UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE CIENCIAS

INTELIGENCIA ARTIFICIAL

Informe 1 :
REDES NEURONALES

Alumno:
Julca Carhuaz Danny Michael
20111320H

Profesor:
Moran Cárdenas Antonio

Índice

1. Redes Neuronales

En el presente informe se analiza los algoritmos de aprendizaje Batch y Patron. La red neuronal presenta una sola capa oculta.
Las variaciones se realizan en las funciones de sigmoidea y gaussiana, variando el parametro de aprendizaje (eta) y el número de neuronas en la única capa oculta (nm).
Todas las pruebas con con bias.

1.1. Entrenamiento Batch

```
%Entrenamiento Batch con bias
clear;
clc;
close all;
a = 3;
b = 4;
x = -4:0.1:4;
x = x';
N = length(x);
yb = a*x + b;
yb = yb + 0.75*randn(N,1);
ne = 1;
nm = 5;
bias = input('Bias:  SI = 1 : ');
if(bias == 1)
ne = ne +1;
x = [ x ones(N,1) ];
end
v = 0.15*randn(ne,nm);
w = 0.15*randn(nm,1);
eta = input('eta : ')
for iter = 1:2000
        dJdv = 0;
        dJdw = 0;
        for k = 1:N
                in = (x(k,:))';
                m = v'*in;
                n = 2.0./(1+exp(-m)) - 1;        %Sigmoidea 2
                %n = exp(-m.^2);                    %Gaussiana
                out = w'*n;
                y(k,1) = out;
                er = out - yb(k,1);
                error(k,1) = er;
                dndm = (1 - n.*n)/2;
                %dndm = -2.0*(n.*m);
```

```
                dJdw = dJdw + er.*n;
                dJdv = dJdv + er.*in*(w.*dndm)';
                %w = w - eta*dJdw/nx;
                %v = v - eta*dJdv/nx;
        end
        w = w - eta*dJdw/N;
        v = v - eta*dJdv/N;
        JJ = 0.5*sum(error.*error)
        J(iter,1) = JJ;
end
figure(1);
plot(x(:,1),y,x(:,1),yb,'*');
figure(2);
plot(J);
```

1.1.1. Función Sigmoidea 2

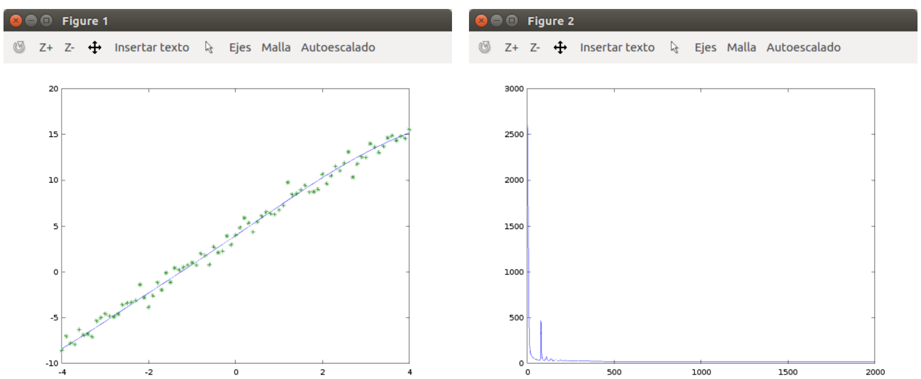

Figura 1: Usando nm 5, eta 0.1

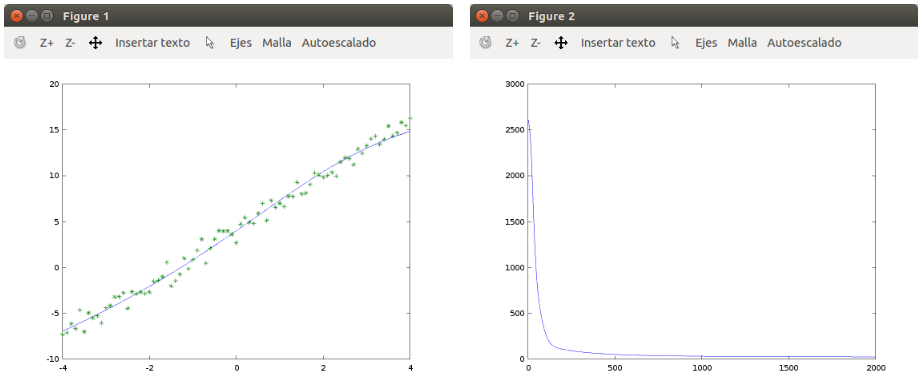

Figura 2: Usando nm 5, eta 0.01

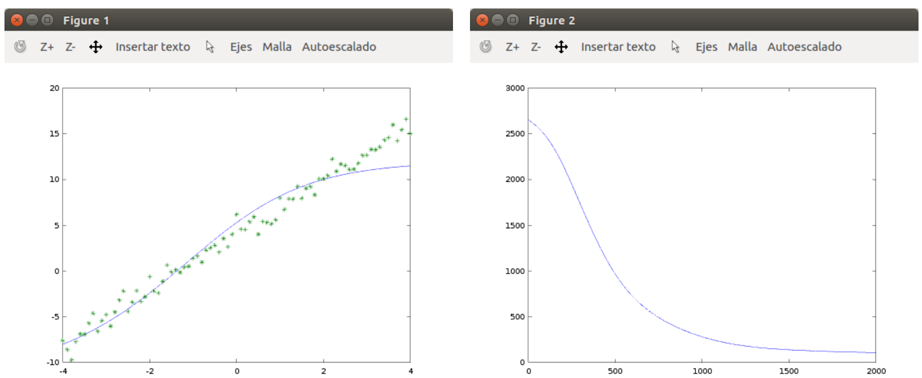

Figura 3: Usando nm 5, eta 0.001

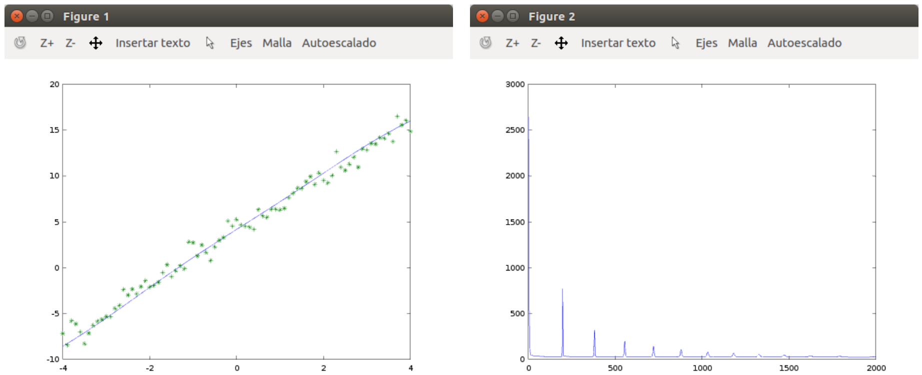

Figura 4: Usando nm 25, eta 0.1

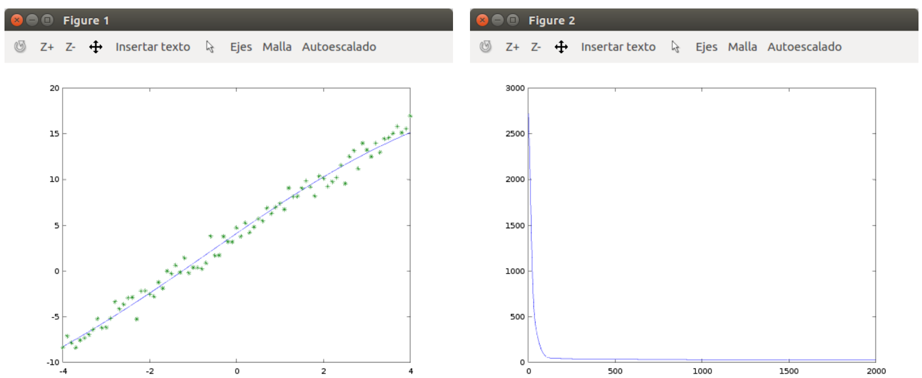

Figura 5: Usando nm 25, eta 0.01

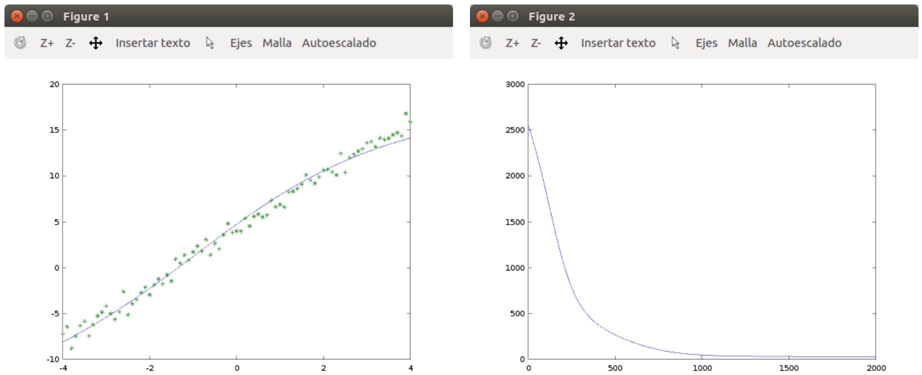

Figura 6: Usando nm 25, eta 0.001

1.1.2. Función Gaussiana

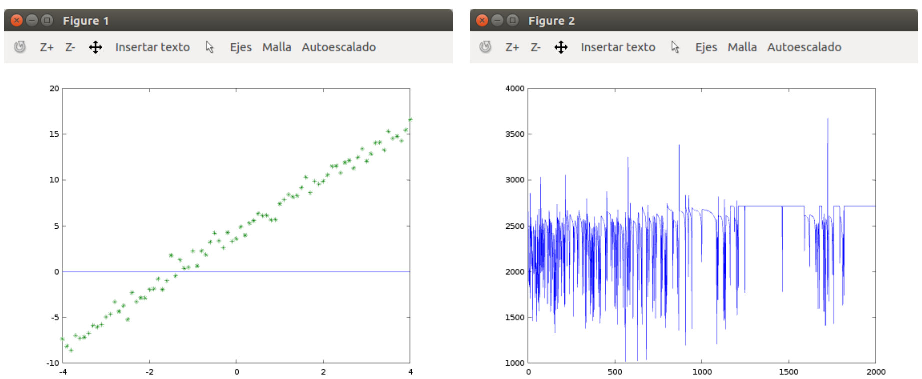

Figura 7: Usando nm 5, eta 0.1

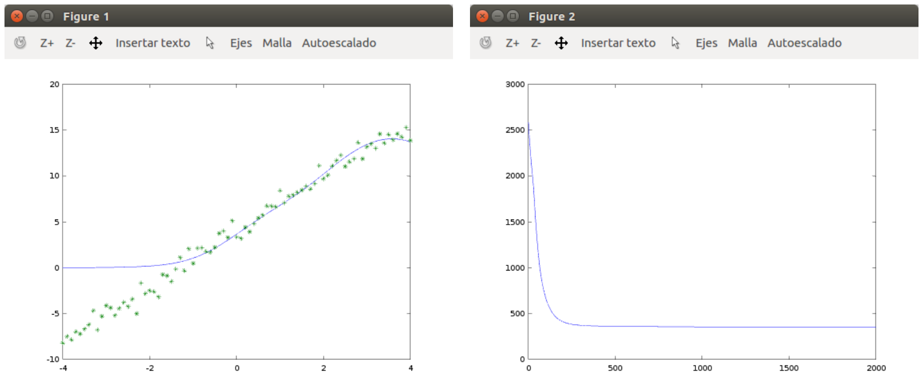

Figura 8: Usando nm 5, eta 0.01

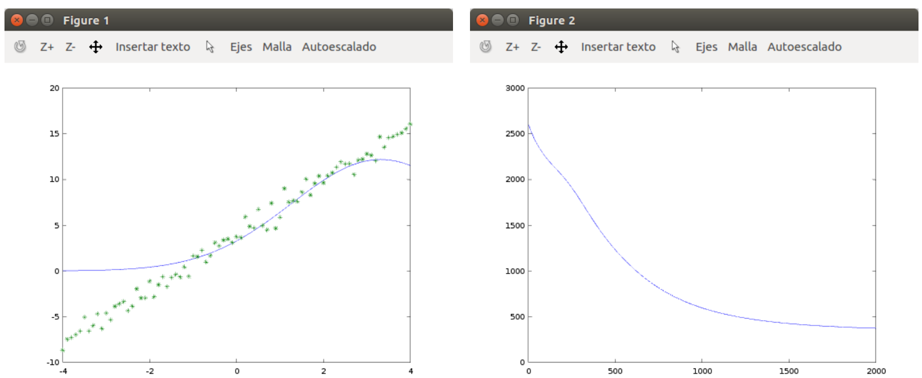

Figura 9: Usando nm 5, eta 0.001

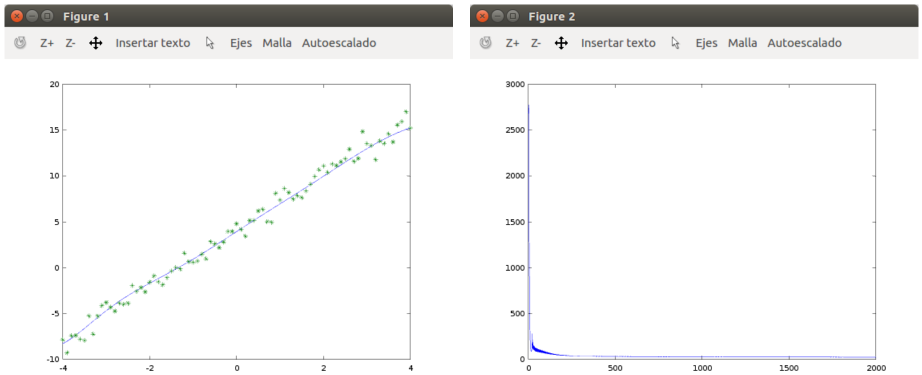

Figura 10: Usando nm 25, eta 0.1

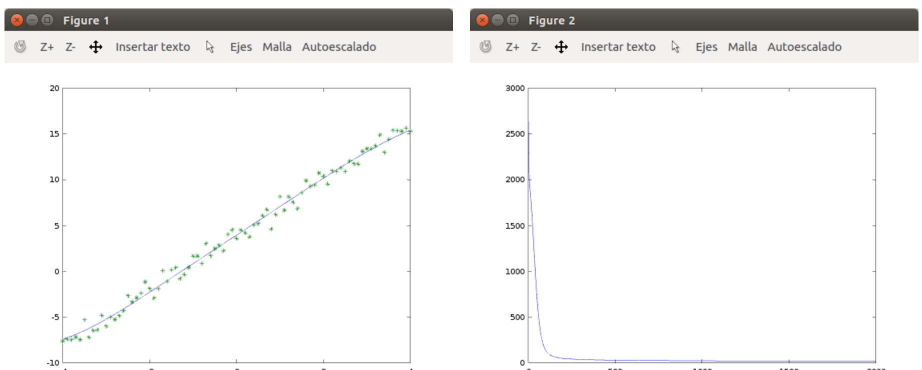

Figura 11: Usando nm 25, eta 0.01

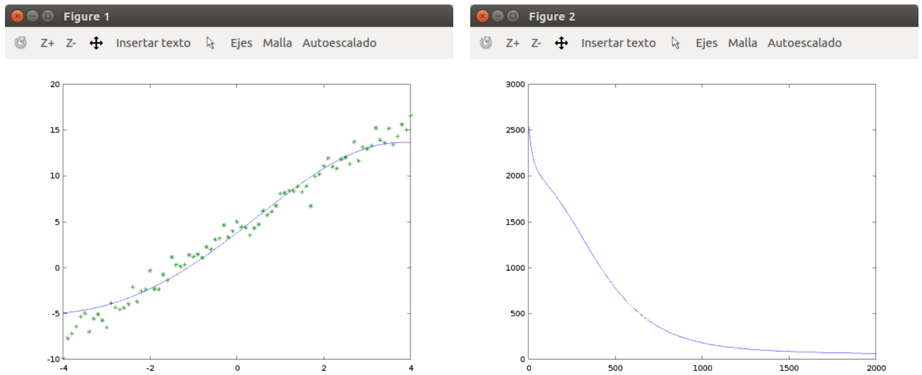

Figura 12: Usando nm 25, eta 0.001

1.2. Entrenamiento Patron

```
% Entrenamiento Patron con bias

clear;
clc;
close all;
disp('Hello');

a = 1;
b = 2;

x = -2:0.05:3;
x = x';
N = length(x);
yb = a*x + b + 0.2*randn(N,1);

ne = 1;
nm = 5;
bias = input('Bias:  SI = 1 : ');
if(bias == 1)
ne = ne +1;
x = [ x ones(N,1) ];
end
v  = 0.25*randn(ne,nm);
w = 0.25*randn(nm,1);

Jold = 1e15;
eta = input('eta : ');
for iter = 1:5000
```

```
        w11(iter,1) = w(1,1);
        dJdv = 0;      dJdw = 0;
        for k = 1:N
                in = (x(k,:))';
                m = v'*in;
                %n = 1.0./(1+exp(-m));          % Sigmoidea 1
                n = 2.0./(1+exp(-m)) - 1;   % Sigmoidea 2
                %n = exp(-m.^2);                % Gaussiana
                out = w'*n;
                y(k,1) = out;
                er = out - yb(k,1);
                error(k,1) = er;
                %dndm = n.*(1-n);       % Sigmoidea 1
                dndm = (1 - n.*n)/2;     % Sigmoidea 2
                %dndm = -2.0*(n.*m);        % Gaussiana
                dJdw = 0*dJdw + er.*n;
                dJdv  = 0*dJdv +  er.*in*(w.*dndm)';
                w = w - eta*dJdw;
                v = v - eta*dJdv;
        end
        %w = w - eta*dJdw/N;
        %v = v - eta*dJdv/N;
        JJ = 0.5*sum(error.*error)
        dJ = abs(JJ - Jold);
        dJpor = sqrt(dJ/JJ)*100;
        if(dJpor < 0.75)      % Porcentual
                break;
        end
        J(iter,1) = JJ;
        Jold = JJ;
end

figure(1);
plot(x(:,1),y,x(:,1),yb,'*');
figure(2);
subplot(2,1,1);   plot(J);
subplot(2,1,2);   plot(w11);
```

1.2.1. Función Sigmoidea 2

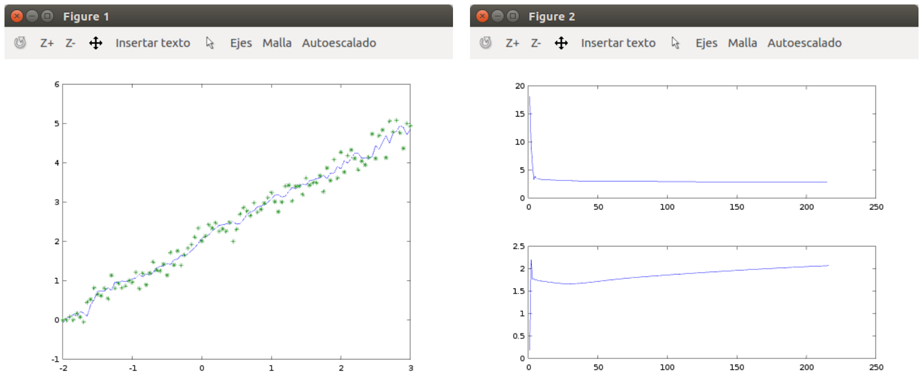

Figura 13: Usando nm 5, eta 0.1

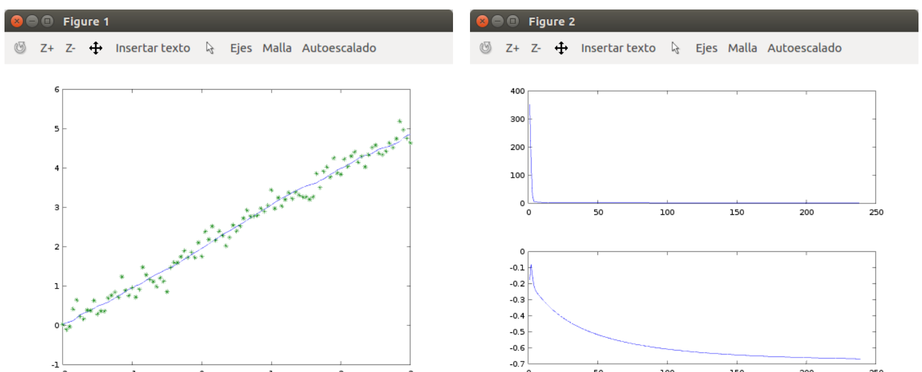

Figura 14: Usando nm 5, eta 0.01

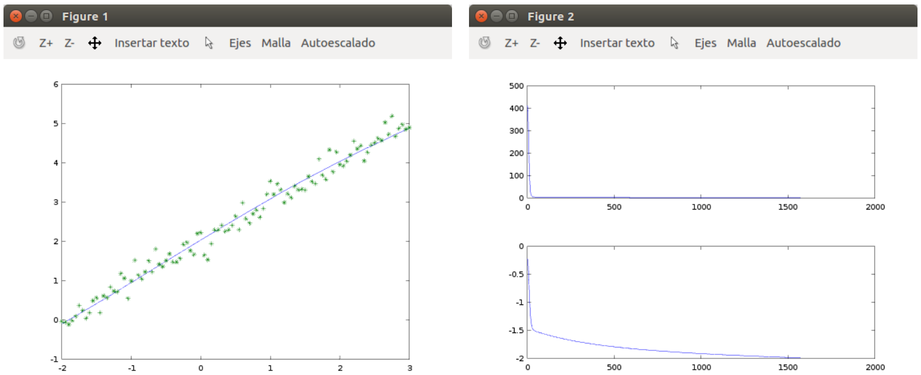

Figura 15: Usando nm 5, eta 0.001

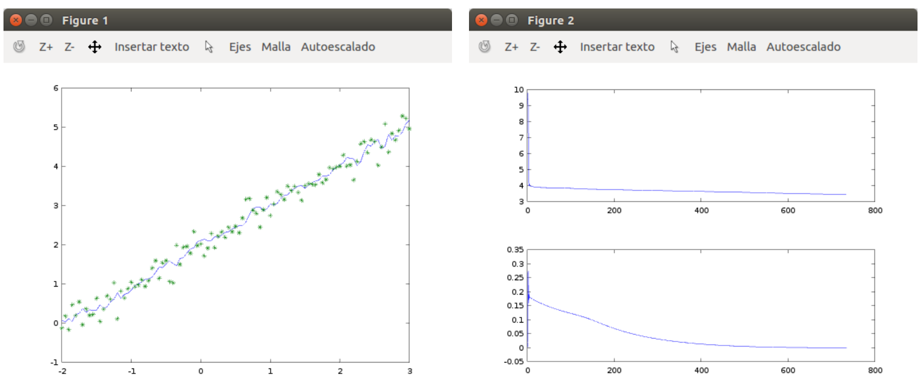

Figura 16: Usando nm 25, eta 0.1

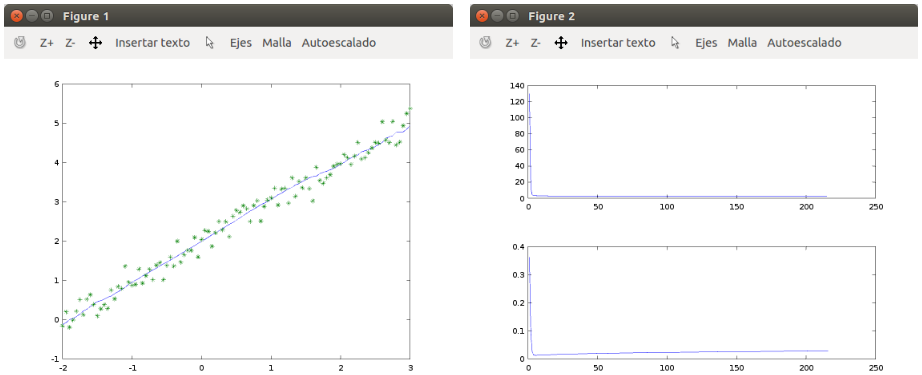

Figura 17: Usando nm 25, eta 0.01

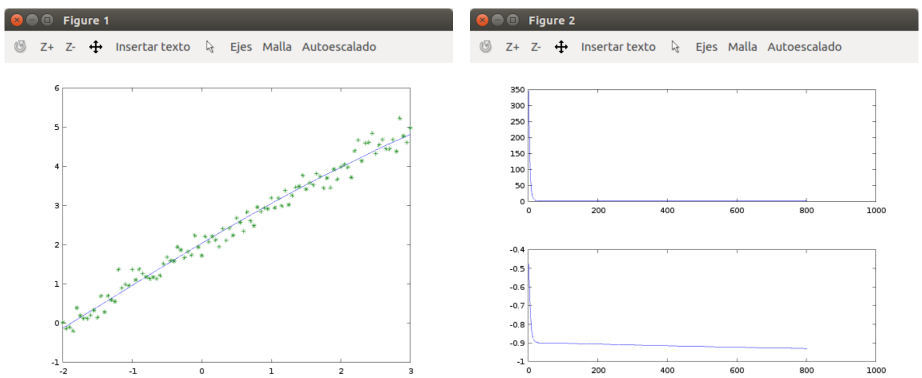

Figura 18: Usando nm 25, eta 0.001

1.2.2. Función Gaussiana

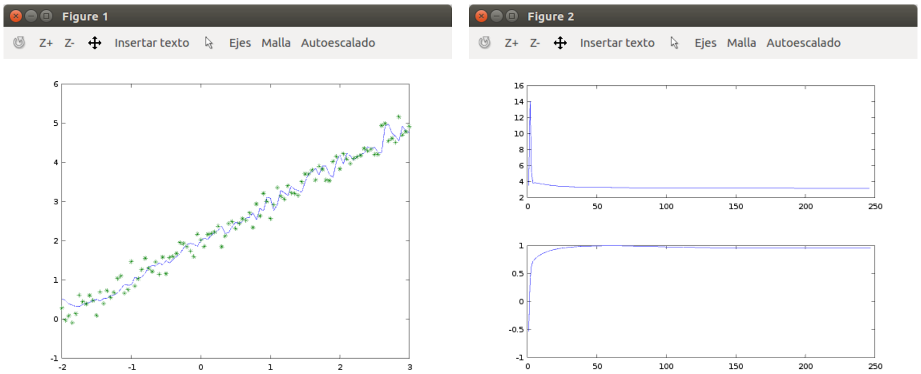

Figura 19: Usando nm 5, eta 0.1

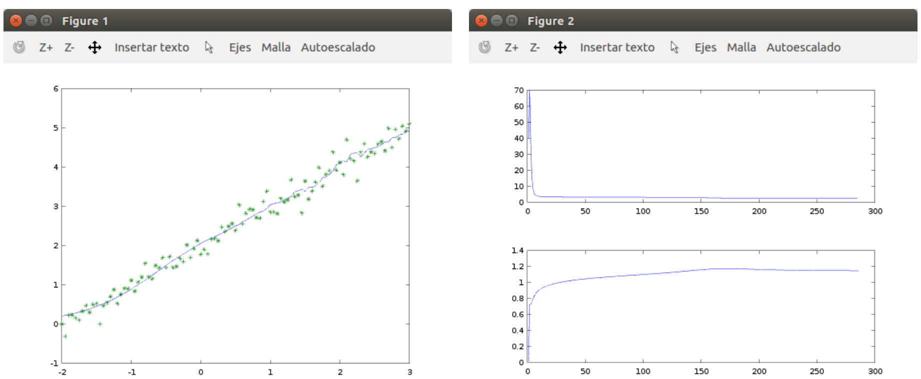

Figura 20: Usando nm 5, eta 0.01

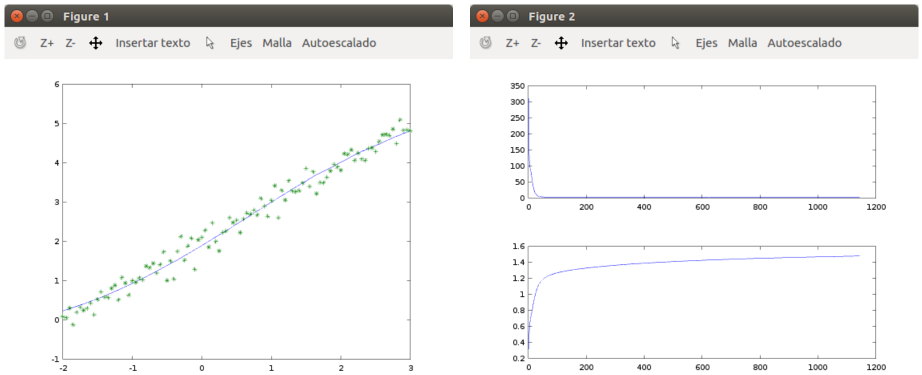

Figura 21: Usando nm 5, eta 0.001

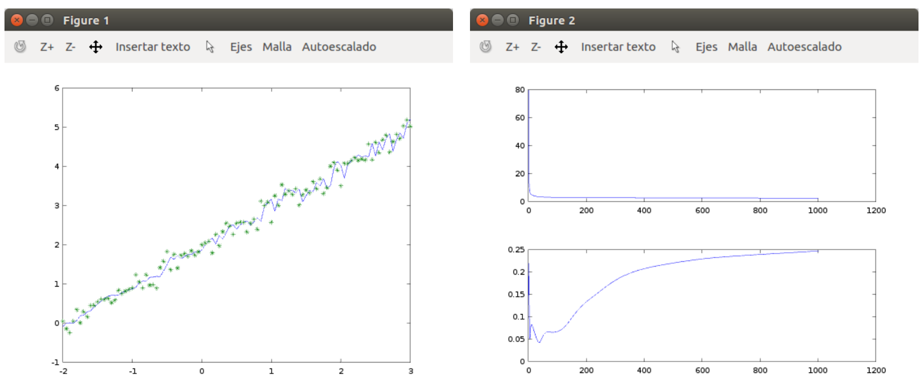

Figura 22: Usando nm 25, eta 0.1

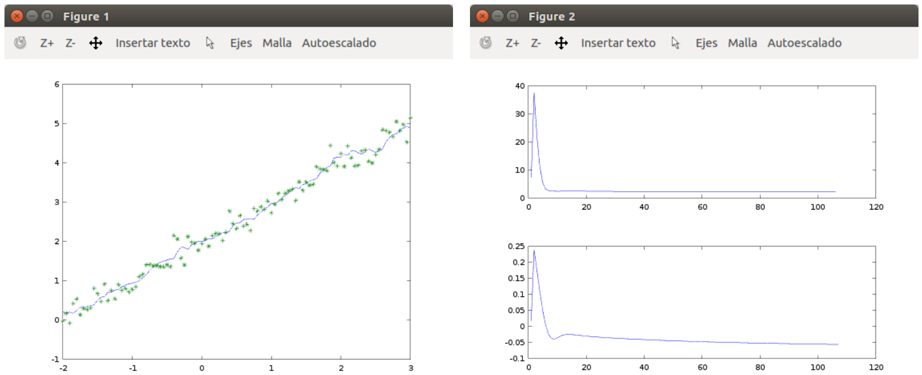

Figura 23: Usando nm 25, eta 0.01

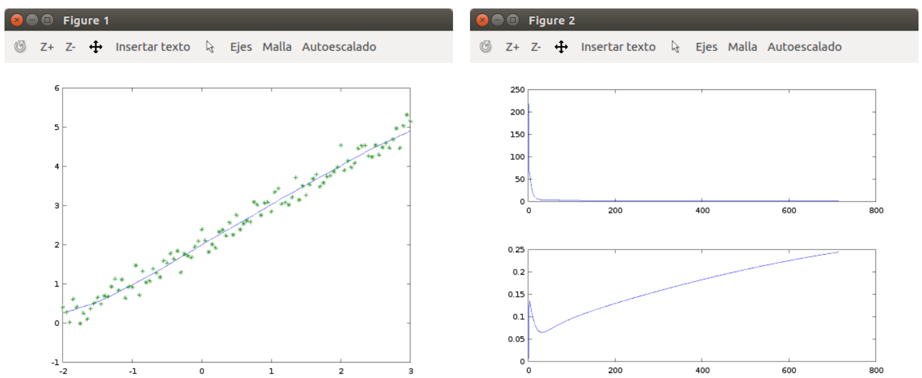

Figura 24: Usando nm 25, eta 0.001