

# Práctica 1. Adaline.

Sistemas Conexionistas - Curso 09/10

---

## 1. Entrenamiento del Adaline

### 1.1. Implementación

Implementar la **función** `trainAdaline` en MATLAB que simule el entrenamiento batch de un Adaline según la regla de Widrow-Hoff o LMS.

La función recibirá como parámetros:

- Los patrones de entrada. El programa debe aceptar  $n$  patrones de  $m$  componentes cada uno.
- Las salidas deseadas. Habrá una salida por patrón de entrada ( $n$  salidas) de 1 componente.
- La velocidad de aprendizaje
- El error máximo permitido
- Un parámetro que indique si el entrenamiento se realizará con bias o no

La función implementará el entrenamiento del adaline con y sin bias siguiendo la regla LMS. En cada iteración de entrenamiento, los pesos se recalcularán tras introducir todos los patrones de entrada. El entrenamiento finalizará cuando se alcance el error máximo permitido o se realicen un número elevado de iteraciones.

La función devolverá como salida:

- Los pesos finales tras el entrenamiento
- El bias final tras el entrenamiento (en entrenamientos con bias; devolver 0 en caso de entrenamiento sin bias)
- La evolución de los pesos a lo largo de las iteraciones
- La evolución del error a lo largo de las iteraciones
- La evolución del bias a lo largo de las iteraciones (en entrenamientos con bias; devolver un vector vacío en caso de entrenamiento sin bias)

## 1.2. Conjuntos de entrenamiento

Utilizar la función `trainAdaline` en los siguientes conjuntos de entrenamiento. Probar distintas velocidades de aprendizaje y considerar un error de 0.0001.

### Conjunto A

Comparar los resultados del entrenamiento con bias y sin bias con el siguiente conjunto de patrones y salidas deseadas:

Patrones de entrada	Salida deseada
5,1,3	6
4,3,-1	4
1,-2,4	5

Representar gráficamente:

- La evolución del error con las iteraciones
- La evolución de los pesos y del bias con las iteraciones en una misma gráfica

### Conjunto B

Utilizar un entrenamiento sin bias con el siguiente conjunto de patrones y salidas deseadas:

Patrones de entrada	Salida deseada
2,-1	3
-2,3	-1
1,1	3

Representar gráficamente:

- La evolución del error con las iteraciones
- La superficie de error y la evolución del error a lo largo de las iteraciones sobre dicha superficie

## 2. Aplicación de un Adaline a un problema real

Disponemos de un dispositivo de funcionamiento desconocido. Este dispositivo recibe una señal de entrada (`s1.mat`) y devuelve una señal de salida (`s2.mat`). Utilizar un adaline para simular el funcionamiento de este dispositivo. Para ello:

### 1. SELECCIONAR UN CONJUNTO DE ENTRENAMIENTO

Tomar patrones de entrada de la señal de entrada  $\mathbf{s1}$  y salidas deseadas de la señal de salida  $\mathbf{s2}$ . Aunque podríamos usar otras alternativas, en este ejemplo vamos a considerar que un patrón de entrada son cinco muestras consecutivas de la señal  $\mathbf{s1}$  ( $\mathbf{s1}(i-2)$ ,  $\mathbf{s1}(i-1)$ ,  $\mathbf{s1}(i)$ ,  $\mathbf{s1}(i+1)$ ,  $\mathbf{s1}(i+1)$ ) y la salida deseada correspondiente es  $\mathbf{s2}(i)$ , siendo  $i$  una posición cualquiera de los vectores que almacenan las señales.

El conjunto de entrenamiento debe ser representativo. El número de muestras debe ser suficiente para el entrenamiento aunque no excesivo por cuestiones de eficiencia computacional.

### 2. ENTRENAMIENTO

Utilizar la función `trainAdaline` sin bias con el conjunto de entrenamiento seleccionado.

### 3. SIMULAR EL COMPORTAMIENTO DEL DISPOSITIVO

Los pesos obtenidos tras el entrenamiento del adaline nos servirán para simular el comportamiento del dispositivo. El dispositivo recibe la señal de entrada  $\mathbf{s1}$  y devuelve la señal  $\mathbf{s2}$ . Implementar una función que tome como entrada la señal  $\mathbf{s1}$  así como los pesos obtenidos tras el entrenamiento y devuelva una señal de salida, teniendo en cuenta que la salida del adaline se calcula multiplicando el vector de pesos por cada patrón de entrada y cada patrón de entrada está formado por 5 componentes consecutivas.

Representar en una gráfica la señal simulada por el adaline y la señal de salida original  $\mathbf{s2}$  (tened en cuenta el instante temporal en que empiezan ambas señales).