

Diferenciación automática (algorítmica) - AD

Qué es AD? : código, software que transforma el código de una función en código de la derivada de la función.

$$y = f(x)$$

derivada simbólica
(humano/computador)

$$y' = f'(x)$$

↓
Programador
 $f(x) \{ \dots \}$
código

derivada automática

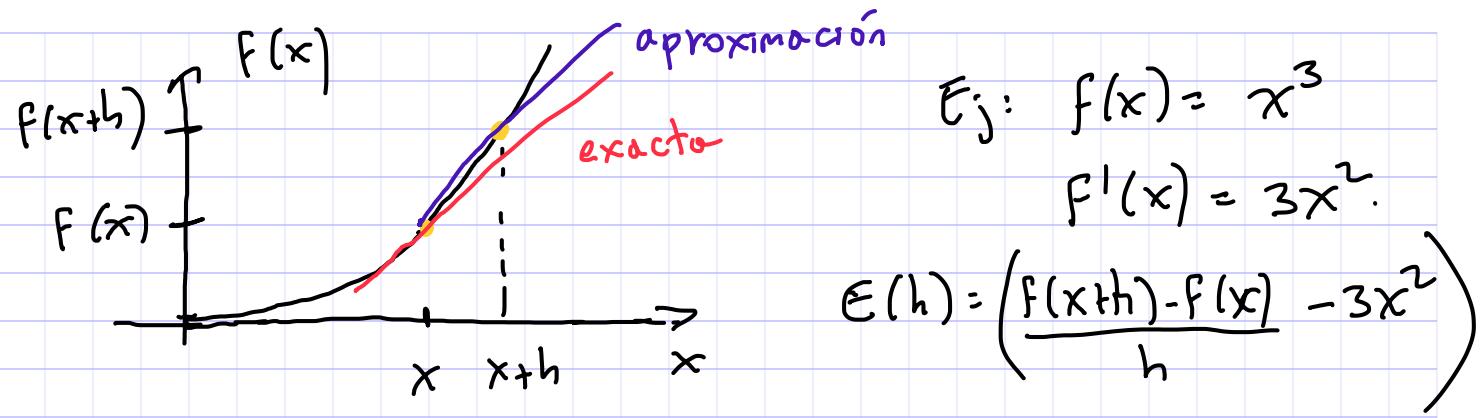
↓
 $\delta f(x) \{ \dots \}$
código

Alternativas para encontrar $f'(x)$:

- Derivada simbólica
- Derivada por diferencias
- Derivada automática AD.

Por definición: $f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$

Aproximación: $f'(x) \approx \frac{f(x+h) - f(x)}{h}$



→ Se sugieren precisiones alrededor de 10^{-12}

→ Para funciones multivariadas se requieren múltiples evaluaciones de f .

$$Ej: f(x_1, x_2) = 3x_1^2 + 5x_2^2$$

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = 6x_1; \quad \frac{\partial f(x_1, x_2)}{\partial x_2} = 10x_2.$$

$$\frac{\partial f(x_1, x_2)}{\partial x_1} \approx \frac{f(x_1+h, x_2) - f(x_1, x_2)}{h}$$

$$\frac{\partial f(x_1, x_2)}{\partial x_2} \approx \frac{f(x_1, x_2+h) - f(x_1, x_2)}{h}.$$

Fundamentos prácticos de AD: Números duales.

$$x \mapsto x + \dot{x}\delta.$$

- δ : símbolo que distingue el segundo componente.
- Análogo al componente imaginario $i = \sqrt{-1}$
- Restricción: $\delta^2 \rightarrow 0$; ($i^2 = -1$).

Aritmética básica sobre números duales:

$$\rightarrow (x + \dot{x}\delta) + (y + \dot{y}\delta) = x + y + (\dot{x} + \dot{y})\delta.$$

$$\begin{aligned} \rightarrow (x + \dot{x}\delta)(y + \dot{y}\delta) &= xy + x\dot{y}\delta + \dot{x}y\delta + \dot{x}\dot{y}\delta^2 \\ &= xy + (x\dot{y} + \dot{x}y)\delta. \end{aligned}$$

$$\rightarrow -(x + \dot{x}\delta) = -x - \dot{x}\delta.$$

$$\begin{aligned} \rightarrow \frac{x + \dot{x}\delta}{y + \dot{y}\delta} &= \frac{x + \dot{x}\delta}{y + \dot{y}\delta} \frac{(y - \dot{y}\delta)}{(y - \dot{y}\delta)} = \frac{xy - x\dot{y}\delta + \dot{x}y\delta - \dot{x}\dot{y}\delta^2}{y^2 - \dot{y}^2\delta^2} \\ &= \frac{xy + (\dot{x}y - x\dot{y})\delta}{y^2} = \frac{x}{y} + \frac{(\dot{x}y - x\dot{y})\delta}{y^2}. \end{aligned}$$

Polinomios sobre números duales

Sea el polinomio: $P(x) = p_0 + p_1x + p_2x^2 + \dots + p_nx^n$

$$\begin{aligned} P(x + \dot{x}\delta) &= p_0 + p_1(x + \dot{x}\delta) + p_2(x + \dot{x}\delta)^2 + \dots + p_n(x + \dot{x}\delta)^n \\ &= p_0 + p_1x + p_1\dot{x}\delta + p_2(x^2 + 2x\dot{x}\delta + \dot{x}^2\delta^2) + \dots \\ &\quad \dots + p_n \sum_{k=0}^n \frac{n!}{k!(n-k)!} x^{n-k} (\dot{x}\delta)^k \end{aligned}$$

Teorema
del binomio.

$$P_n \sum_{k=0}^n \frac{n!}{k!(n-k)!} x^{n-k} (\dot{x}\delta)^k = P_n \left[\frac{n!}{0!n!} x^n + \frac{n!}{(n-1)!} x^{n-1} \dot{x}\delta + \dots + \frac{n!}{2!(n-2)!} x^{n-2} \dot{x}^2 \delta^2 + \dots \right]$$

$\rightarrow \Phi$

$$= P_n x^n + P_n n x^{n-1} \dot{x}\delta.$$

$$\begin{aligned} P(x+\dot{x}\delta) &= P_0 + P_1 x + P_2 x^2 + \dots + P_n x^n + \dots \\ &\quad \dots + P_1 \dot{x}\delta + 2P_2 x \dot{x}\delta + \dots + nP_n x^{n-1} \dot{x}\delta \\ &= P(x) + P'(x) \dot{x}\delta \end{aligned}$$

NOTA: \dot{x} se puede escoger por conveniencia en $\dot{x}=1$.

$$P(x+\delta) = P(x) + \underbrace{P'(x)\delta}_{\text{derivada}}$$

Otras propiedades:

$$\sin(x+\dot{x}\delta) = \sin(x) + \cos(x) \dot{x}\delta$$

$$\cos(x+\dot{x}\delta) = \cos(x) - \sin(x) \dot{x}\delta$$

$$e^{(x+\dot{x}\delta)} = e^x + e^x \dot{x}\delta$$

$$\log(x+\dot{x}\delta) = \log(x) + \frac{\dot{x}\delta}{x}; \quad x \neq 0$$

$$\sqrt{x+\dot{x}\delta} = \sqrt{x} + \frac{\dot{x}\delta}{2\sqrt{x}}; \quad x \neq 0$$

Utilidad de los números duales:

→ Una función aplicada a un número dual retornará su derivada en el componente dual.

Y para funciones multivariadas?

Ej: $f(x_1, x_2) = x_1 x_2 + \sin(x_1)$

$$f(x_1 + \dot{x}_1 d_1, x_2 + \dot{x}_2 d_2) = (x_1 + \dot{x}_1 d_1)(x_2 + \dot{x}_2 d_2) + \sin(x_1 + \dot{x}_1 d_1)$$

$$= x_1 x_2 + x_1 \dot{x}_2 d_2 + x_2 \dot{x}_1 d_1 + \underbrace{\dot{x}_1 \dot{x}_2 d_1 d_2}_{\rightarrow 0} + \sin(x_1) + \cos(x_1) \dot{x}_1 d_1$$

$$= \underbrace{x_1 x_2 + \sin(x_1)}_{f(x_1, x_2)} + \underbrace{(x_2 + \cos(x_1)) \dot{x}_1 d_1}_{\frac{\partial f(x_1, x_2)}{\partial x_1}} + \underbrace{\dot{x}_1 \dot{x}_2 d_2}_{\frac{\partial f(x_1, x_2)}{\partial x_2}}$$

Descomposición de funciones y regla de la cadena.

Código para $f(x_1, x_2) = x_1 x_2 + \sin(x_1)$

Programa original

Programa dual

$$w_1 = x_1$$

$$w_2 = x_2$$

$$w_3 = w_1 \cdot w_2$$

$$w_4 = \sin(w_1)$$

$$w_5 = w_3 + w_4$$

$$\dot{w}_1 \rightarrow \dot{x}_1 d_1$$

$$\dot{w}_2 \rightarrow \dot{x}_2 d_2$$

$$\dot{w}_3 = w_1 \dot{w}_2 + w_2 \dot{w}_1$$

$$\dot{w}_4 = \cos(w_1) \dot{w}_1$$

$$\dot{w}_5 = \dot{w}_3 + \dot{w}_4$$

Utilizando regla de la cadena se pueden propagar los componentes duales para el cálculo.

$$\text{Por ejemplo: } \frac{\partial f(x_1, x_2)}{\partial x_2} = \frac{\partial f}{\partial w_5} \frac{\partial w_5}{\partial w_3} \frac{\partial w_3}{\partial w_2} \frac{\partial w_2}{\partial x_2}$$

En ese caso $w_1 = 0$; $w_2 = 1$. para $\frac{\partial f}{\partial x_2}$.

Programa dual

$$w_1 = 0$$

$$w_2 = 1$$

$$w_3 = w_1 \cdot w_2 + w_1 \cdot w_2 = 0 \cdot w_2 + w_1 \cdot 1 = 0 \cdot x_2 + x_1 \cdot 1 = x_1.$$

$$w_4 = \cos(w_1) \quad w_1 = \cos(w_1) \cdot 0 = 0.$$

$$w_5 = w_3 + w_4 = x_1 + 0 = x_1$$

$$\text{Comprobemos: } \frac{\partial}{\partial x_2} \{ x_1 x_2 + \sin(x_1) \} = x_1$$

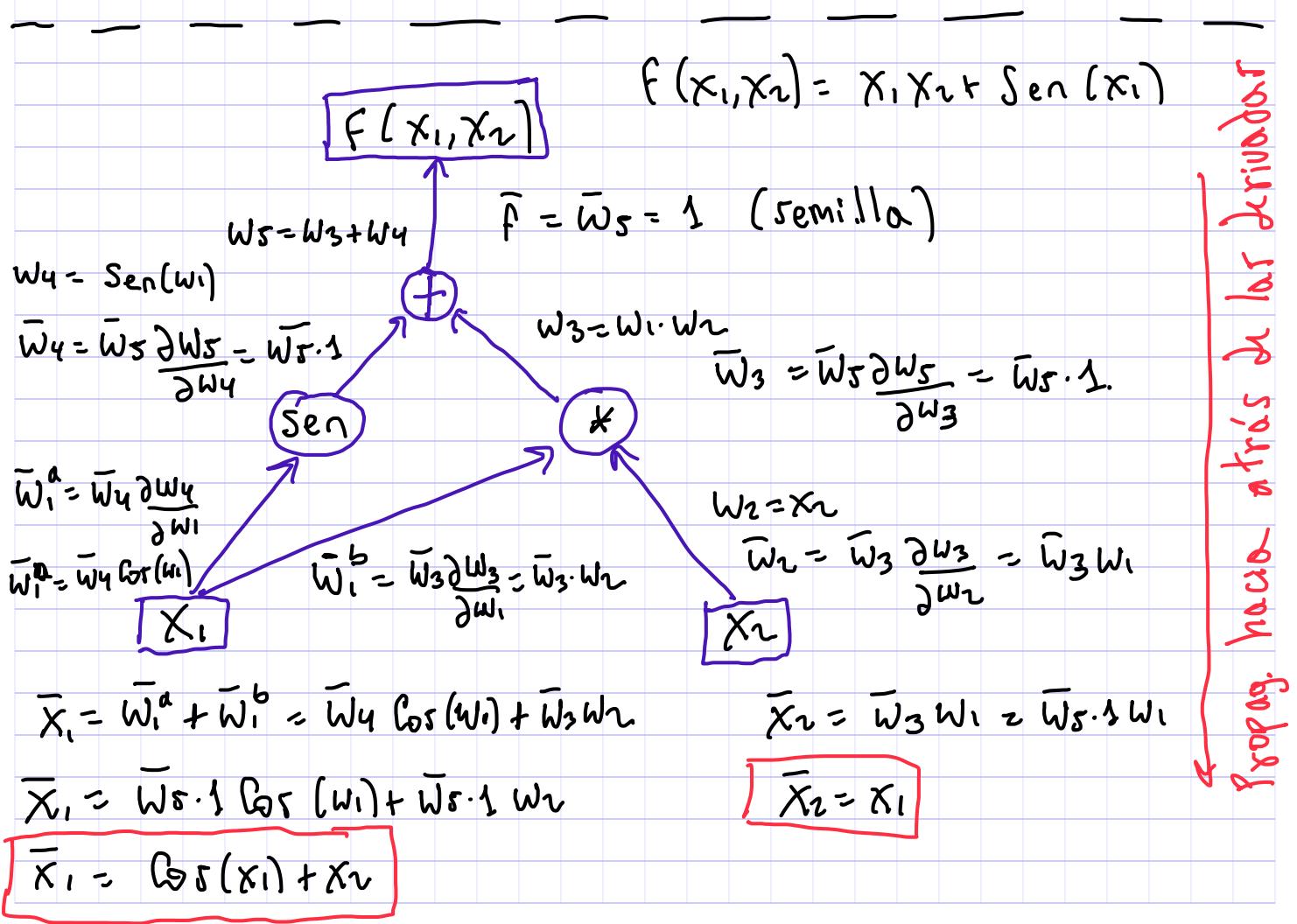
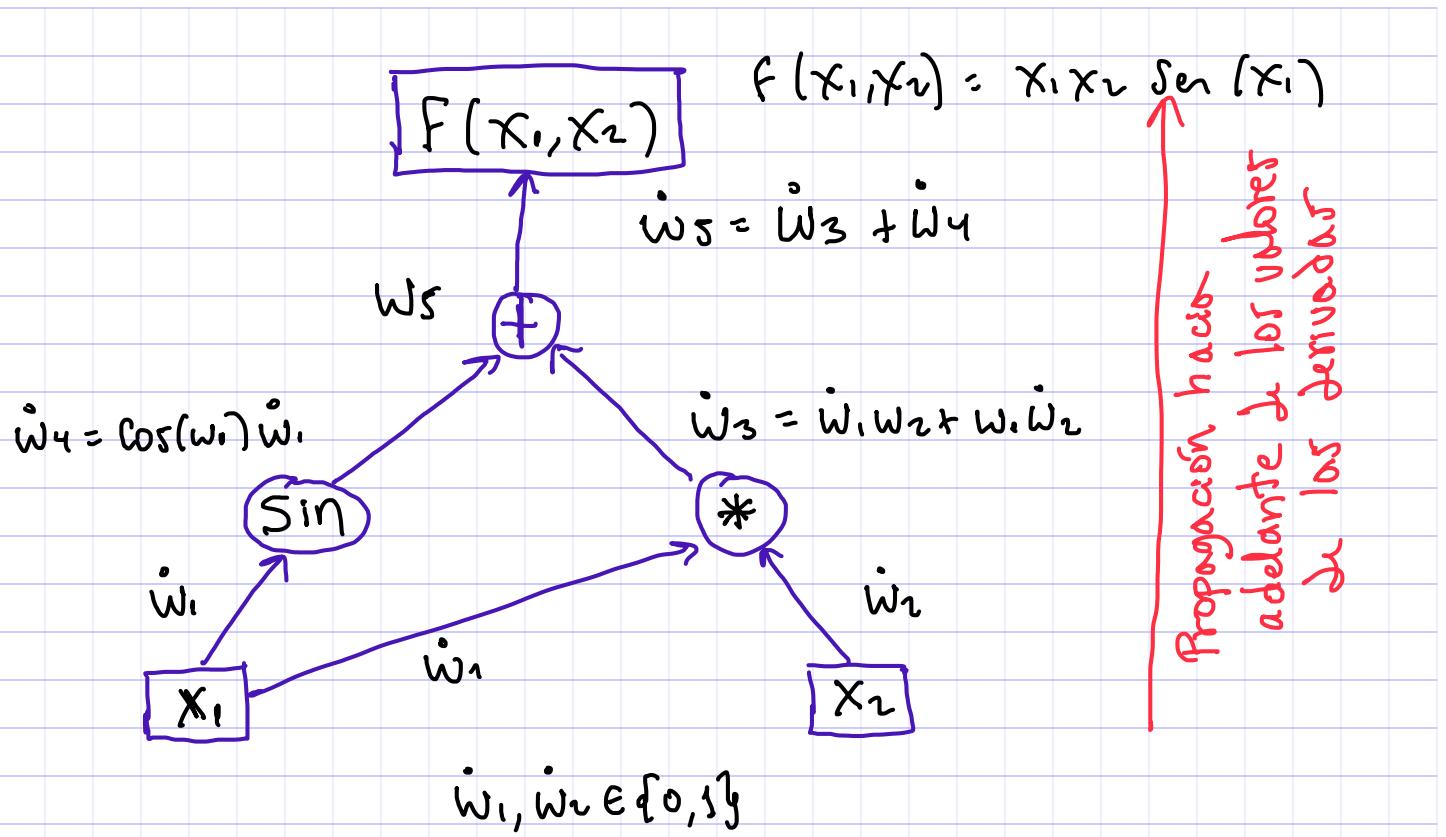
Pasos AD:

1. Descomponer la función (código) en funciones

intrínsecas \rightarrow NODOS EN GRAFO COMPUTACIONAL
EN DEEP LEARNING

2. Perseguir funciones intrínsecas \rightarrow MODELOS SIMBÓLICOS
EN NÚMEROS DUALES.

3. Operar según reglas de la cadena.



El Jacobiano en AD

Sea la función $f: \mathbb{R}^N \rightarrow \mathbb{R}^M$ con Jacobiano

$$J = DF(x) \in \mathbb{R}^{M \times N}$$

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_M}{\partial x_1} & \frac{\partial f_M}{\partial x_2} & \cdots & \frac{\partial f_M}{\partial x_N} \end{bmatrix}_{M \times N}$$

→ Forward model: calcula vector columna de J ,
 $J\dot{x}$; \dot{x} : vector columna semilla

→ Reverse model: calcula vector fila de J ,
 $\bar{y}^T J$; \bar{y} : vector fila semilla.

→ Costos computacionales equivalentes.

→ Reverse model requiere + memoria, + variables intermedias

→ Forward model AD más adecuado para:

$$g: \mathbb{R} \rightarrow \mathbb{R}^M$$

→ Reverse model AD más adecuado para:

$$f: \mathbb{R}^N \rightarrow \mathbb{R} \quad \Rightarrow \text{DEEP LEARNING CONVENCIONAL}$$

Por qué AD en redes neuronales?

→ En general: $s = f(x; w) = Wx$

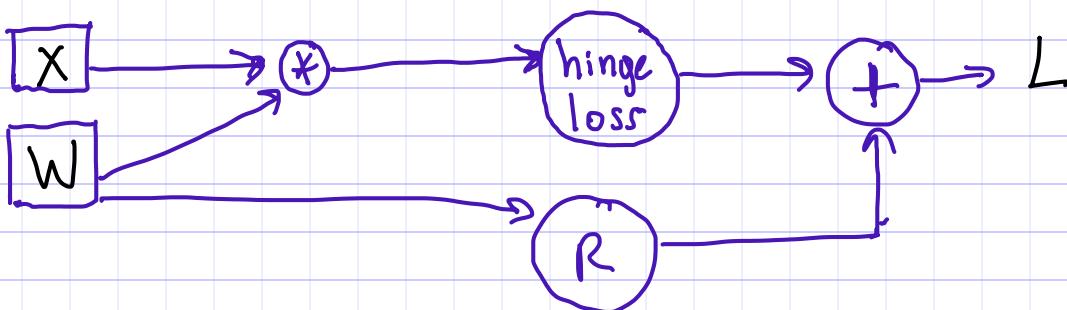
(claseficación) $L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$; Hinge loss (SVM)

$$L = \frac{1}{N} \sum_{i=1}^N L_i + \sum_k w_k^2; R(w) = \sum_k w_k^2$$

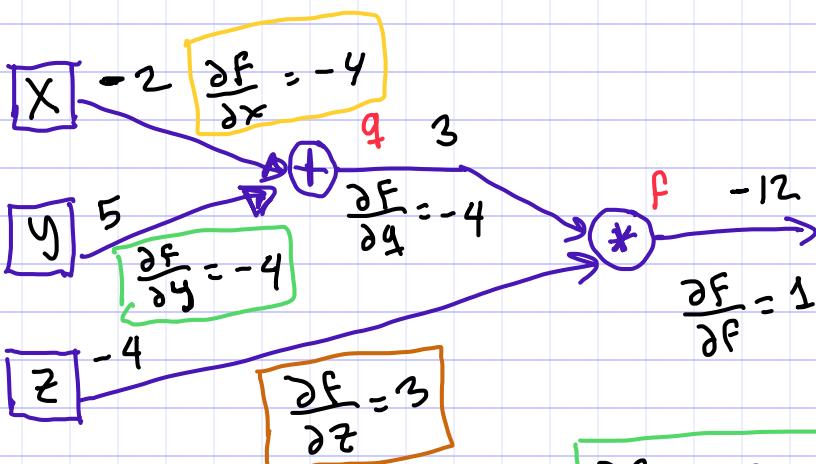
→ Se requiere: $\nabla_w L$

Grafo computacional:

$$f(x) = Wx$$



Backpropagation. $f(x, y, z) = (x+y)z; x=-2, y=5, z=-4$



$$g = x + y; \frac{\partial g}{\partial x} = 1$$

$$\frac{\partial g}{\partial y} = 1$$

$$F = g z; \frac{\partial F}{\partial g} = z$$

$$\frac{\partial F}{\partial y} = \frac{\partial F}{\partial g} \frac{\partial g}{\partial y}$$

$$\frac{\partial F}{\partial z} = g$$

$$\frac{\partial F}{\partial x} = \frac{\partial F}{\partial g} \frac{\partial g}{\partial x}$$

$$\nabla L = \left[\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}, \frac{\partial F}{\partial z} \right]$$

$$\frac{\partial F}{\partial x} = \frac{\partial F}{\partial g} \frac{\partial g}{\partial x}$$