INSTITUTO POLITECNICO NACIONAL

ESCUELA SUPERIOR DE COMPUTO

“PERCEPTRON”

Alumno: Chávez Chávez Javier

Maestro: Moreno Armendáriz Marco Antonio

Materia: Redes Neuronales

INTRODUCCIÓN

La primera red neuronal conocida, fue desarrollada en 1943 por Warren McCulloch y Walter Pitts; esta consistía en una suma de las señales de entrada, multiplicadas por unos valores de pesos escogidos aleatoriamente. La entrada es comparada con un patrón preestablecido para determinar la salida de la red. Si en la comparación, la suma de las entradas multiplicadas por los pesos es mayor o igual que el patrón preestablecido la salida de la red es uno (l), en caso contrario la salida es cero (0). Al inicio del desarrollo de los sistemas de inteligencia artificial, se encontró gran similitud entre su comportamiento y el de los sistemas biológicos y en principio se creyó que este modelo podía computar cualquier función aritmética o lógica. La red tipo Perceptrón fue inventada por el sicólogo Frank Rosenblatt en el año 1957. Su intención era ilustrar algunas propiedades fundamentales de los sistemas inteligentes en general, sin entrar en
mayores detalles con respecto a condiciones específicas y desconocidas para organismos biológicos concretos. Rosenblatt opinaba que la herramienta de análisis más apropiada era la teoría de probabilidades, y esto lo llevó a una teoría de separabilidad estadística que utilizaba para caracterizar las propiedades más visibles de estas redes de interconexión ligeramente aleatorias. El primer modelo de Perceptrón fue desarrollado en un ambiente biológico imitando el funcionamiento del ojo humano, el foto perceptrón como se le llamo era un dispositivo que respondía a señales ópticas; como se muestra en el figura 2.I.1 la luz incide en los puntos sensibles (S) de la estructura de la retina, cada punto S responde en forma todo-nada a la luz entrante, los impulsos generados por los puntos S se transmiten a las unidades de asociación (A) de la capa de asociación; cada unidad A está conectada a un conjunto aleatorio de puntos S, denominados conjunto fuente de la unidad A, y las conexiones pueden ser tanto excitatorias como inhibitorias. Las conexiones tienen los valores posibles +1, -l y 0, cuando aparece un conjunto de estímulos en la retina, una unidad A se activa si la suma de sus entradas sobrepasa algún valor umbral; si la unidad esta activada. A produce una salida Que se envía a la siguiente cana de unidades.

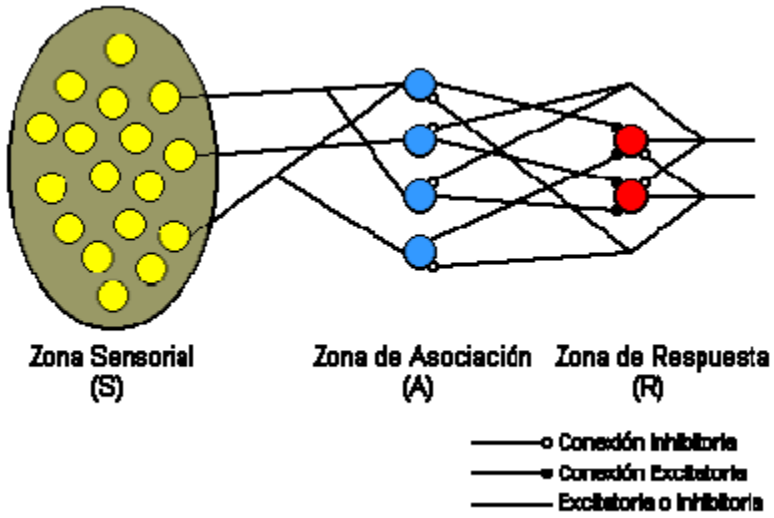


Figura 2.1.1 Modelo del Fotoperceptrón de Rosenblatt

El Perceptrón era inicialmente un dispositivo de aprendizaje, en su configuración inicial no estaba en capacidad de distinguir patrones de entrada muy complejos, sin embargo, mediante un proceso de aprendizaje era capaz de adquirir esta capacidad. En esencia, el entrenamiento implicaba un proceso de refuerzo mediante el cual la salida de las unidades A se incrementaba o se decrementaba dependiendo de si las unidades A contribuían o no a las respuestas correctas del Perceptrón para una entrada dada. Se aplicaba una entrada a la retina, y el estímulo se propagaba a través de las capas hasta que se activase una unidad de respuesta. Si se había activado la unidad de respuesta correcta, se incrementaba la salida de las unidades A que hubieran contribuido. Si se activaba una unidad R

incorrecta, se hacía disminuir la salida de las unidades A que hubiesen contribuido. Mediante estas investigaciones se pudo demostrar que el Perceptrón era capaz de clasificar patrones correctamente, en lo que Rosenblatt denominaba un entorno diferenciado, en el cual cada clase estaba formada por patrones similares. El Perceptrón también era capaz de responder de manera congruente frente a patrones aleatorios, pero su precisión iba disminuyendo a medida que aumentaba el número de patrones que intentaba aprender.

CÓDIGO

```
fprintf('RED PERCEPTRON\n');
fprintf('Caso 1: Solo para dos dimenciones (Metodo grafico) \n');
fprintf('Caso 2: Para cualquier numero de dimenciones (Regla de aprendizaje) \n');
caso=input('Ingrese el caso que desea aplicar: \n');
compuerta= input('Compuerta (AND = 1) (OR = 2)\n');
itmax=50;
switch caso
    case 1
        %Asignamos valores a los pesos
         w(1)=1/randi([-5 5]);
         w(2)=1/randi([-5 5]);
         display(w);
        %Asignamos valores a el bias
        b=1/randi([-5 5]);
        display(b);
        switch compuerta
            case 1
                %Creamos el target con la solicion
                ne=2;
                    for t=1:(2^ne)
                       if (2^ne) == t
                           vect(t)=1;
                       else
                           vect(t)=0;
                       end
                       t=t+1;
                    end
                    display(vect);
                %Creamos la matriz de puntos
                    for i=1:(2^ne)
                       entra=de2bi(i-1, ne);
                       puntos(i,1)=entra(1);
                       puntos(i,2)=entra(2);
                       i=i+1;
                    end
                    display(puntos);
                    for i=1:10
                        for j=1:(2^ne)
                            a=hardlim(w * puntos(j,:)' + b);
                            e=vect(j)-a;
                            w= w + e * puntos(j,:);
                            b=b+e;
                            display(w);
                            display(a);
                        end
                    end
                        x=puntos(:,1);
                        y=puntos(:,2);
                        linea(1)=-b/w(1);
                        linea(2)=-b/w(2);
                        plot(x,y,'*')
                        line([linea(1), 0],[0,linea(2)]);
```

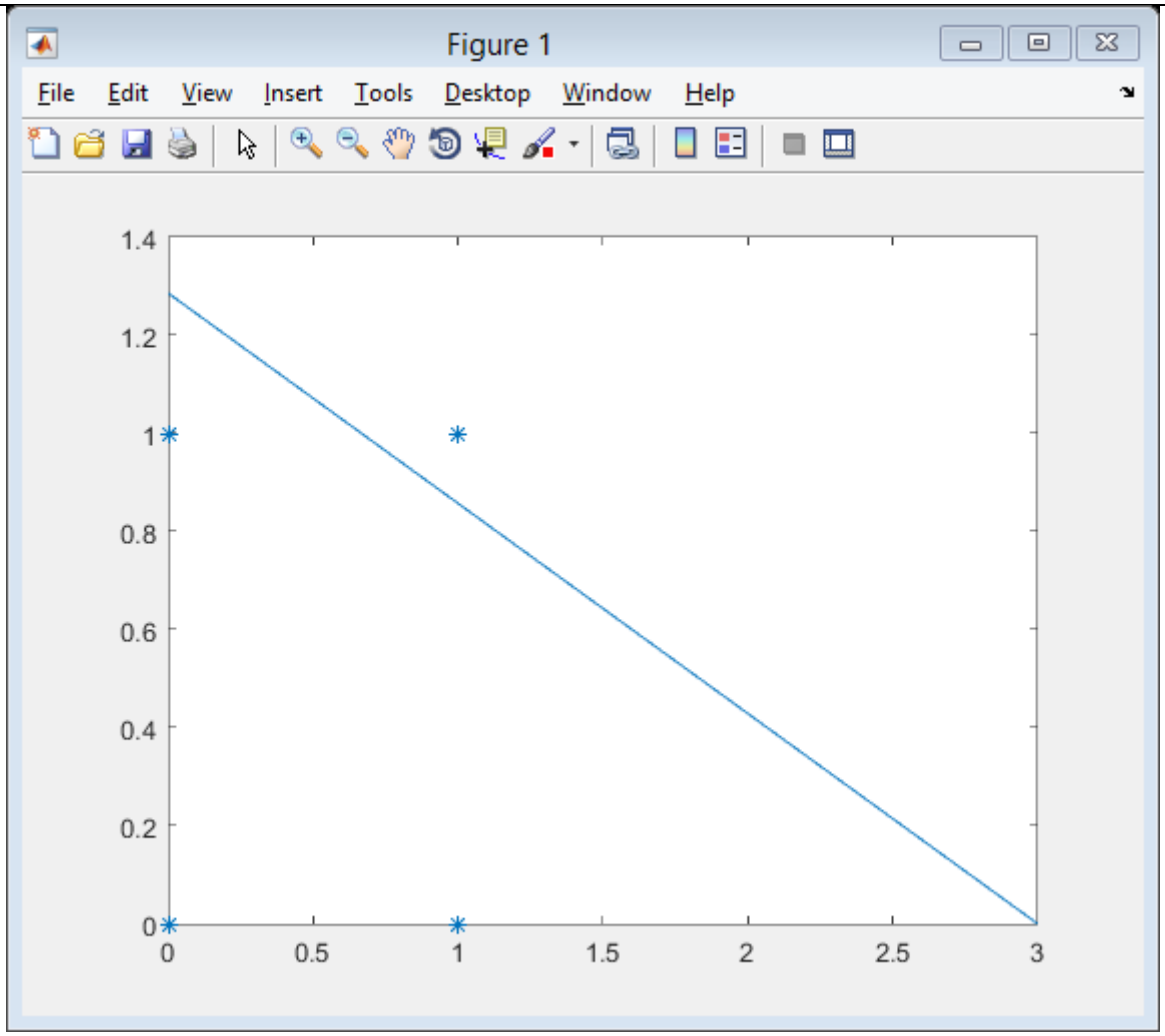
```
        case 2
            %Creamos el target con la solicion
            ne=2;
                for t=1:(2^ne)
                   if 1 == t
                       vect(t)=0;
                   else
                       vect(t)=1;
                   end
                   t=t+1;
                end
                display(vect);
            %Creamos la matriz de puntos
                for i=1:(2^ne)
                   entra=de2bi(i-1, ne);
                   puntos(i,1)=entra(1);
                   puntos(i,2)=entra(2);
                   i=i+1;
                end
                display(puntos);
                for i=1:10
                    for j=1:(2^ne)
                        a=hardlim(w * puntos(j,:)' + b);
                        e=vect(j)-a;
                        w= w + e * puntos(j,:);
                        b=b+e;
                        display(w);
                        display(a);
                    end
                end
                    x=puntos(:,1);
                    y=puntos(:,2);
                    linea(1)=-b/w(1);
                    linea(2)=-b/w(2);
                    plot(x,y,'*')
                    line([linea(1), 0],[0,linea(2)]);
        otherwise
        fprintf('La compuerta no existe');
    end
case 2
    %Genermos la matriz con n entradas
    ne=input('Ingrese el numero de entradas\n');
    %Cramos los pesos y los bias
    for k=1:ne
        w(k)=1/randi([-5 5]);
        k=k+1;
    end
    display(w);
    b=1/randi([-5 5]);
    display(b);
    fid=fopen('C:\Users\javis\Documents\MATLAB\Practica4\Pesos.txt','w');
    switch compuerta
        case 1
              %Generamos el target
              for t=1:(2^ne)
                   if (2^ne) == t
                       vect(t)=1;
                   else
                       vect(t)=0;
                   end
                   t=t+1;
              end
```

```
                for i=1:100
                    for j=1:(2^ne)
                        punto=de2bi(j-1,ne);
                        a=hardlim(w * punto' + b);
                          e=vect(j)-a;
                          w= w + e * punto;
                          fprintf(fid,'%2.2f ',w);
                          b=b+e;
                        j=j+1;
                        fprintf(fid,'\n ');
                    end

                    i=i+1;
                end
                fclose(fid);
                graf=load('Pesos.txt');%Cargamos los datos de pesos escritos en txt
                plot(graf);%graficacion de los valores de W y b
                title('Perceptron Network');
                xlabel('Convergencia')%Asignamos el titulo del eje x
                ylabel('Tranformación de w')%Asignamos el titulo del eje y
            case 2
                %Generamos el target
                for t=1:(2^ne)
                     if 1 == t
                         vect(t)=0;
                     else
                         vect(t)=1;
                     end
                     t=t+1;
                end
                for i=1:100
                    for j=1:(2^ne)
                        punto=de2bi(j-1,ne);
                        a=hardlim(w * punto' + b);
                          e=vect(j)-a;
                          w= w + e * punto;
                          fprintf(fid,'%2.2f ',w);
                          b=b+e;
                        j=j+1;
                        fprintf(fid,'\n ');
                    end

                    i=i+1;
                end
                fclose(fid);
                graf=load('Pesos.txt');%Cargamos los datos de pesos escritos en txt
                plot(graf);%graficacion de los valores de W y b
                title('Perceptron Network');
                xlabel('Convergencia')%Asignamos el titulo del eje x
                ylabel('Tranformación de w')%Asignamos el titulo del eje y
            otherwise
                fprintf('La compuerta no existe\n');
        end

    otherwise
        fprintf('No esxiste el caso');
end
```

PRUEBAS

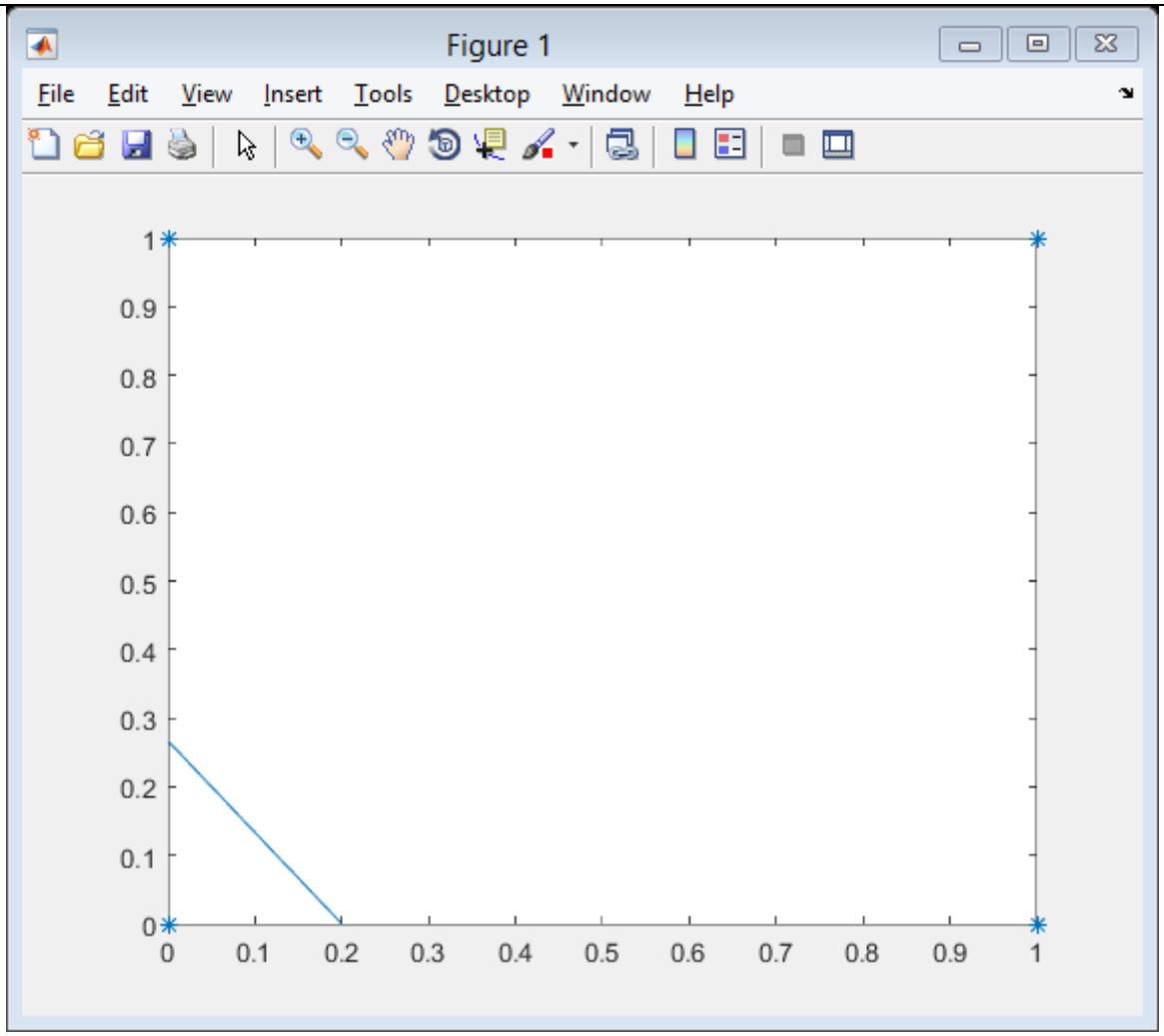
COMPUERTA AND DOS DIMENCIONES

```
RED PERCEPTRON
Caso 1: Solo para dos dimenciones (Metodo grafico)
Caso 2: Para cualquier numero de dimenciones (Regla de aprendizaje)
Ingrese el caso que desea aplicar:
1
Compuerta (AND = 1) (OR = 2)
1

w =
   1.0000    0.3333
b =
   -1
vect =
    0    0    0    1
puntos =
    0    0
    1    0
    0    1
    1    1
```



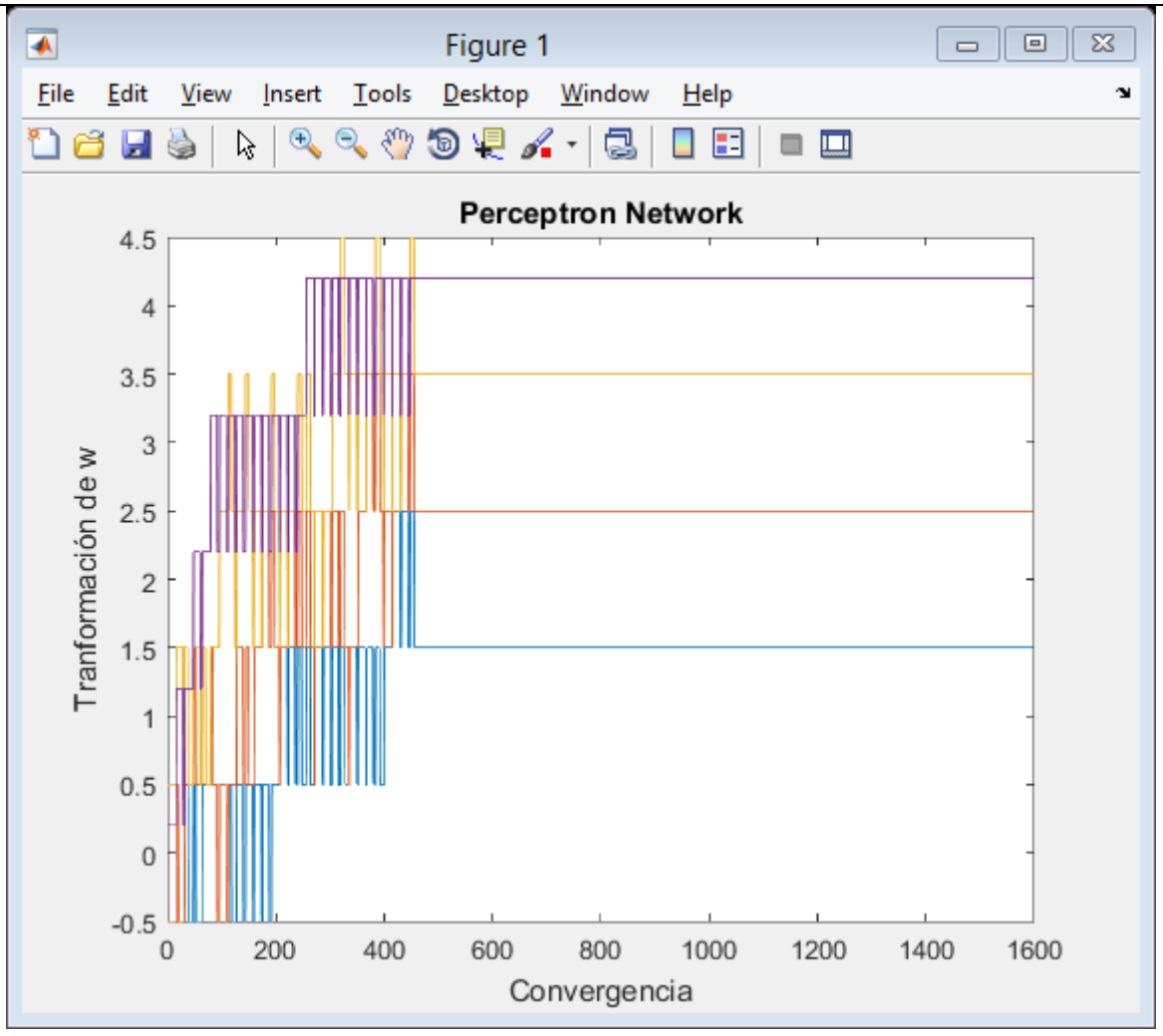
COMPUERTA OR DOS DIMENCIONES

```
Caso 1: Solo para dos dimenciones (Metodo grafico)
Caso 2: Para cualquier numero de dimenciones (Regla de aprendizaje)
Ingrese el caso que desea aplicar:
1
Compuerta (AND = 1) (OR = 2)
2
w =
   -1.0000   -0.2500
b =
   -0.2000
vect =
     0     1     1     1
puntos =
     0     0
     1     0
     0     1
     1     1
```



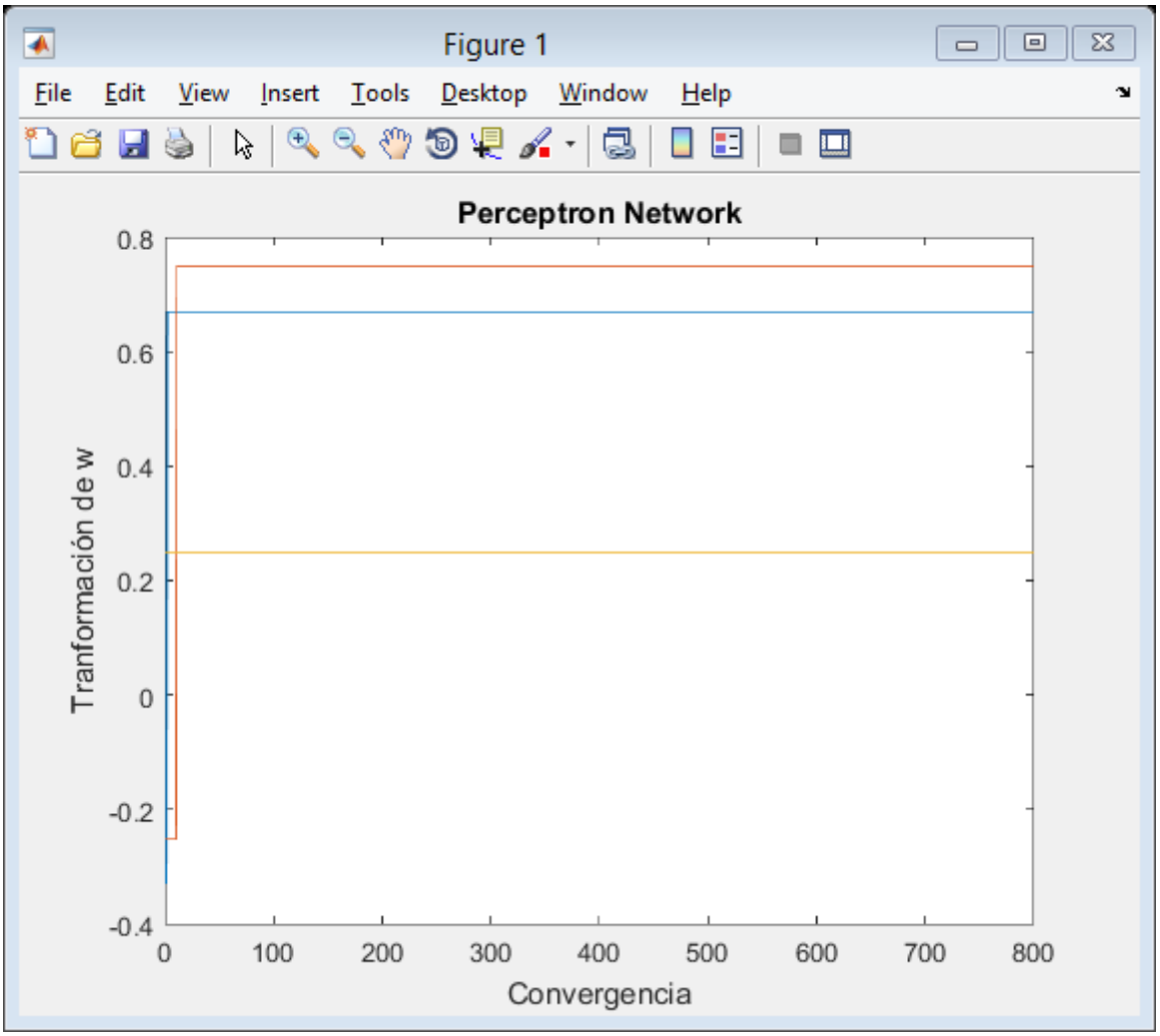
COMPUERTA AND N DIMENCIONES

```
RED PERCEPTRON
Caso 1: Solo para dos dimenciones (Metodo grafico)
Caso 2: Para cualquier numero de dimenciones (Regla de aprendizaje)
Ingrese el caso que desea aplicar:
2
Compuerta (AND = 1) (OR = 2)
1
Ingrese el numero de entradas
4
w =
   -0.5000   -0.5000    0.5000    0.2000
b =
    0.2000
```



COMPUERTA OR N DIMENCIONES

```
RED PERCEPTRON
Caso 1: Solo para dos dimenciones (Metodo grafico)
Caso 2: Para cualquier numero de dimenciones (Regla de aprendizaje)
Ingrese el caso que desea aplicar:
2
Compuerta (AND = 1) (OR = 2)
2
Ingrese el numero de entradas
3
w =
   -0.3333   -0.2500    0.2500
b =
   -0.2500
```



REFERENCIAS

(2018). *Disi.unal.edu.co*. Retrieved 29 April 2018, from http://disi.unal.edu.co/~lctorress/RedNeu/LiR