de la funcion de coste, lo realizamos dentro del for para que sea entrenamiento patron, podemos ver que su linea de tendencia es un distorcionada en comparacion con una sigmoidea, ademas es mucho mas lento.

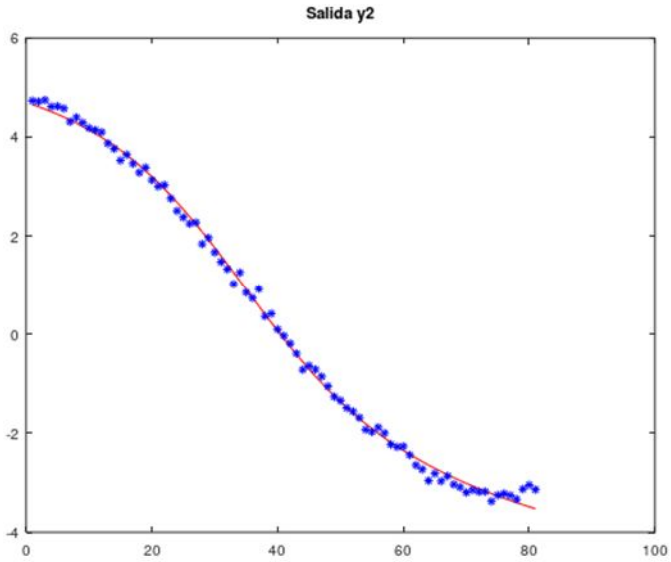
Para una neurona lineal la mejor convergencia la tiene cuando se utiliza la funcion sigmoidea, utiliza bias y un entrenamiento patron.

Nota cuando se utiliza un eta muy grande, se tiene un bucle infinito y nunca se llega a una buena aproximación, es mejor utilizar un valor pequeño entre 0.01 y 0.1.

3. Neurona Dos Entradas

Ahora se utilizarán 2 entradas y dos salidas, para la primera prueba utilizamos una eta = 0.03, con Bias, una función de activación sigmoidea 2 y un entrenamiento Batch.



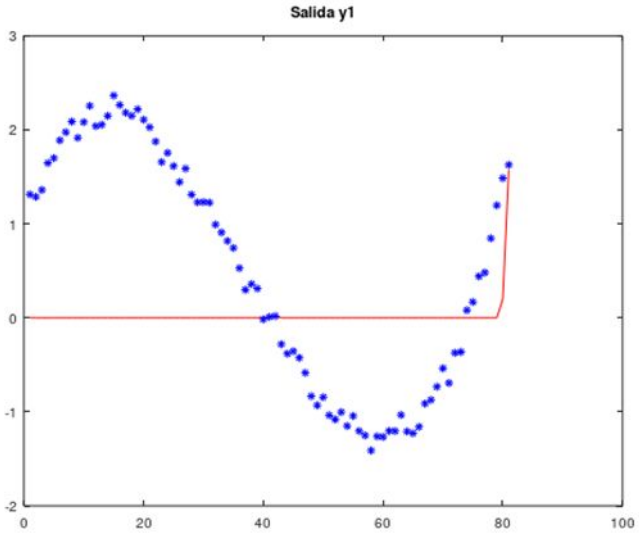


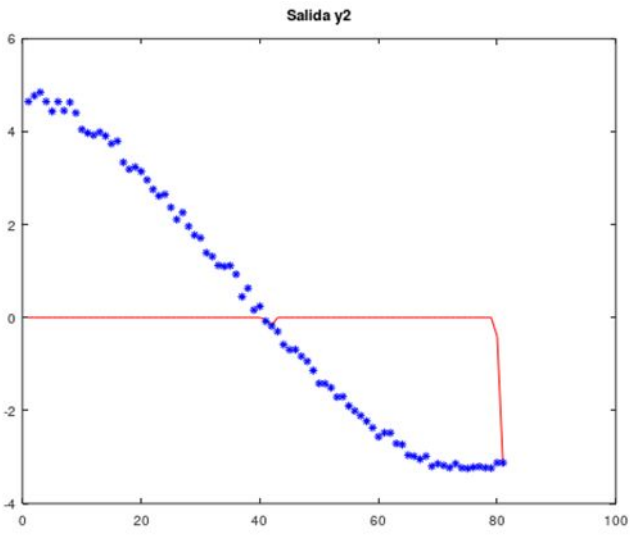
Se puede notar que la salida y2 tiene un mejor ajuste en comparacion con la salida y1.

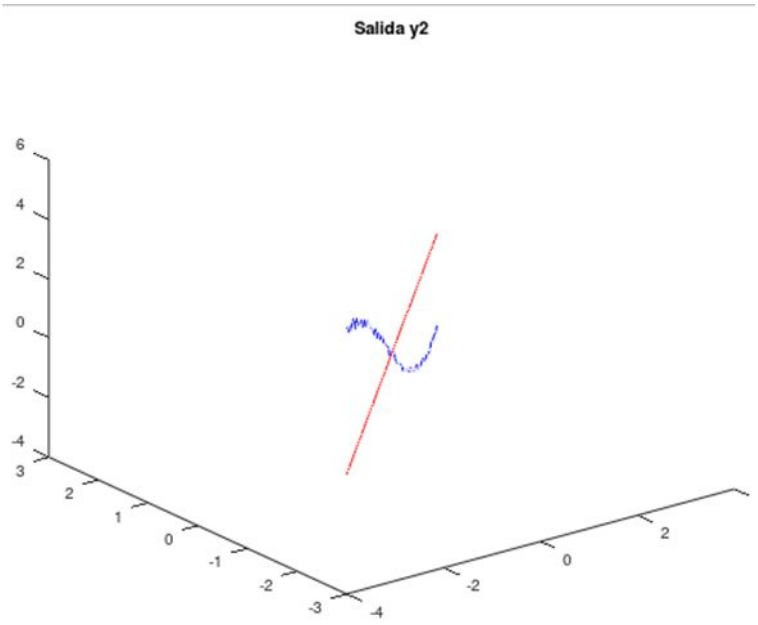
Ahora utilizemos un entrenamiento Patron y una funcion de activacion gaussiana.

Se puede notar que en ninguna de las salidas hubo una buena aproximación, siendo fatal la combinacion de un patron con una funcion gaussiana.

Cuando se aumenta el eta de 0.03 a 0.4 los resultados se empeoran mas. Se puede ver en la siguiente imagen 3D.





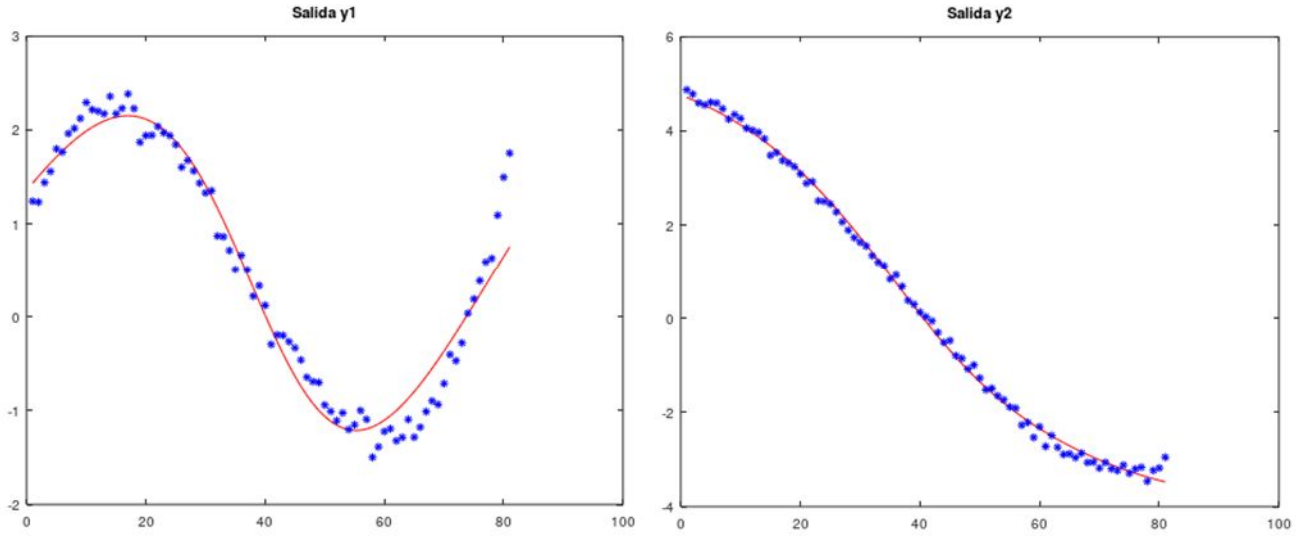


Salida y2 de una entrenamiento patron y gaussiana

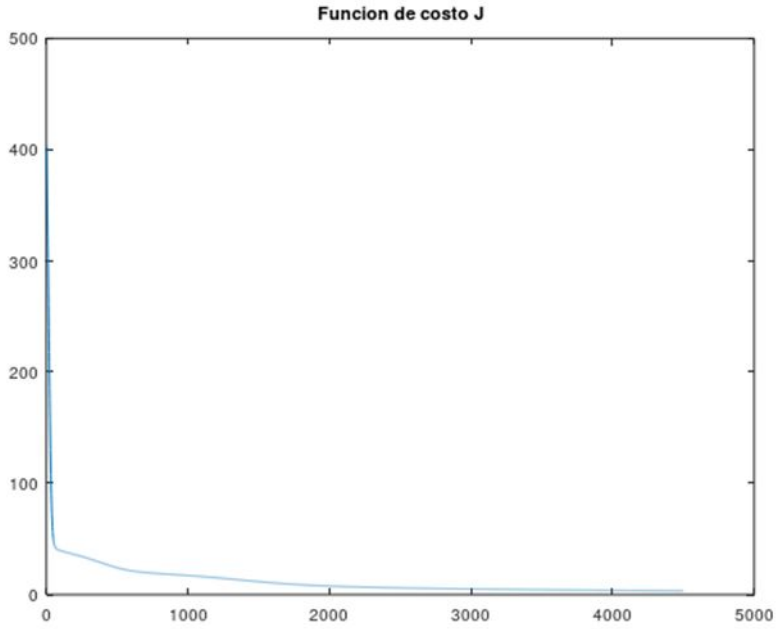
4. Neurona Dos Entradas Dos Intermedias

Ya vimos que, teniendo como función de activación a la sigmoidea2, con una eta pequeña, se encuentran buenos resultados en la aproximación. Si bien se puede simular cómo funcionan las redes neuronales con una sola neurona, en la actualidad se requiere de muchas neuronas intermedias para un aprendizaje más efectivo y profundo.

Para ello agregamos una neurona intermedia más, utilizamos una sigmoidea y un entrenamiento batch, tenemos lo siguiente.



Si bien las aproximaciones de y1 y y2 son mejores que con una sola neurona intermedia, estas convergen más lento, debido que tienen que realizar el doble de calculo que las anteriores, se puede apreciar que y2 tiene una muy buena aproximacion, En la aproximacion de la funcion de coste error se llega a una buena aproximacion a partir de las 2000 interaciones.



Si se utiliza un entrenamiento patrón las aproximaciones se disparan y se tienen valores indeterminados.

```
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
JJ =  NaN
```