UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

IIMPI
INSTITUTO DE
INGENIERIA MECANICA
Y PRODUCCION INDUSTRIAL

Seminario en redes neuronales basadas en la Física

PINNs

[Physics-Informed Neural Networks]

MSc. Ing. Christian Díaz-Cuadro

Estaremos conversando sobre:

Estaremos conversando sobre:

- ¿Qué es una Red Neuronal?

Estaremos conversando sobre:

- ¿Qué es una Red Neuronal?
- ¿Qué es una PINN?

Estaremos conversando sobre:

- ¿Qué es una Red Neuronal?
- ¿Qué es una PINN?
- Aplicación a mecánica del sólido

Estaremos conversando sobre:

- ¿Qué es una Red Neuronal?
- ¿Qué es una PINN?
- Aplicación a mecánica del sólido
- Resultados

Estaremos conversando sobre:

- ¿Qué es una Red Neuronal?
- ¿Qué es una PINN?
- Aplicación a mecánica del sólido
- Resultados
- Ejemplo

¿Qué es una Red Neuronal?

¿Qué es una Red Neuronal?

Definición básica

Una red neuronal es un modelo computacional inspirado en la estructura y el funcionamiento del cerebro humano.

¿Qué es una Red Neuronal?

Definición básica

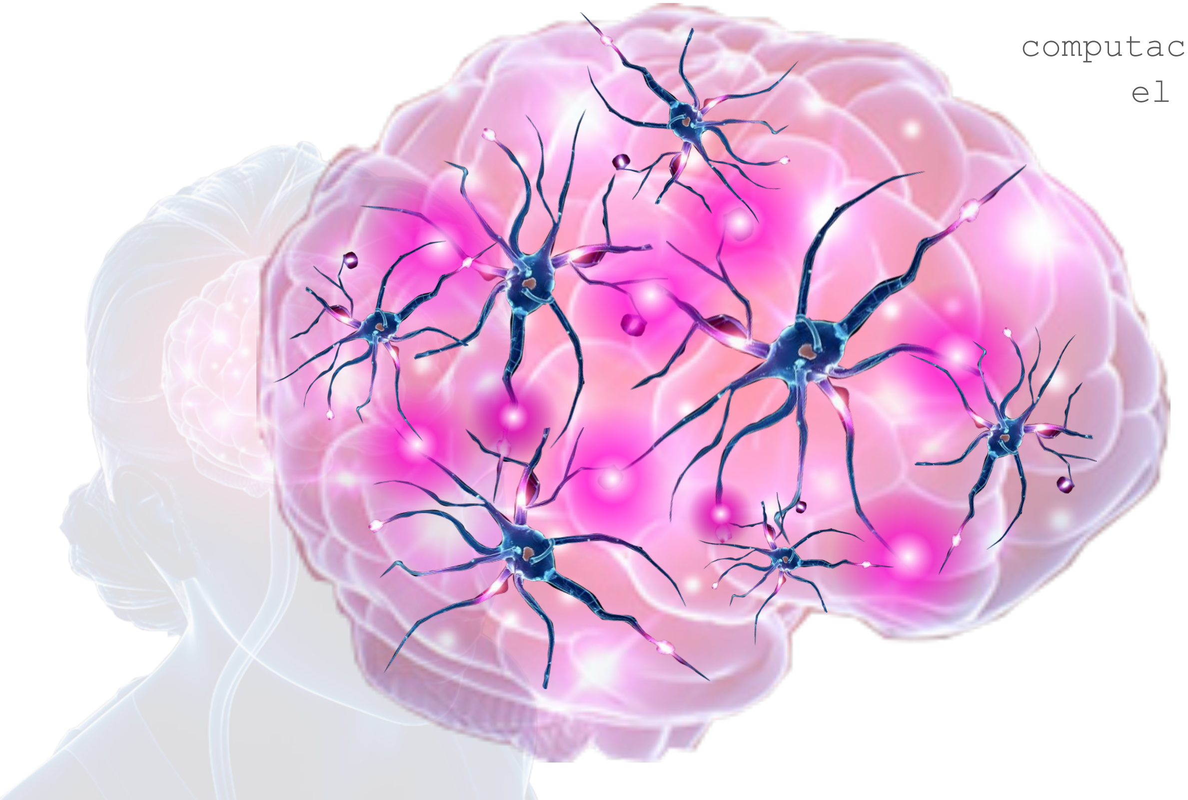
Una red neuronal es un modelo computacional inspirado en la estructura y el funcionamiento del cerebro humano.

¿Qué es una Red Neuronal?

Definición básica

Una red neuronal es un modelo computacional inspirado en la estructura y el funcionamiento del cerebro humano.

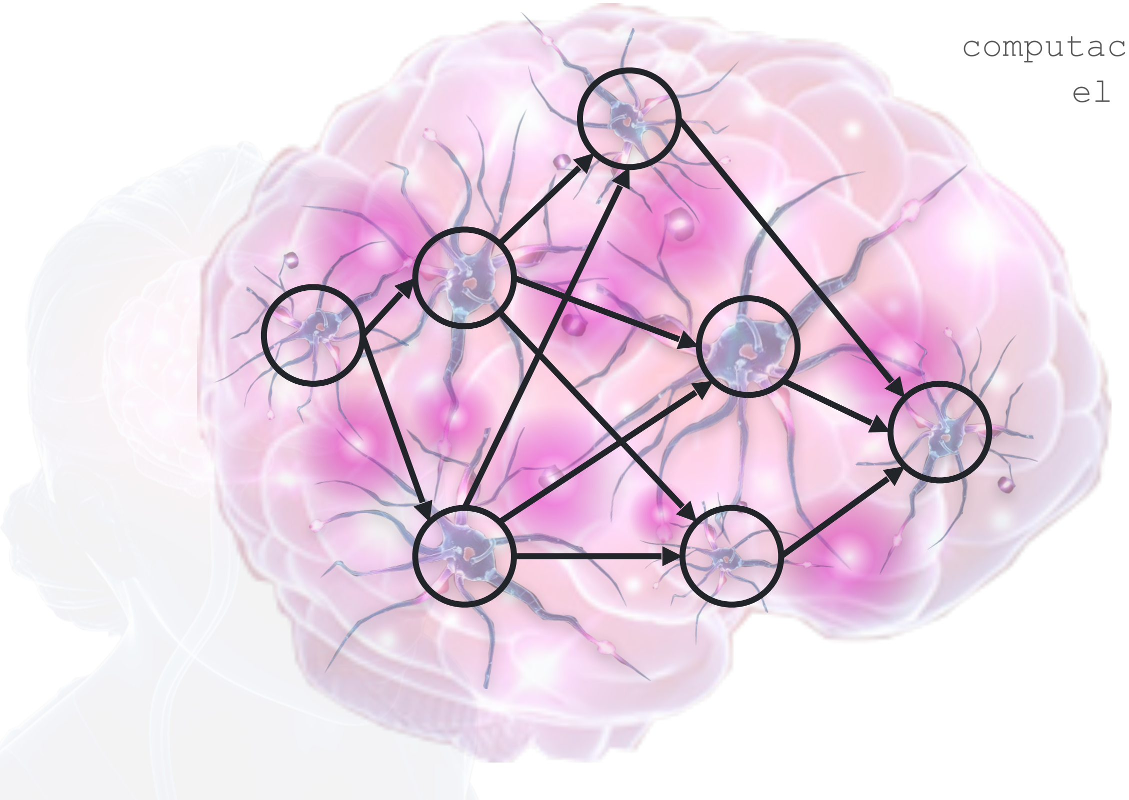
¿Qué es una Red Neuronal?



Definición básica

Una red neuronal es un modelo computacional inspirado en la estructura y el funcionamiento del cerebro humano.

¿Qué es una Red Neuronal?



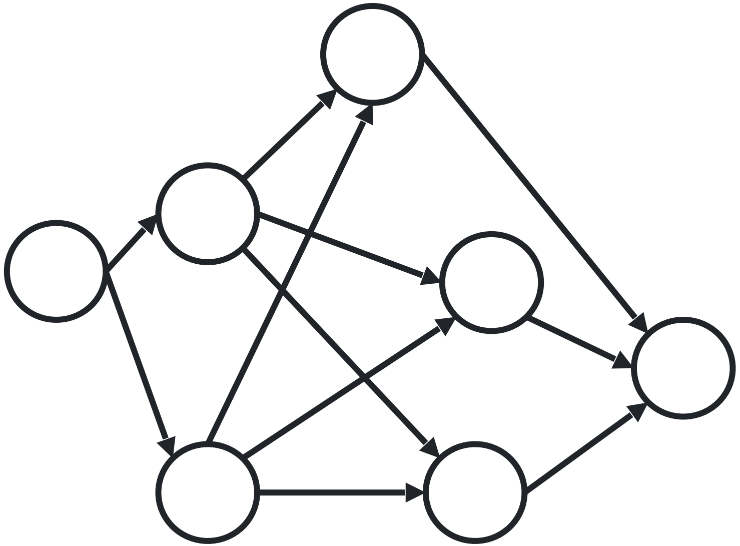
Definición básica

Una red neuronal es un modelo computacional inspirado en la estructura y el funcionamiento del cerebro humano.

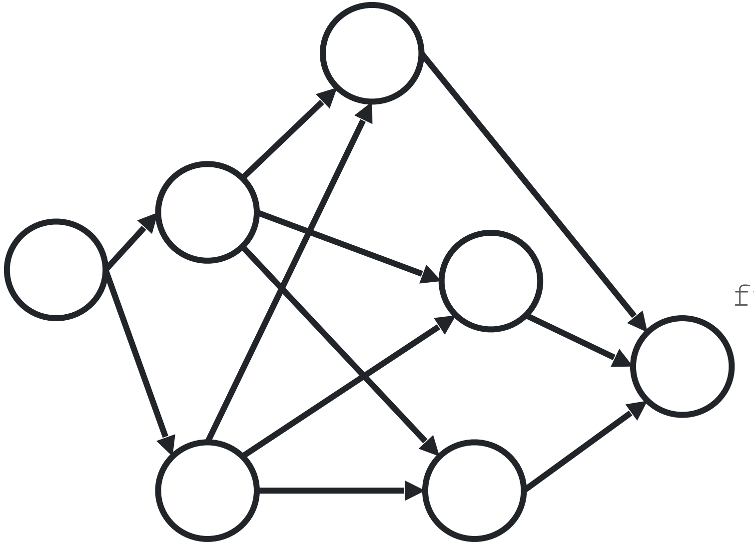
¿Qué es una Red Neuronal?

Definición básica

Una red neuronal es un modelo computacional inspirado en la estructura y el funcionamiento del cerebro humano.



¿Qué es una Red Neuronal?



¿Para qué sirve?

Es un sistema de procesamiento de información que aprende relaciones no lineales entre entradas y salidas.

Teorema de aproximación universal[1]

Se han demostrado que una Red Neuronal de **1 capa** puede aproximar cualquier función continua en el espacio $\mathbf{R}^n$, simpre que:

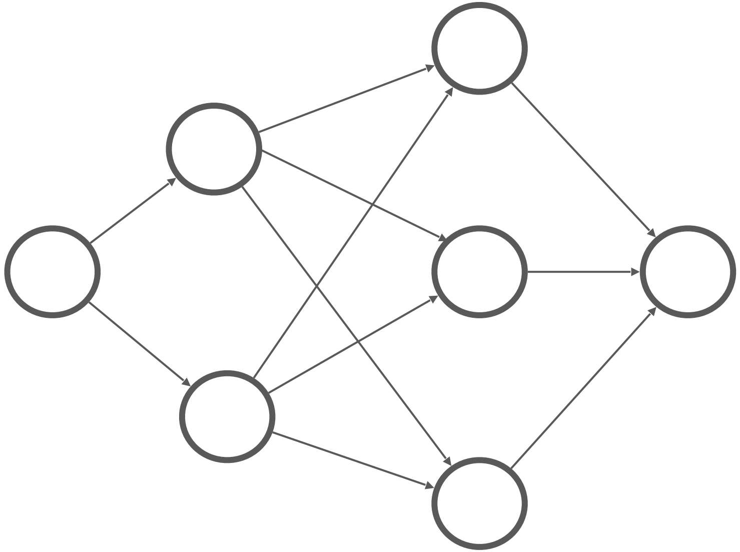
-Utilice funciones de activación apropiadas

- Sea o suficientemente "ancha"

1. Hornik, Kurt; Stinchcombe, Maxwell; White, Halbert (January 1989). "Multilayer feedforward networks are universal approximators". Neural Networks. 2 (5): 359–366.

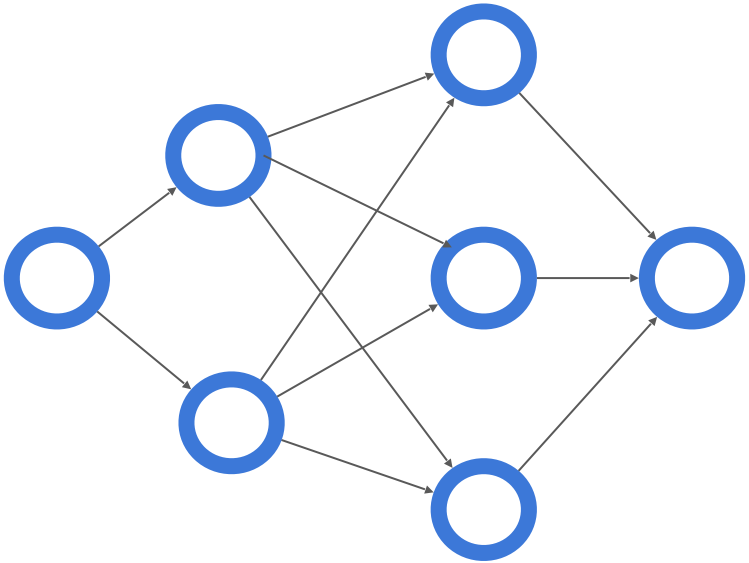
¿Qué es una Red Neuronal?

Características principales



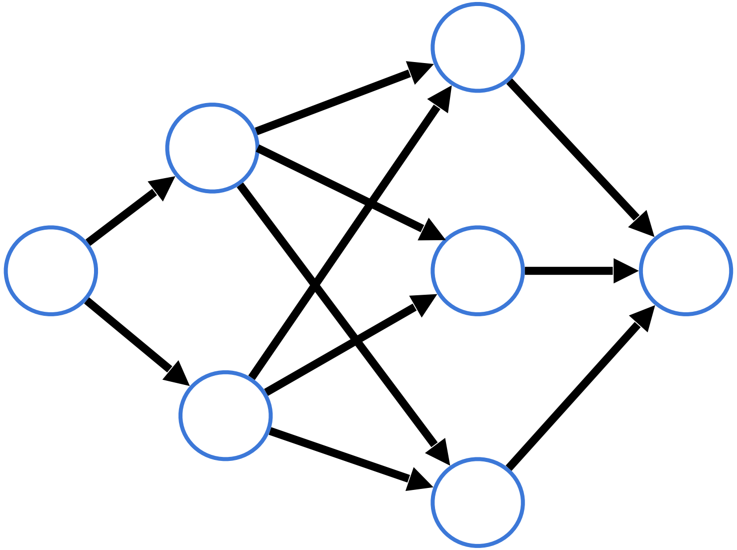
¿Qué es una Red Neuronal?

Características principales



Nodos

¿Qué es una Red Neuronal?

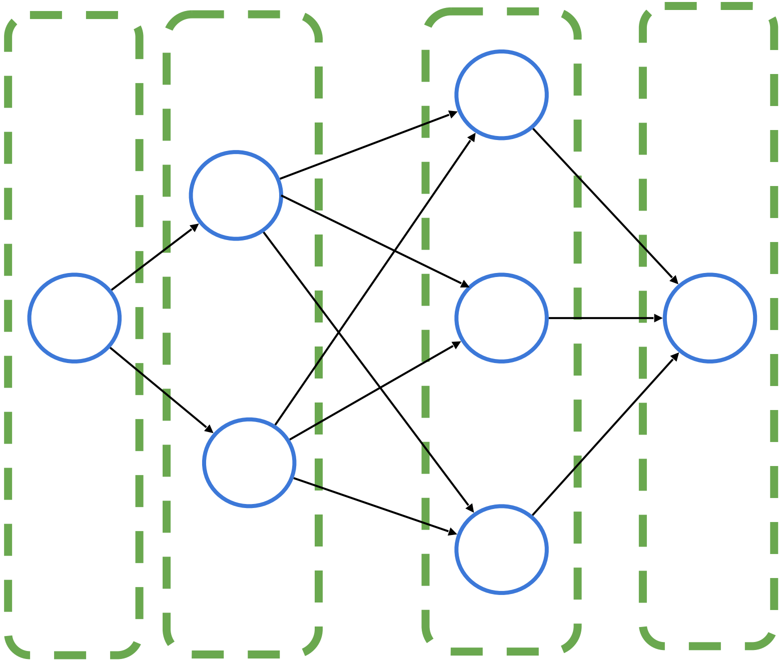


Características principales

Nodos

Conexiones

¿Qué es una Red Neuronal?



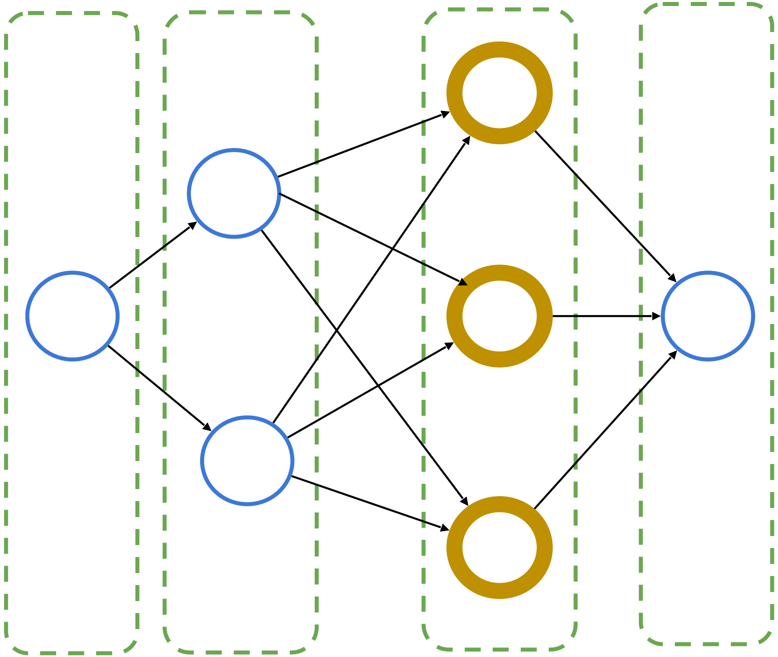
Características principales

Nodos

Conexiones

Capas (o layers)

¿Qué es una Red Neuronal?



Características principales

Nodos

Conexiones

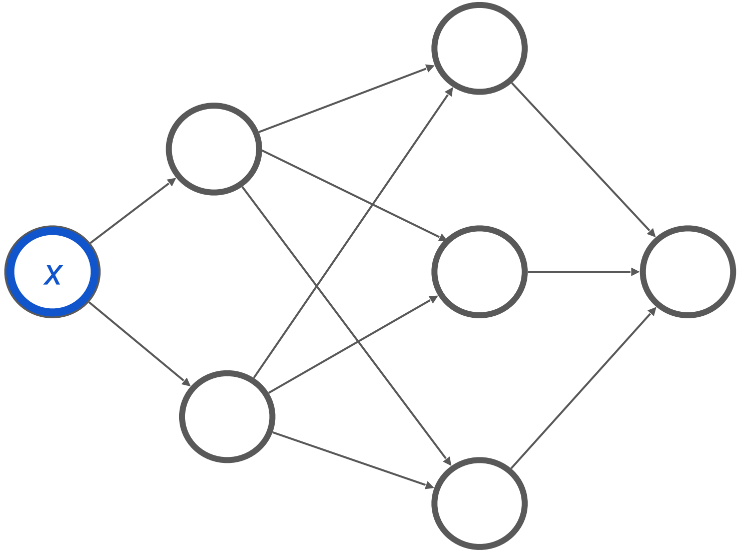
Capas (o layers)

Nodos por capa (3 en esta capa)

¿Qué es una Red Neuronal?

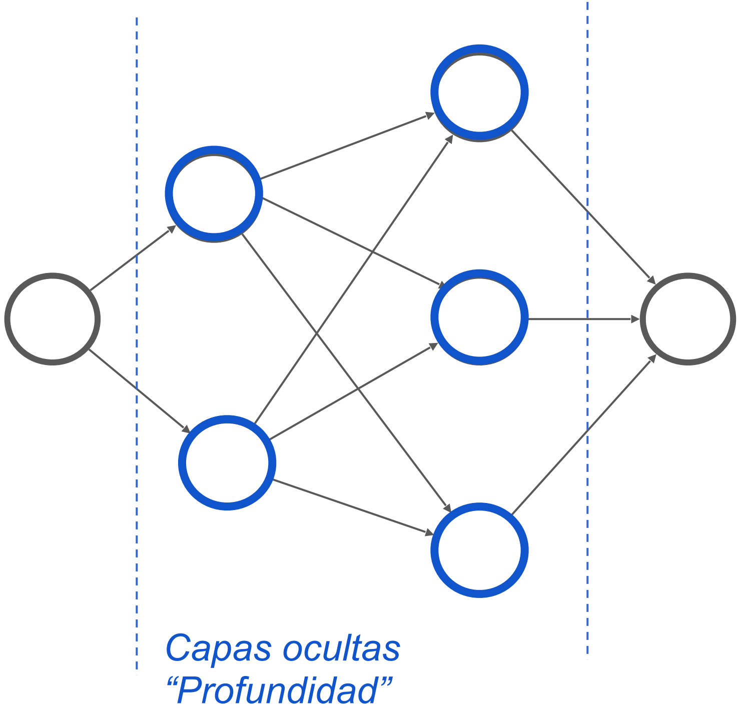
Características principales

Capa de entrada



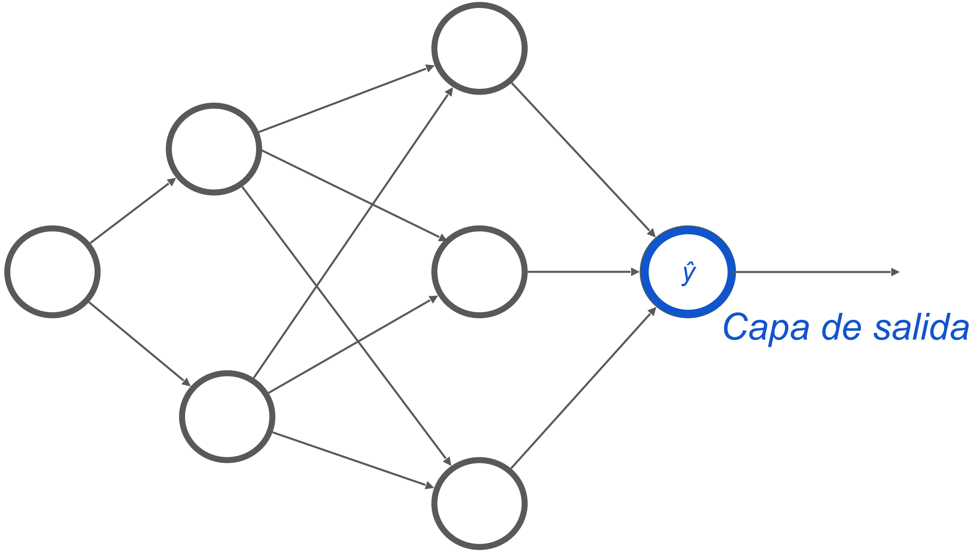
¿Qué es una Red Neuronal?

Características principales

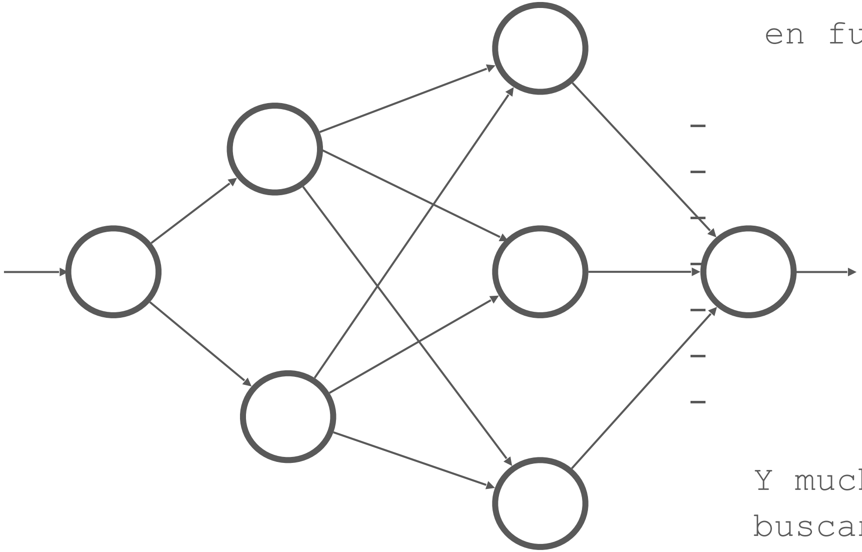


¿Qué es una Red Neuronal?

Características principales



¿Qué es una Red Neuronal?

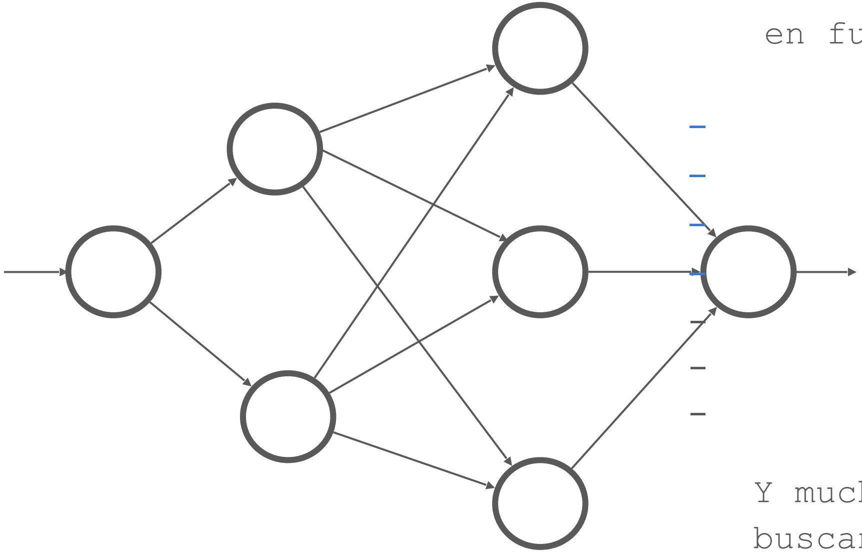


Características principales

- Esta red es conocida como **Fully connected.**
- Existen gran cantidad de variantes de **NN** en función de la operación que se realizan en las neuronas:
- Feedforward Neural Network
- Redes Neuronales Recurrentes
- Convolutional neural network
- Generative Neural Networks
- Autoencoders
- Transformers Neural Networks
- Spatial Neural Networks

Y muchas veces se combinan varias de estas buscando combinar sus fortalezas.

¿Qué es una Red Neuronal?

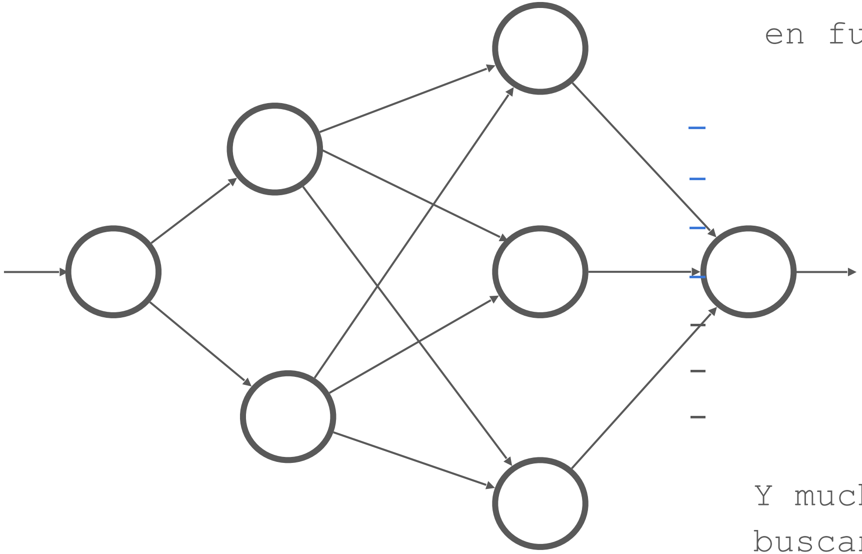


Características principales

- Esta red es conocida como **Fully connected.**
- Existen gran cantidad de variantes de **NN** en función de la operación que se realizan en las neuronas:
- **Feedforward Neural Network**
- **Redes Neuronales Recurrentes**
- **Convolutional neural network**
- **Generative Neural Networks**
- Autoencoders
- Transformers Neural Networks
- Spatial Neural Networks

Y muchas veces se combinan varias de estas buscando combinar sus fortalezas.

¿Qué es una Red Neuronal?



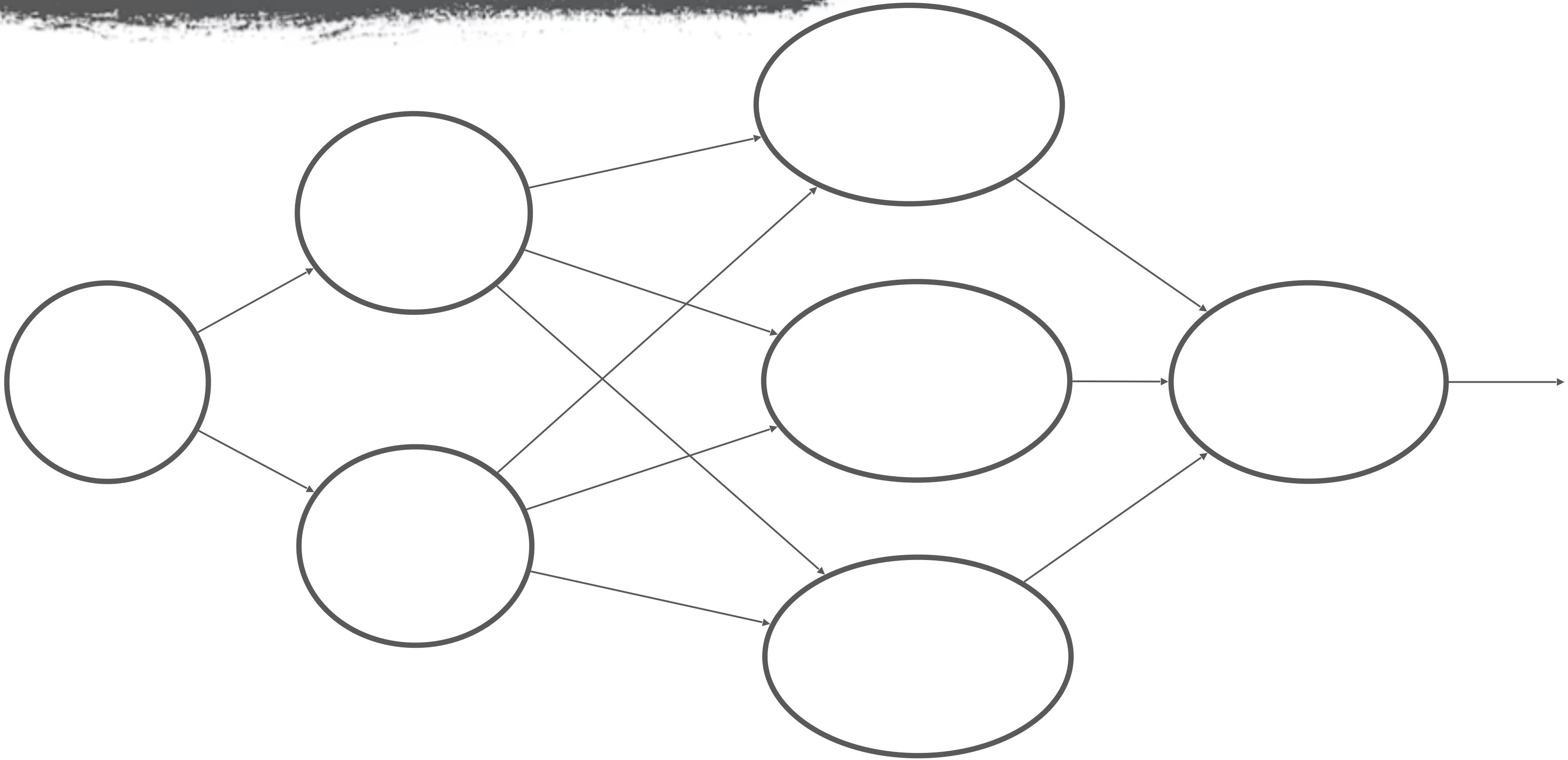
Características principales

- Esta red es conocida como **Fully connected.**
- Existen gran cantidad de variantes de **NN** en función de la operación que se realizan en las neuronas:
- **Feedforward Neural Network**
- Redes Neuronales Recurrentes
- Convolutional neural network
- Generative Neural Networks
- Autoencoders
- Transformers Neural Networks
- Spatial Neural Networks

Y muchas veces se combinan varias de estas buscando combinar sus fortalezas.

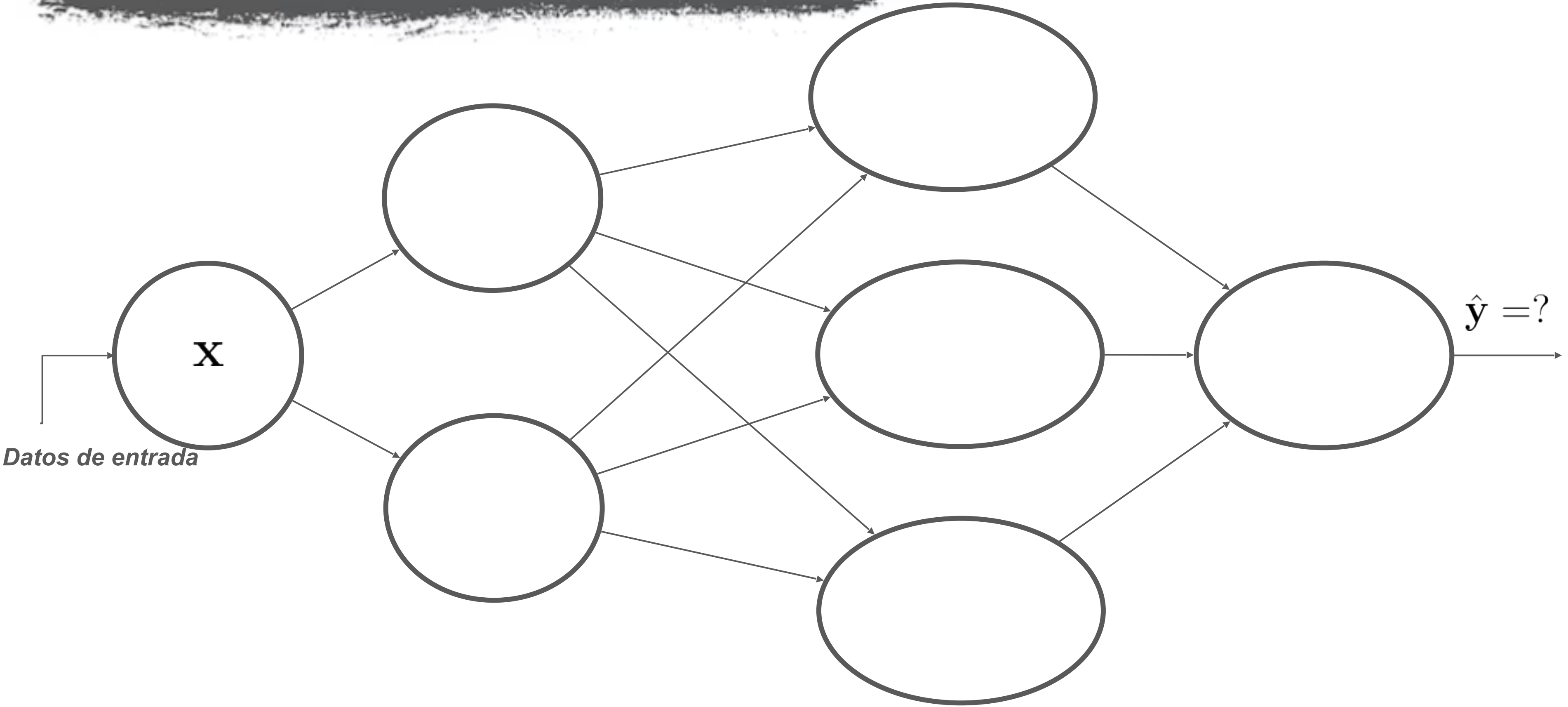
¿Qué es una Red Neuronal?

Funcionamiento



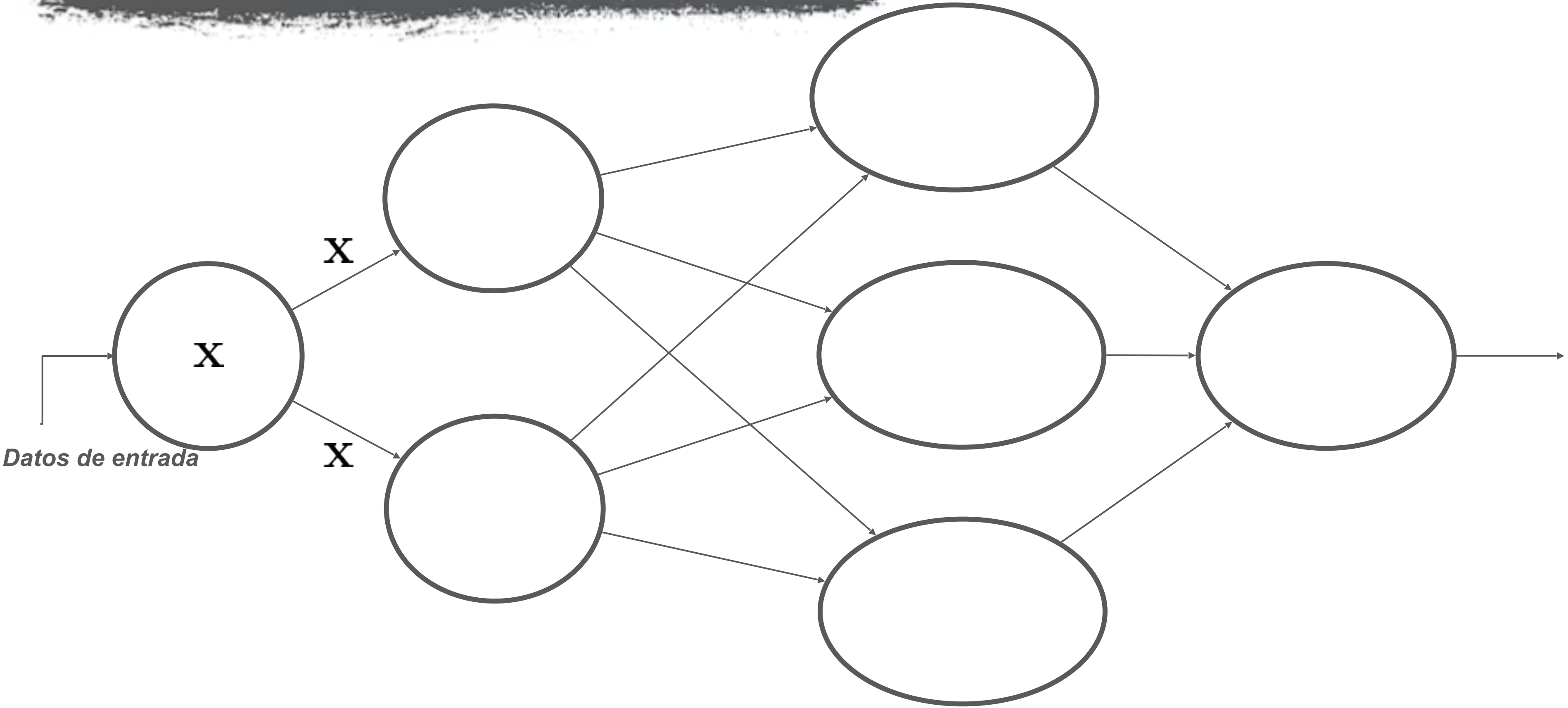
¿Qué es una Red Neuronal?

Funcionamiento



¿Qué es una Red Neuronal?

Funcionamiento

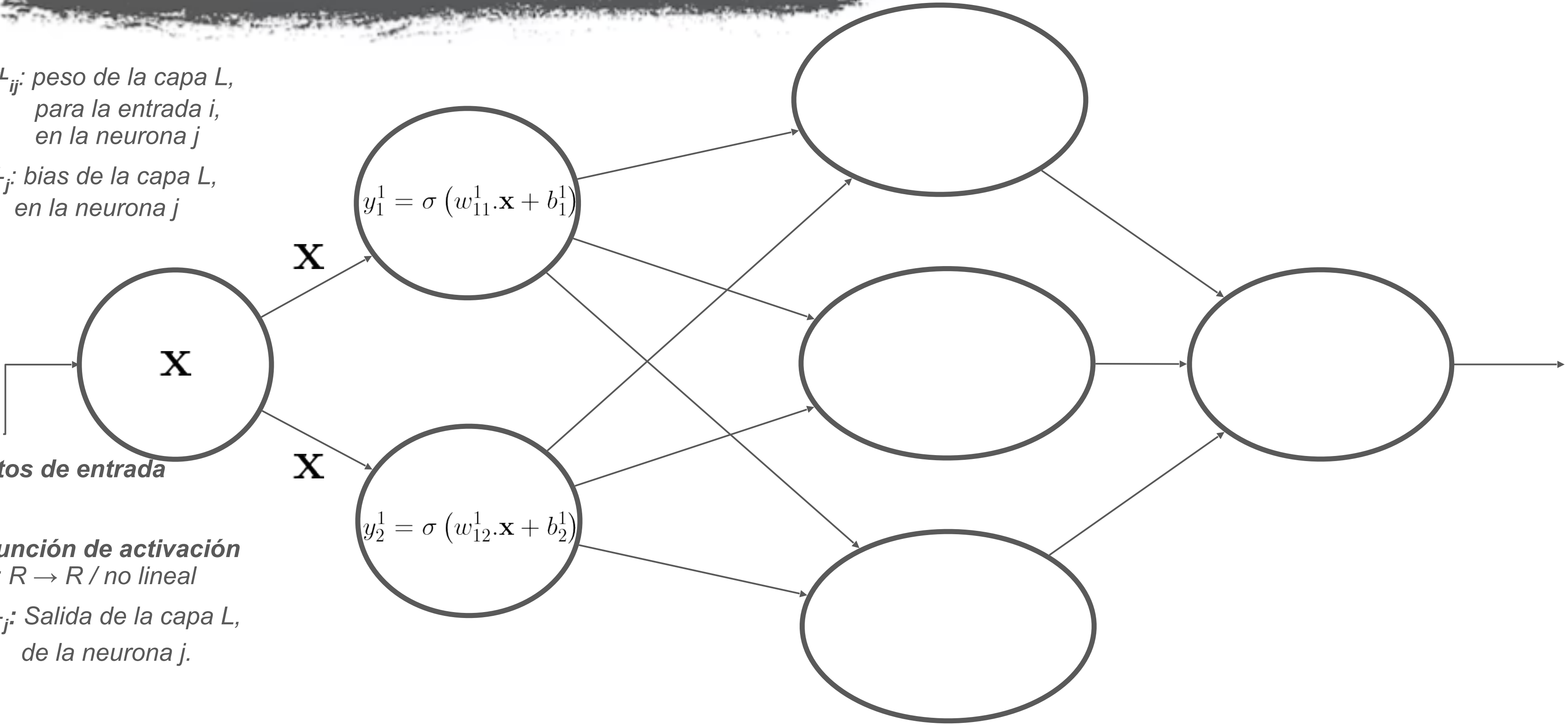


¿Qué es una Red Neuronal?

Funcionamiento

w^L_{ij}: *peso de la capa L, para la entrada i, en la neurona j*

b^L_j: *bias de la capa L, en la neurona j*



Función de activación
σ: R → R / no lineal

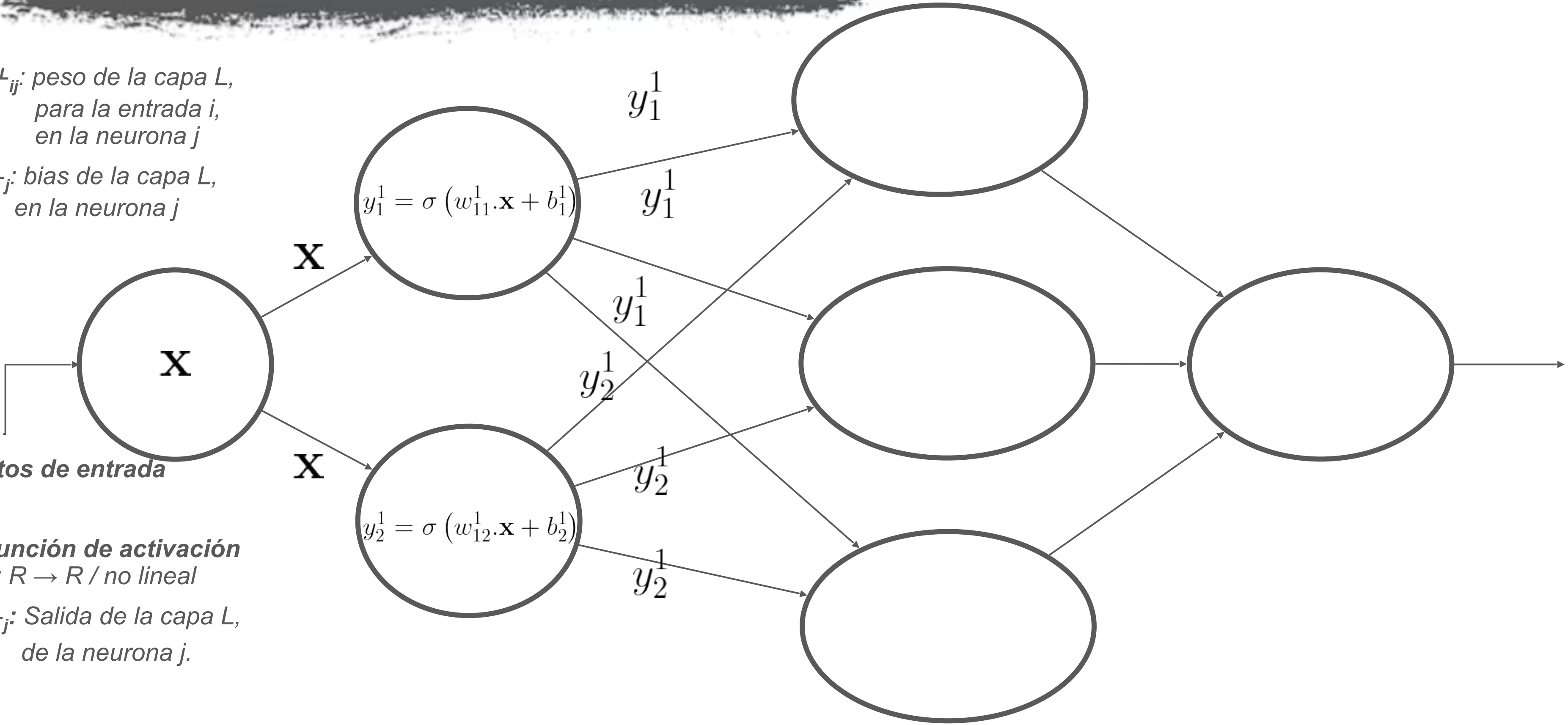
y^L_j: *Salida de la capa L, de la neurona j.*

¿Qué es una Red Neuronal?

Funcionamiento

***w^L_{ij}**: peso de la capa L, para la entrada i, en la neurona j*

***b^L_j**: bias de la capa L, en la neurona j*



Datos de entrada

Función de activación
σ: R → R / no lineal

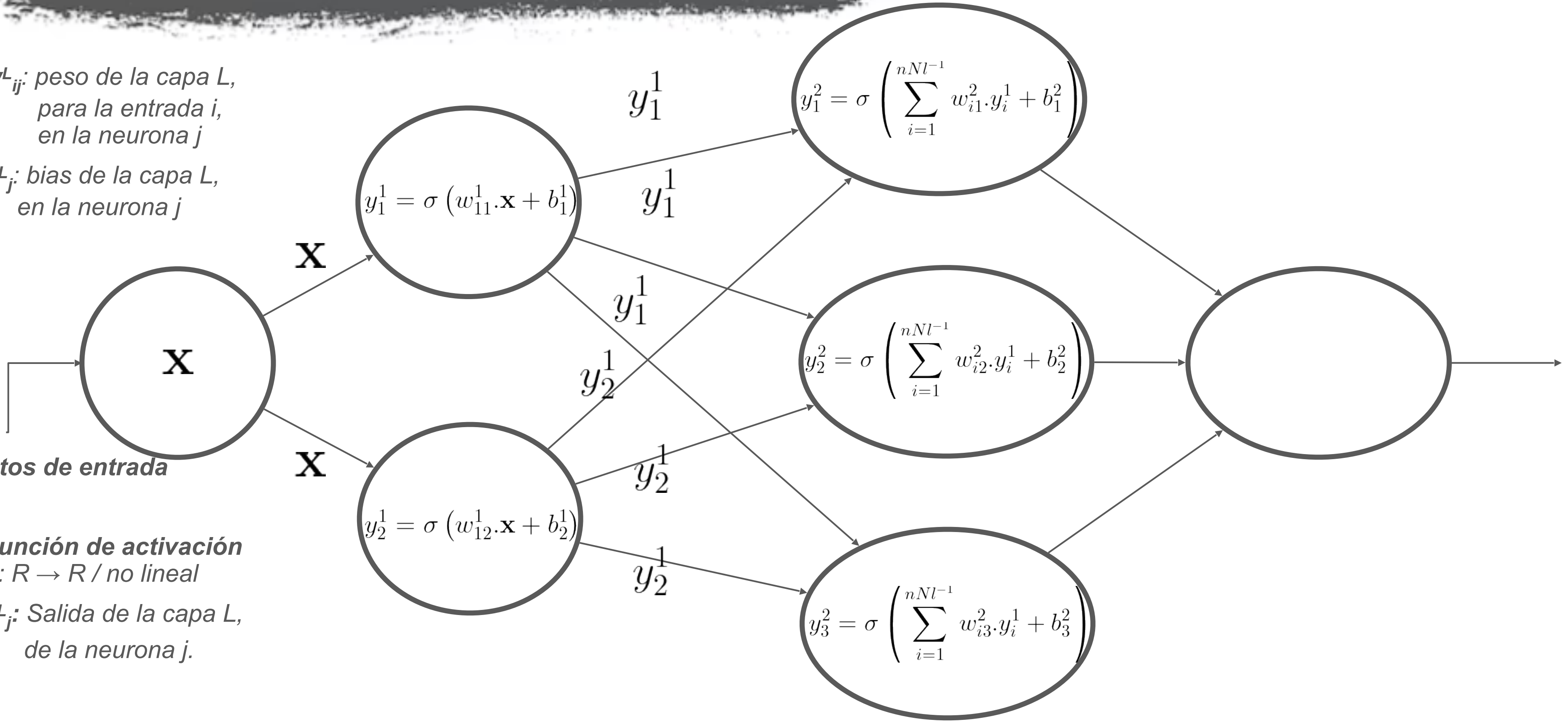
***y^L_j**: Salida de la capa L, de la neurona j.*

¿Qué es una Red Neuronal?

Funcionamiento

***w^L_{ij}**: peso de la capa L, para la entrada i, en la neurona j*

***b^L_j**: bias de la capa L, en la neurona j*



Datos de entrada

Función de activación
σ: R → R / no lineal

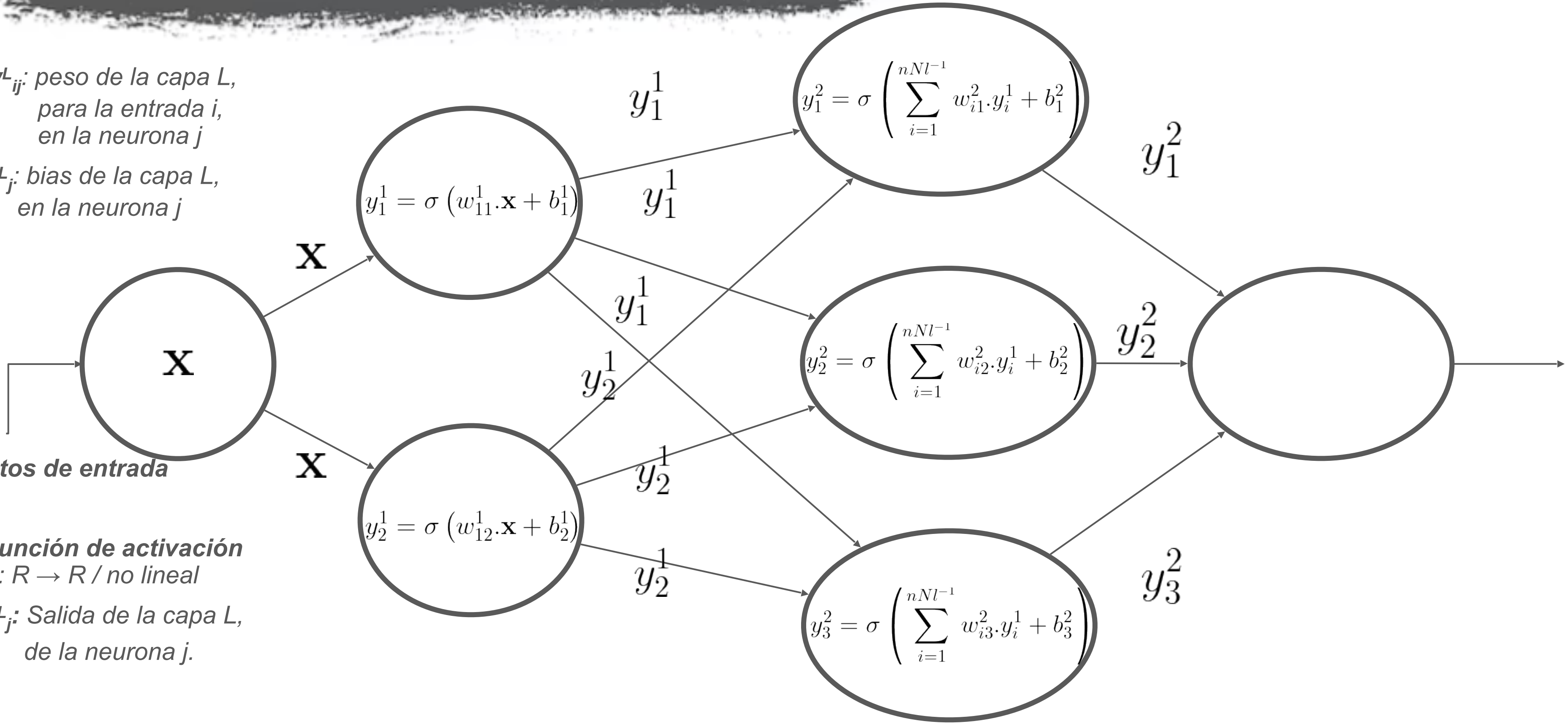
***y^L_j**: Salida de la capa L, de la neurona j.*

¿Qué es una Red Neuronal?

Funcionamiento

***w^L_{ij}**: peso de la capa L,*
para la entrada i,
en la neurona j

***b^L_j**: bias de la capa L,*
en la neurona j



Función de activación
σ: R → R / no lineal

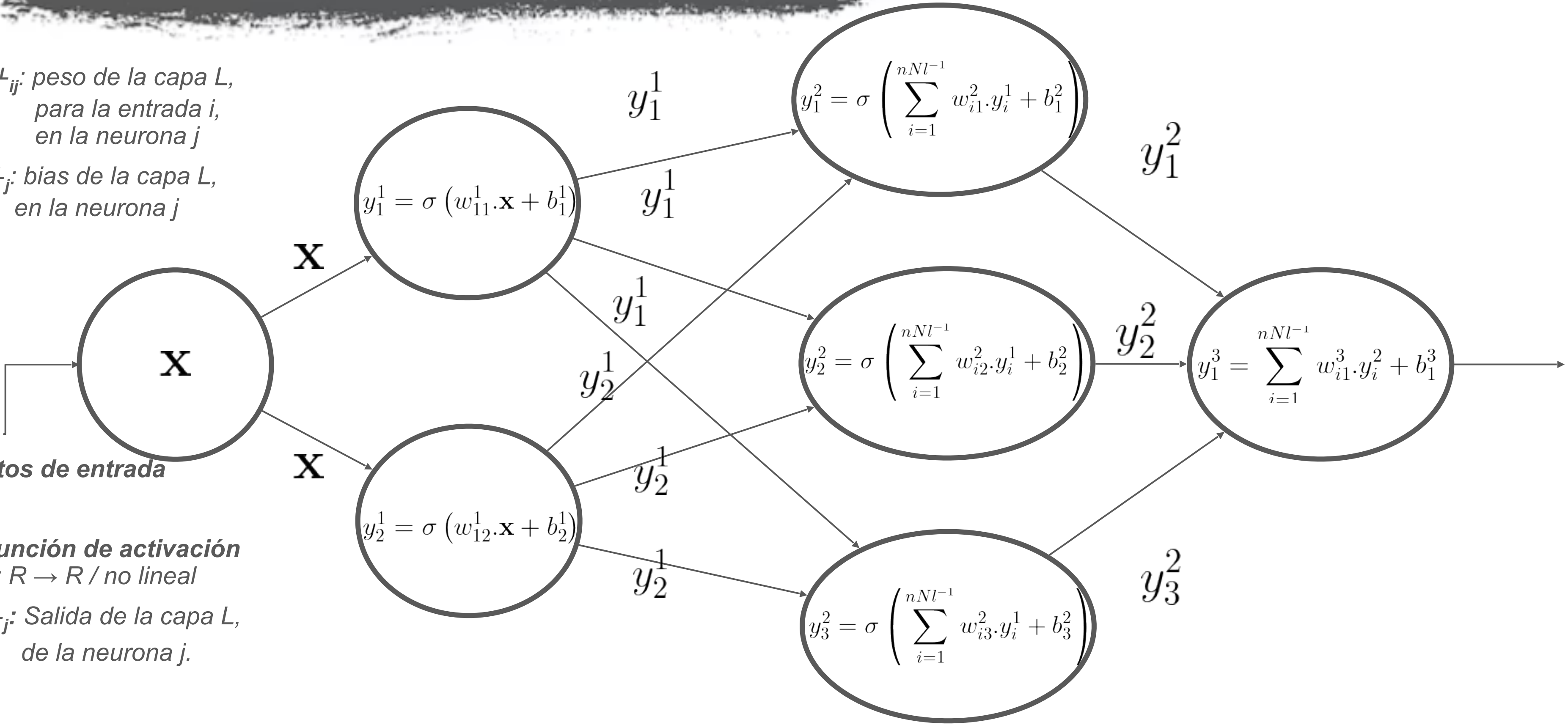
***y^L_j**: Salida de la capa L,*
de la neurona j.

¿Qué es una Red Neuronal?

Funcionamiento

***w^L_{ij}**: peso de la capa L,*
para la entrada i,
en la neurona j

***b^L_j**: bias de la capa L,*
en la neurona j



Función de activación
σ: R → R / no lineal

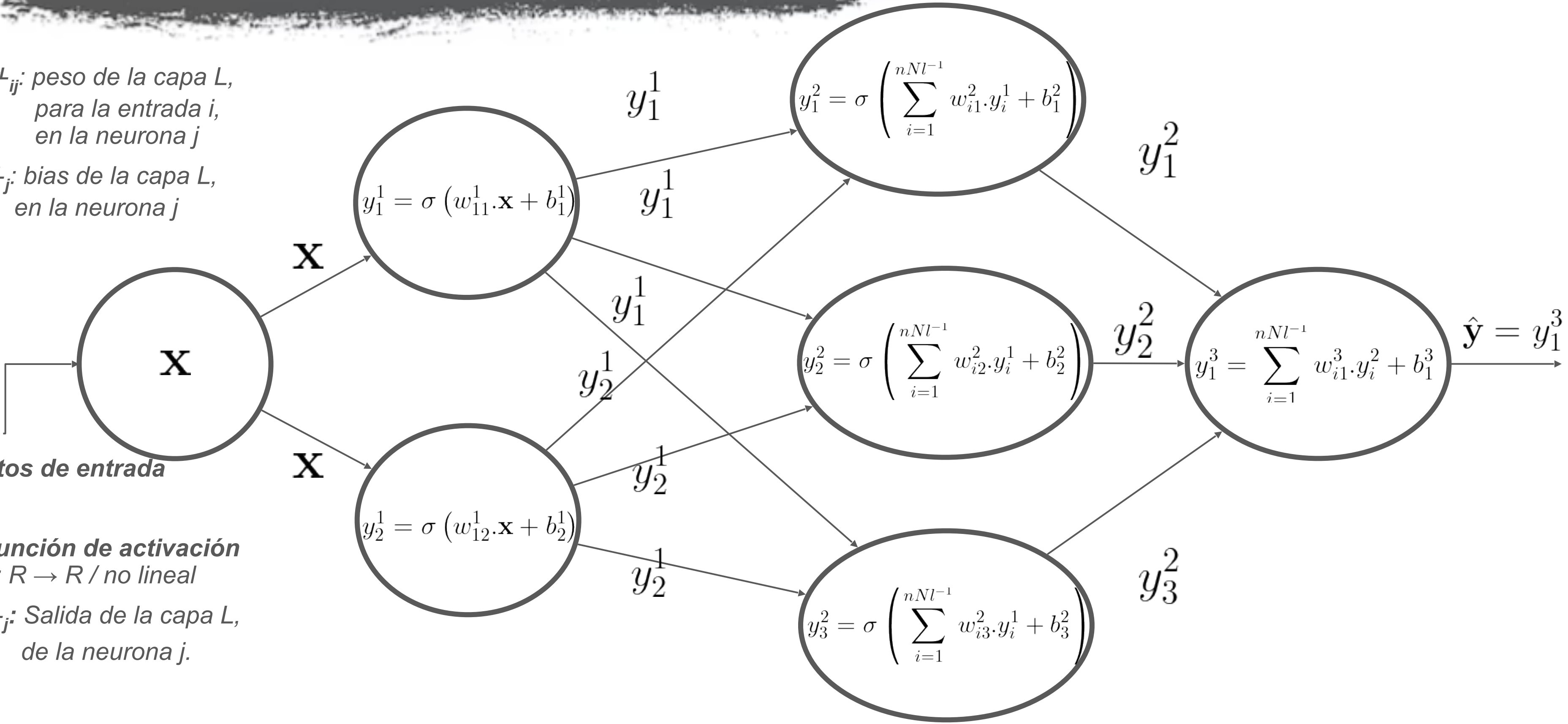
***y^L_j**: Salida de la capa L,*
de la neurona j.

¿Qué es una Red Neuronal?

Funcionamiento

***w^L_{ij}**: peso de la capa L, para la entrada i, en la neurona j*

***b^L_j**: bias de la capa L, en la neurona j*



Función de activación
σ: R → R / no lineal

***y^L_j**: Salida de la capa L, de la neurona j.*

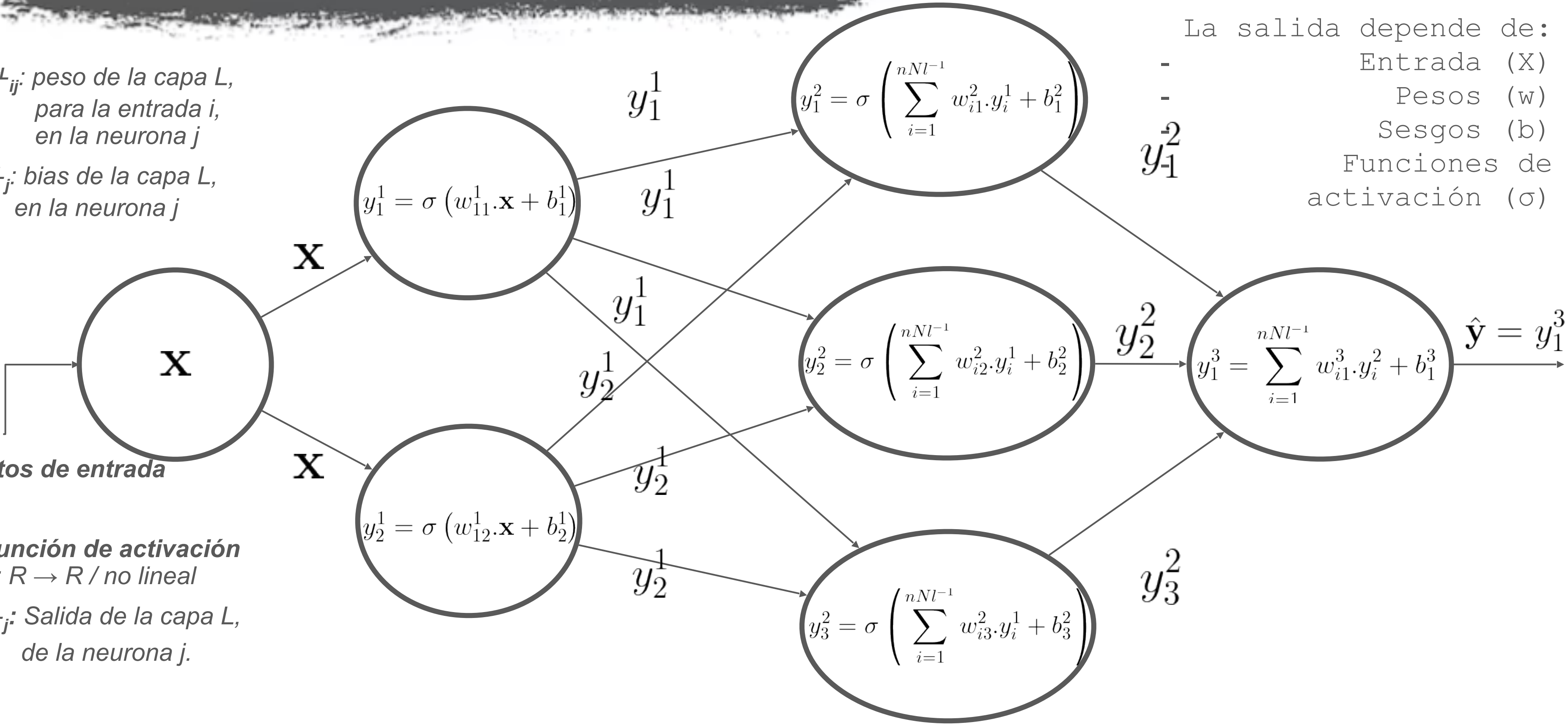
¿Qué es una Red Neuronal?

Funcionamiento

La salida depende de:
- Entrada (X)
- Pesos (w)
- Sesgos (b)
- Funciones de activación (σ)

***w^L_{ij}**: peso de la capa L, para la entrada i, en la neurona j*

***b^L_j**: bias de la capa L, en la neurona j*



Función de activación
σ: R → R / no lineal

***y^L_j**: Salida de la capa L, de la neurona j.*

¿Qué es una Red Neuronal?

Funcionamiento

La salida depende de:
- Entrada (X)
- Pesos (w)
- Sesgos (b)
- Funciones de activación (σ)

***w^L_{ij}**: peso de la capa L, para la entrada i, en la neurona j*

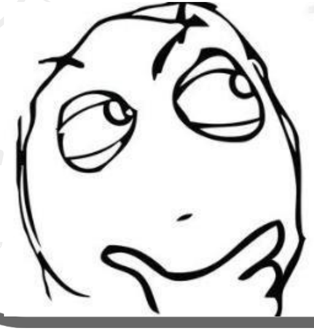
***b^L_j**: bias de la capa L, en la neurona j*

Datos de entrada

Función de activación
σ: R → R / no lineal

***y^L_j**: Salida de la capa L, de la neurona j.*

¿Cómo hacemos que la red aprenda la relación entre *x* e *y*? es decir: f(x): y=f(x)



¿Qué es una Red Neuronal?

Funcionamiento

La salida depende de:
- Entrada (X)
- Pesos (w)
- Sesgos (b)
- Funciones de activación (σ)

w^L_{ij}: peso de la capa L, para la entrada i, en la neurona j

b^L_j: bias de la capa L, en la neurona j

Datos de entrada

Función de activación
σ: R → R / no lineal

y^L_j: Salida de la capa L, de la neurona j.

$$\hat{\mathbf{y}} = y_1^3$$

Hay que entrenarla!!

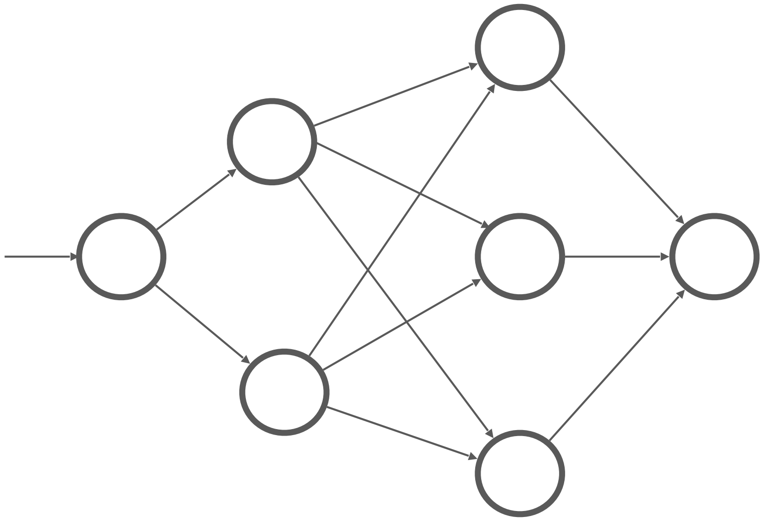
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x_d} = \{\mathbf{x_i} \in \mathbf{R^n}: \mathbf{i} = 1, \ldots \mathbf{N_d}]$

Se conoce $\mathbf{y_d} = \{\mathbf{y_i} \in \mathbf{R^m}: \mathbf{y_i} = \mathbf{f(x_i)}\}$

Siendo $\mathbf{f(x)}$ la función a estimar con $\mathbf{f(x)}: \mathbf{R^n} \rightarrow \mathbf{R^m}$



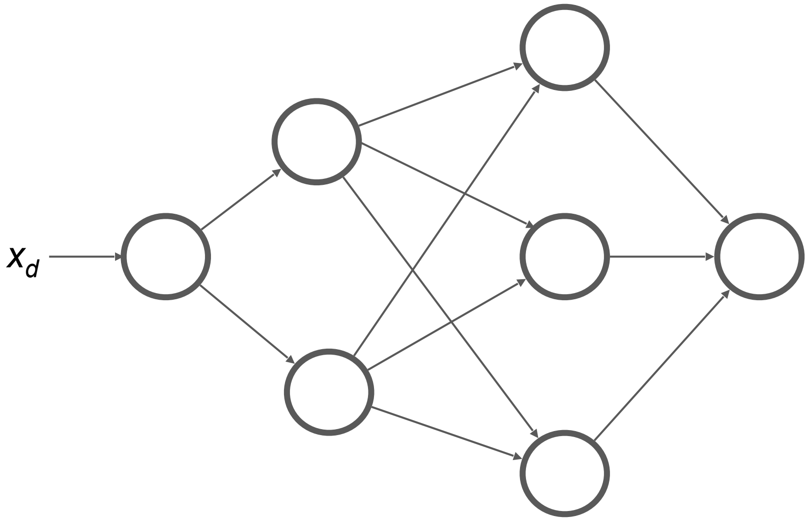
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x}_d = \{\mathbf{x}_i \in \mathbf{R}^n :\ \mathbf{i} = 1, .. \mathbf{N}_d]$

Se conoce $\mathbf{y}_d = \{\mathbf{y}_i \in \mathbf{R}^m :\ \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)\}$

Siendo $\mathbf{f(x)}$ la función a estimar con $\mathbf{f(x)}: \mathbf{R}^n \rightarrow \mathbf{R}^m$



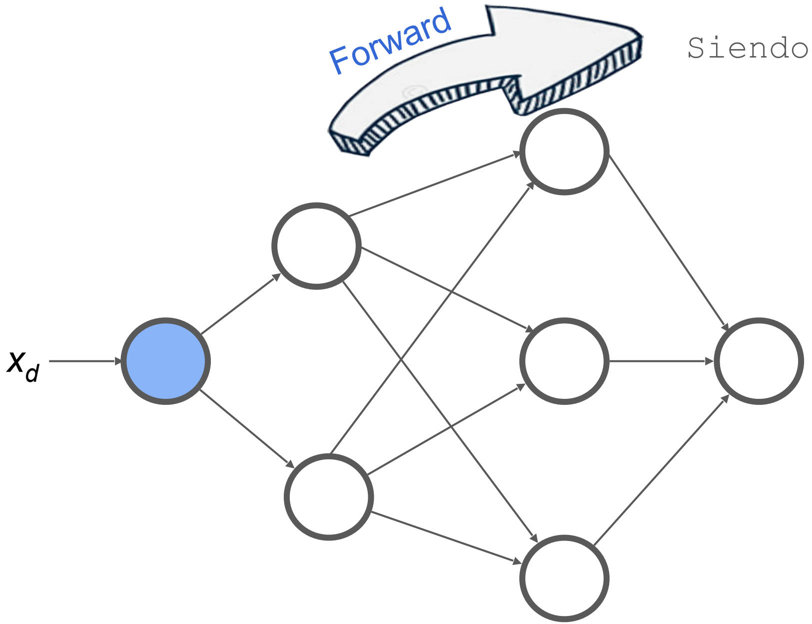
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x_d} = \{\mathbf{x_i} \in \mathbf{R^n}: \mathbf{i= 1,.. N_d}]$

Se conoce $\mathbf{y_d}=\{\mathbf{y_i} \in \mathbf{R^m}: \mathbf{y_i=f(x_i)}\}$

Siendo $\mathbf{f(x)}$ la función a estimar con $\mathbf{f(x)}: \mathbf{R^n \to R^m}$



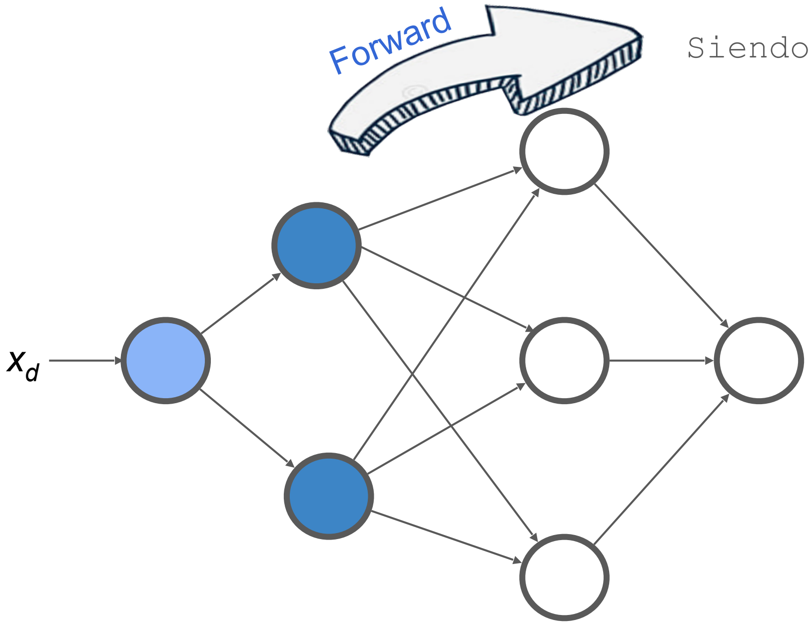
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x_d} = \{\mathbf{x_i} \in \mathbf{R^n}: \mathbf{i} = 1, .. \mathbf{N_d}]$

Se conoce $\mathbf{y_d} = \{\mathbf{y_i} \in \mathbf{R^m}: \mathbf{y_i} = \mathbf{f(x_i)}\}$

Siendo $\mathbf{f(x)}$ la función a estimar con $\mathbf{f(x)}: \mathbf{R^n} \to \mathbf{R^m}$



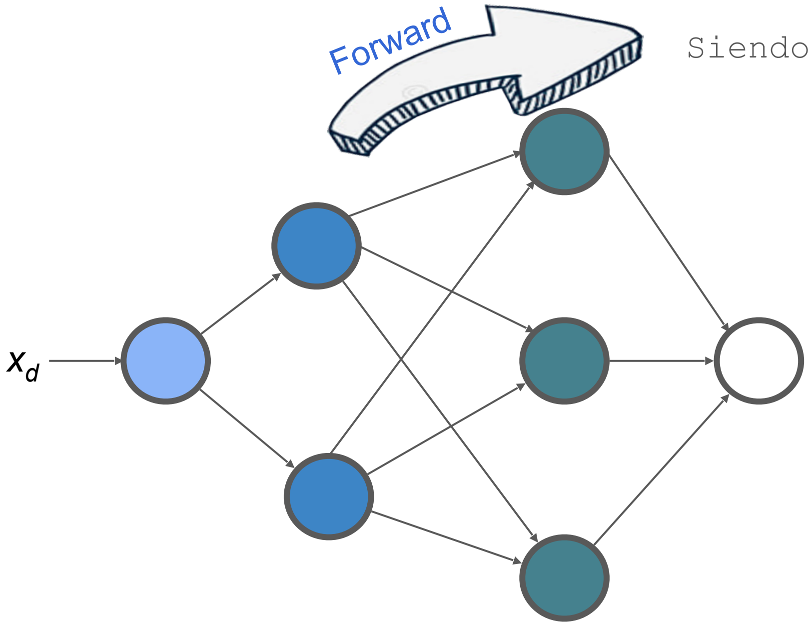
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x}_d = \{\mathbf{x}_i \in \mathbb{R}^n :\ i = 1, \ldots N_d]$

Se conoce $\mathbf{y}_d = \{\mathbf{y}_i \in \mathbb{R}^m :\ \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)\}$

Siendo $\mathbf{f}(\mathbf{x})$ la función a estimar con $\mathbf{f}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$



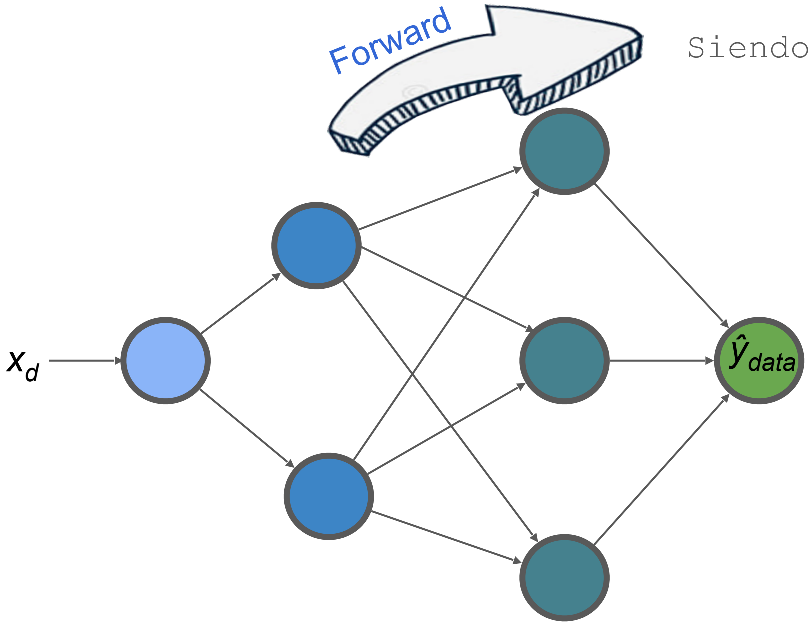
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x_d} = \{\mathbf{x_i} \in \mathbf{R^n}: \mathbf{i}= 1,.. \mathbf{N_d}\}$

Se conoce $\mathbf{y_d}=\{\mathbf{y_i} \in \mathbf{R^m}: \mathbf{y_i}=\mathbf{f(x_i)}\}$

Siendo $\mathbf{f(x)}$ la función a estimar con $\mathbf{f(x)}: \mathbf{R^n} \to \mathbf{R^m}$



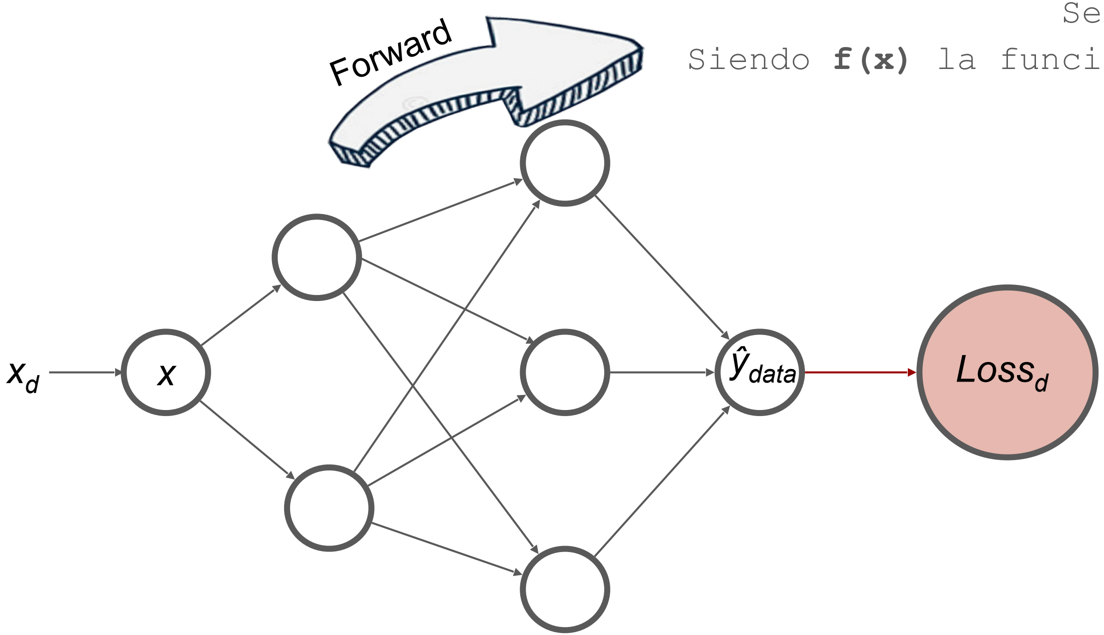
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x}_d = \{\mathbf{x}_i \in \mathbf{R}^n : \mathbf{i} = 1, \ldots \mathbf{N}_d]$

Se conoce $\mathbf{y}_d = \{\mathbf{y}_i \in \mathbf{R}^m : \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)\}$

Siendo $\mathbf{f(x)}$ la función a estimar con $\mathbf{f(x)}: \mathbf{R}^n \to \mathbf{R}^m$



Loss:

Función de pérdida que evalúa la diferencia entre el valor esperado $\mathbf{y}_d$ y el valor obtenido de la red $\hat{\mathbf{y}}_d$

$$L_d = \frac{1}{N_d} \sum_{i=1}^{N_d} ||\mathbf{y}_d - \hat{\mathbf{y}}_d||^2 = MSE(\mathbf{y}_d, \hat{\mathbf{y}}_d)$$

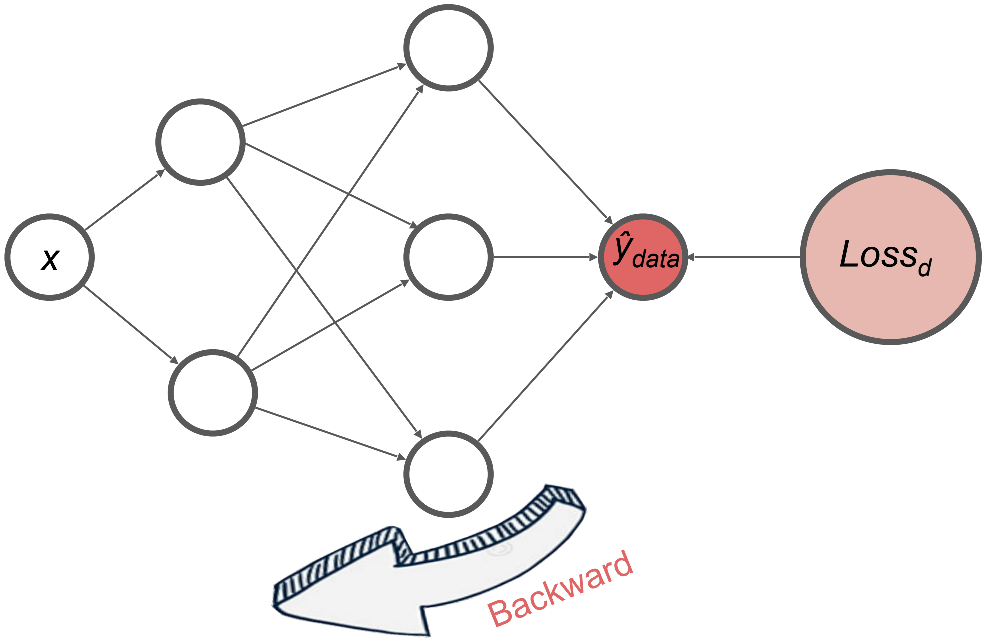
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x}_d = \{\mathbf{x}_i \in \mathbf{R}^n: i = 1, .. N_d]$

Se conoce $\mathbf{y}_d = \{\mathbf{y}_i \in \mathbf{R}^m: \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)\}$

Siendo $\mathbf{f(x)}$ la función a estimar con $\mathbf{f(x)}: \mathbf{R}^n \to \mathbf{R}^m$



Loss:

Función de pérdida que evalÚa la diferencia entre el valor esperado $\mathbf{y}_d$ y el valor obtenido de la red $\hat{\mathbf{y}}_d$

$$L_d = \frac{1}{N_d}\sum_{i=1}^{N_d} ||\mathbf{y}_d - \hat{\mathbf{y}}_d||^2 = MSE(\mathbf{y}_d, \hat{\mathbf{y}}_d)$$

Actualiza parámetros

$$\omega_{li}^{t+1} = \omega_{li}^{t} \quad - \eta\frac{\partial Loss}{\partial \omega_{li}}$$

$$b_l^{t+1} = b_l^t \quad - \eta\frac{\partial Loss}{\partial b_l}$$

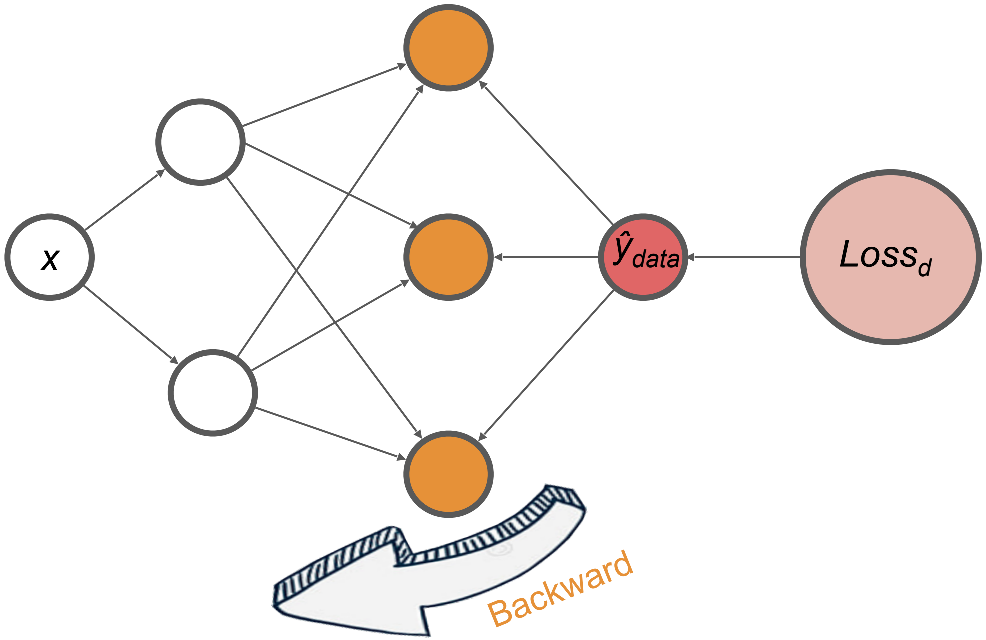
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x_d} = \{\mathbf{x_i} \in \mathbf{R^n}: \mathbf{i= 1,.. N_d}]$

Se conoce $\mathbf{y_d}=\{\mathbf{y_i} \in \mathbf{R^m}: \mathbf{y_i=f(x_i)}\}$

Siendo $\mathbf{f(x)}$ la función a estimar con $\mathbf{f(x)}: \mathbf{R^n \to R^m}$



Loss:

Función de pérdida que evalÚa la diferencia entre el valor esperado $\mathbf{y_d}$ y el valor obtenido de la red $\mathbf{\hat{y}_d}$

$$L_d = \frac{1}{N_d}\sum_{i=1}^{N_d} ||\mathbf{y}_d - \hat{\mathbf{y}}_d||^2 = MSE(\mathbf{y}_d, \hat{\mathbf{y}}_d)$$

Actualiza parámetros

$$\omega_{li}^{t+1} = \omega_{li}^{t} - \eta\frac{\partial Loss}{\partial \omega_{li}}$$

$$b_l^{t+1} = b_l^t - \eta\frac{\partial Loss}{\partial b_l}$$

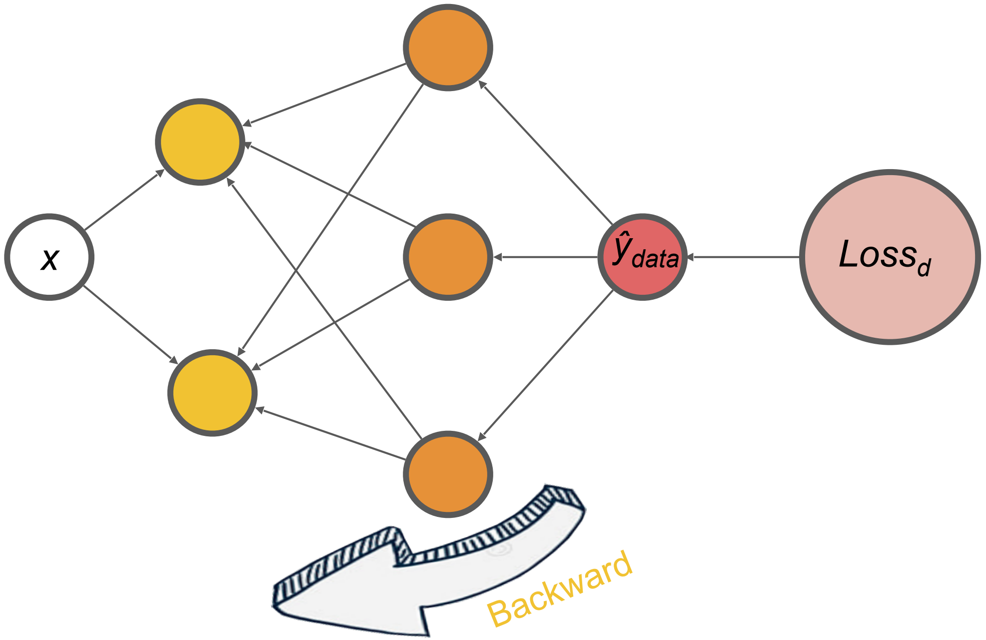
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x}_d = \{\mathbf{x}_i \in \mathbf{R}^n : i = 1, .. N_d]$

Se conoce $\mathbf{y}_d = \{\mathbf{y}_i \in \mathbf{R}^m : \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)\}$

Siendo $\mathbf{f}(\mathbf{x})$ la función a estimar con $\mathbf{f}(\mathbf{x}) : \mathbf{R}^n \rightarrow \mathbf{R}^m$



Loss:

Función de pérdida que evalÚa la diferencia entre el valor esperado $\mathbf{y}_d$ y el valor obtenido de la red $\hat{\mathbf{y}}_d$

$$L_d = \frac{1}{N_d} \sum_{i=1}^{N_d} ||\mathbf{y}_d - \hat{\mathbf{y}}_d||^2 = MSE(\mathbf{y}_d, \hat{\mathbf{y}}_d)$$

Actualiza parámetros

$$\omega_{li}^{t+1} = \omega_{li}^{t} \quad - \eta \frac{\partial Loss}{\partial \omega_{li}}$$

$$b_l^{t+1} = b_l^{t} \quad - \eta \frac{\partial Loss}{\partial b_l}$$

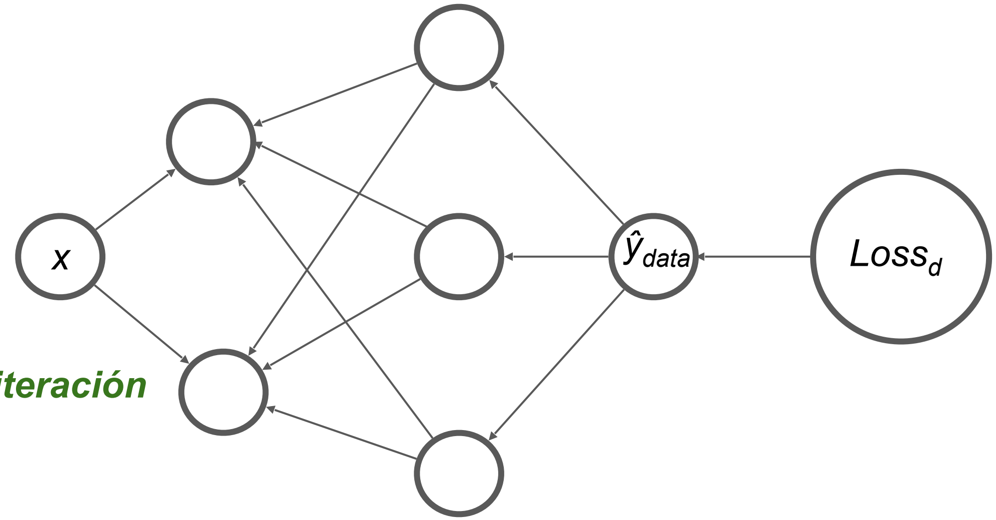
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x}_d = \{\mathbf{x}_i \in \mathbf{R}^n : i = 1, \ldots N_d]$

Se conoce $\mathbf{y}_d = \{\mathbf{y}_i \in \mathbf{R}^m : \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)\}$

Siendo $\mathbf{f}(\mathbf{x})$ la función a estimar con $\mathbf{f}(\mathbf{x}): \mathbf{R}^n \rightarrow \mathbf{R}^m$



Se completó una iteración

Loss:

Función de pérdida que evalÚa la diferencia entre el valor esperado $\mathbf{y}_d$ y el valor obtenido de la red $\hat{\mathbf{y}}_d$

$$L_d = \frac{1}{N_d}\sum_{i=1}^{N_d} ||\mathbf{y}_d - \hat{\mathbf{y}}_d||^2 = MSE(\mathbf{y}_d, \hat{\mathbf{y}}_d)$$

Actualiza parámetros

$$\omega_{li}^{t+1} = \omega_{li}^{t} \quad - \eta\frac{\partial Loss}{\partial \omega_{li}}$$

$$b_l^{t+1} = b_l^{t} \quad - \eta\frac{\partial Loss}{\partial b_l}$$

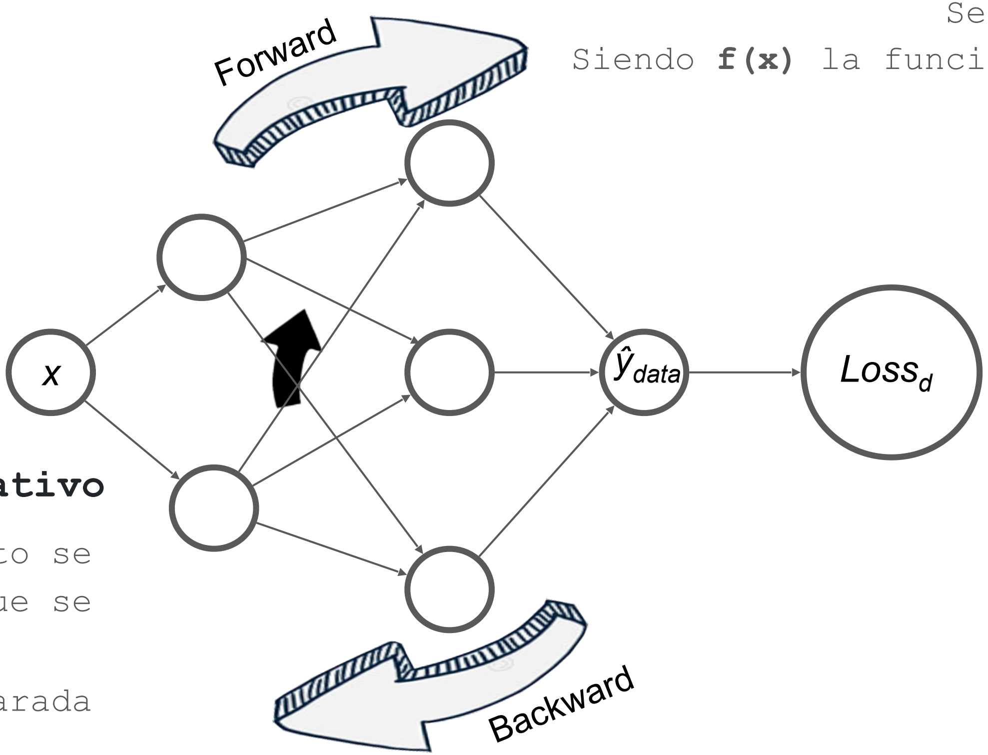
¿Qué es una Red Neuronal?

Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x}_d = \{\mathbf{x}_i \in \mathbb{R}^n: i = 1, .. N_d]$

Se conoce $\mathbf{y}_d = \{\mathbf{y}_i \in \mathbb{R}^m: \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)\}$

Siendo $\mathbf{f}(\mathbf{x})$ la función a estimar con $\mathbf{f}(\mathbf{x}): \mathbb{R}^n \rightarrow \mathbb{R}^m$



Loss:

Función de pérdida que evalÚa la diferencia entre el valor esperado $\mathbf{y}_d$ y el valor obtenido de la red $\hat{\mathbf{y}}_d$

$$L_d = \frac{1}{N_d}\sum_{i=1}^{N_d} ||\mathbf{y}_d - \hat{\mathbf{y}}_d||^2 = MSE(\mathbf{y}_d, \hat{\mathbf{y}}_d)$$

Proceso iterativo

El entrenamiento se repite hasta que se cumple alguna condición de parada

Actualiza parámetros

$$\omega_{li}^{t+1} = \omega_{li}^{t} - \eta\frac{\partial Loss}{\partial \omega_{li}}$$

$$b_{l}^{t+1} = b_{l}^{t} - \eta\frac{\partial Loss}{\partial b_{l}}$$

¿Qué es una Red Neuronal?

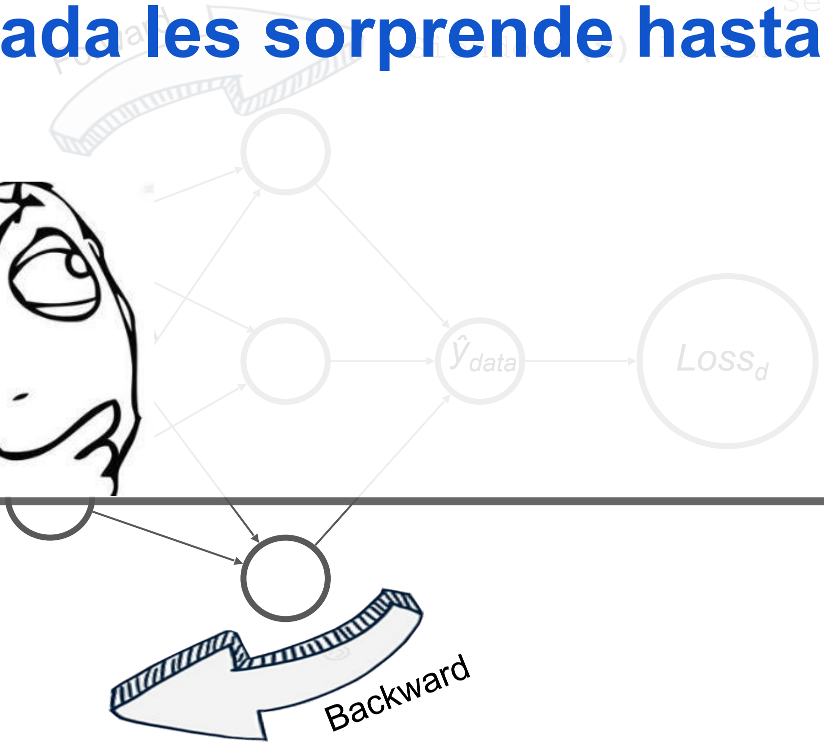
Entrenamiento

Para un conjunto de N_d datos → $x_d = \{x_i \in R^n;\ i = 1, ..\ N_d]$

Se conoce $y_d = \{y_i \in R^m;\ y_i = f(x_i)\}$

... con $f(x): R^n \to R^m$

¿Nada les sorprende hasta ahora?



Loss:

Función de pérdida que evalÚa la diferencia entre el valor esperado $\mathbf{y_d}$ y el valor obtenido de la red $\hat{\mathbf{y}}_d$

$$L_d = \frac{1}{N_d}\sum_{i=1}^{N_d} \|\mathbf{y}_d - \hat{\mathbf{y}}_d\|^2 = MSE(\mathbf{y}_d, \hat{\mathbf{y}}_d)$$

Proceso itera

El entrenamiento se repite hasta que se cumple alguna condición de parada

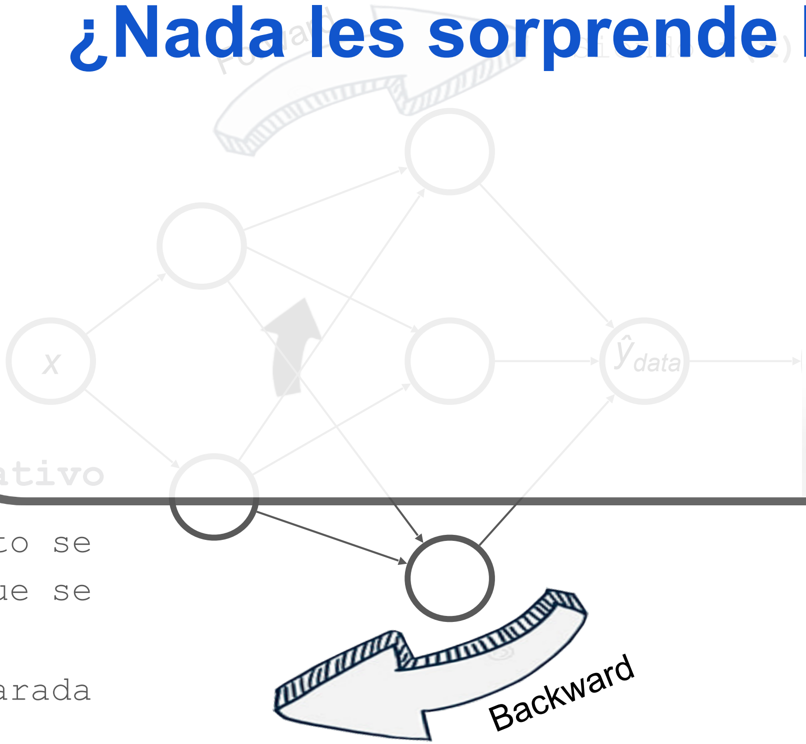
Actualiza parámetros

$$\omega_{li}^{t+1} = \omega_{li}^{t} - \eta\frac{\partial Loss}{\partial \omega_{li}}$$

$$b_l^{t+1} = b_l^{t} - \eta\frac{\partial Loss}{\partial b_l}$$

¿Qué es una Red Neuronal?

¿Nada les sorprende hasta ahora?



Entrenamiento

$i = 1, \ldots N_d]$

$y_i = f(x_i)\}$

$f(x): R^n \rightarrow R^m$

Loss:

pérdida que

cia entre

do $\mathbf{y}_d$ y el

la red $\hat{\mathbf{y}}_d$

$= MSE(\mathbf{y}_d, \hat{\mathbf{y}}_d)$

Proceso iterativo

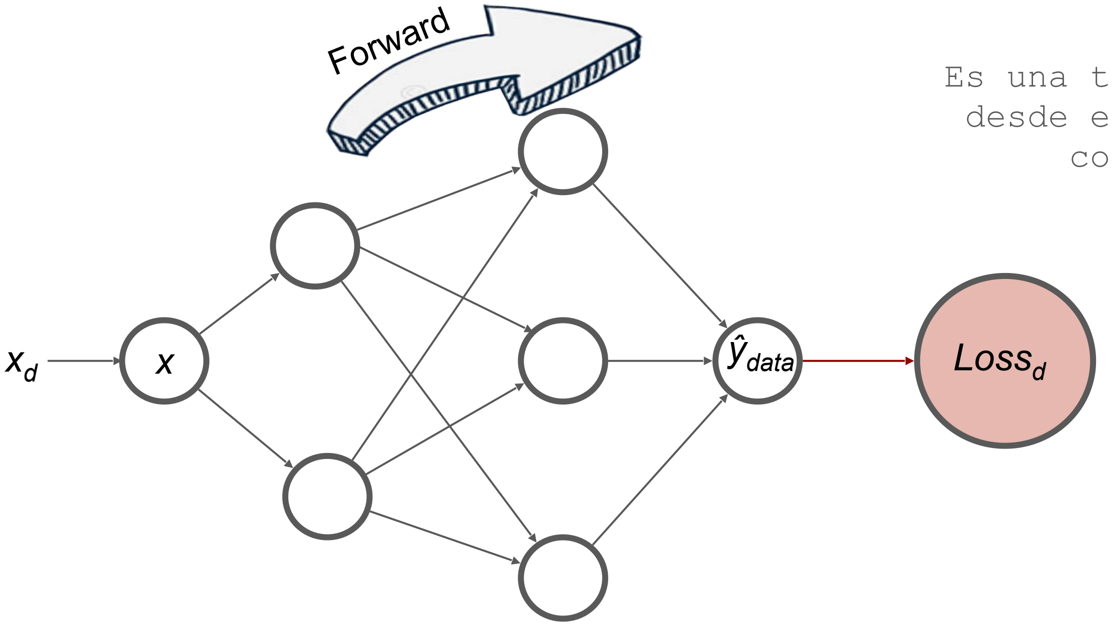
El entrenamiento se repite hasta que se cumple alguna condición de parada

Actualiza parámetros

$$\omega_{li}^{t+1} = \omega_{li}^{t} - \eta \frac{\partial Loss}{\partial \omega_{li}}$$

$$b_{l}^{t+1} = b_{l}^{t} - \eta \frac{\partial Loss}{\partial b_{l}}$$

¿Qué es una Red Neuronal?



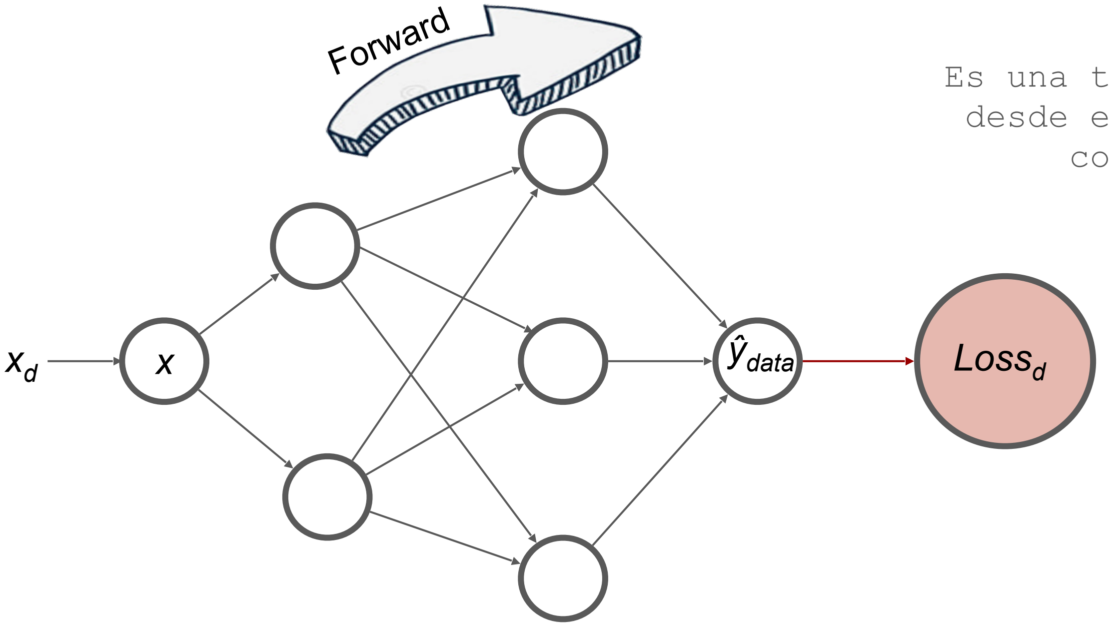
¿Cómo calculo $\dfrac{\partial Loss}{\partial \omega_{li}}$ **?**

Automatic differentiation

Es una técnica computacional que existe desde el siglo pasado pero que explotó con el aumento de la capacidad de cómputo de las últimas décadas

¿Qué es una Red Neuronal?

¿Cómo calculo $\dfrac{\partial Loss}{\partial \omega_{li}}$?

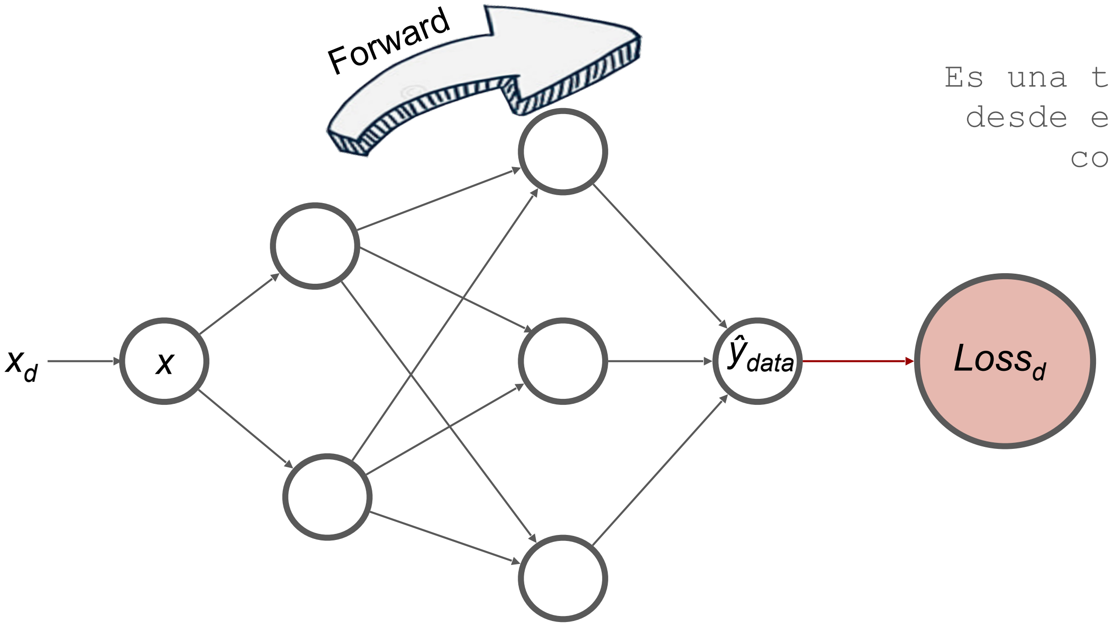


Automatic differentiation

Es una técnica computacional que existe desde el siglo pasado pero que explotó con el aumento de la capacidad de cómputo de las últimas décadas

Crea un grafo computacional

¿Qué es una Red Neuronal?



¿Cómo calculo $\dfrac{\partial Loss}{\partial \omega_{li}}$ **?**

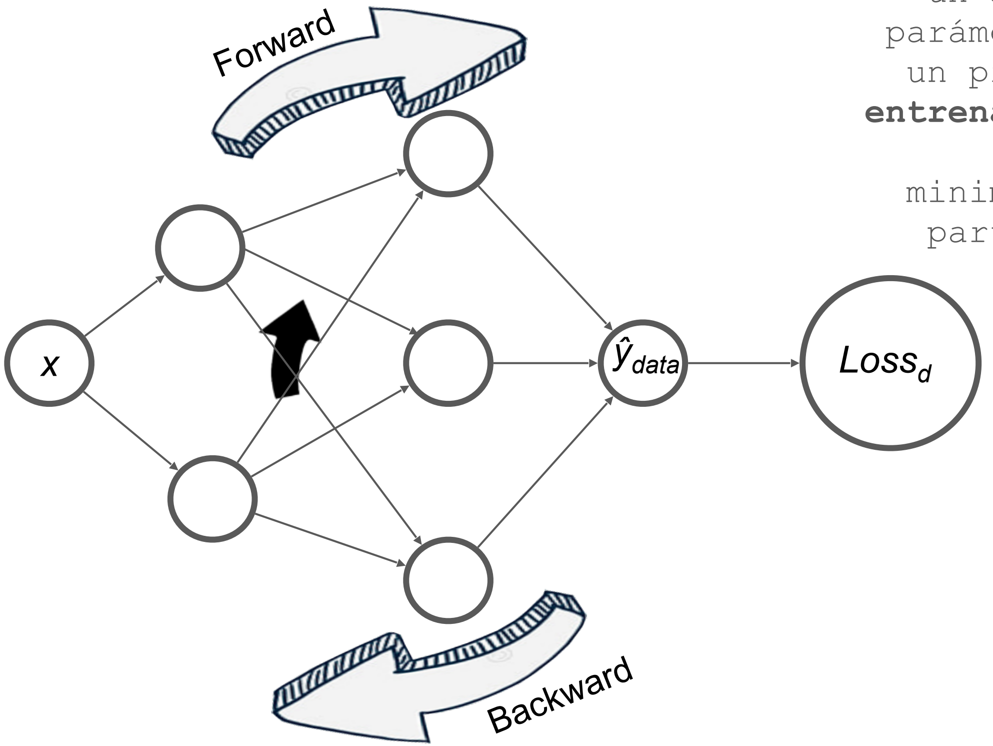
Automatic differentiation

Es una técnica computacional que existe desde el siglo pasado pero que explotó con el aumento de la capacidad de cómputo de las últimas décadas

Crea un grafo computacional

Aplica la regla de la cadena

¿Qué es una Red Neuronal?



Resumen de Red Neuronal

Se puede decir que una red neuronal es un **aproximador de funciones**, cuyos parámetros son ajustados a partir de un proceso iterativo (conocido como **entrenamiento**) que optimiza los **pesos** y los **sesgos** de cada neurona, minimizando una **función objetivo**, a partir de técnicas de **descenso por gradiente** para lo que utiliza **Automatic differentiation.**

$$\arg\min_{w,b} Loss_d(\mathbf{x}, w, b)$$

¿Qué es una PINN?

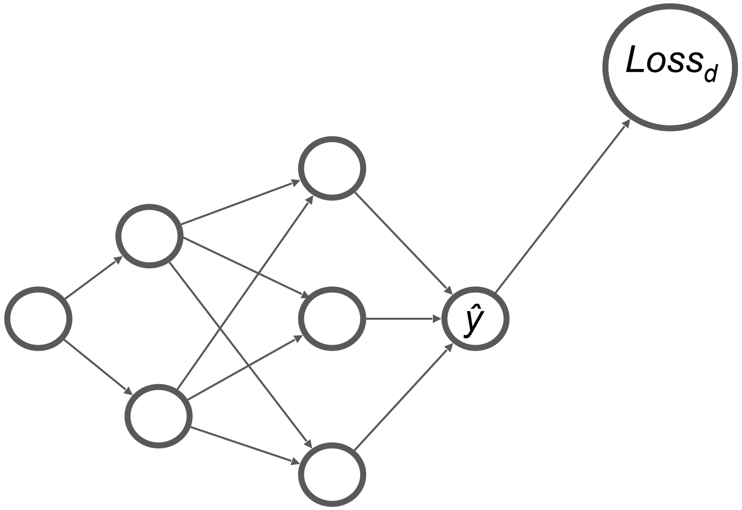
¿Qué es una PINN?

Características principales

Una PINN se obtiene cuando se dota a la **NN** de información proveniente de la **física del problema**, con el objetivo de:

- Ayudar al proceso de optimización
- Aprender con menos datos
- Mejorar la generalización
- Aprender de datos ruidosos.

¿Qué es una PINN?

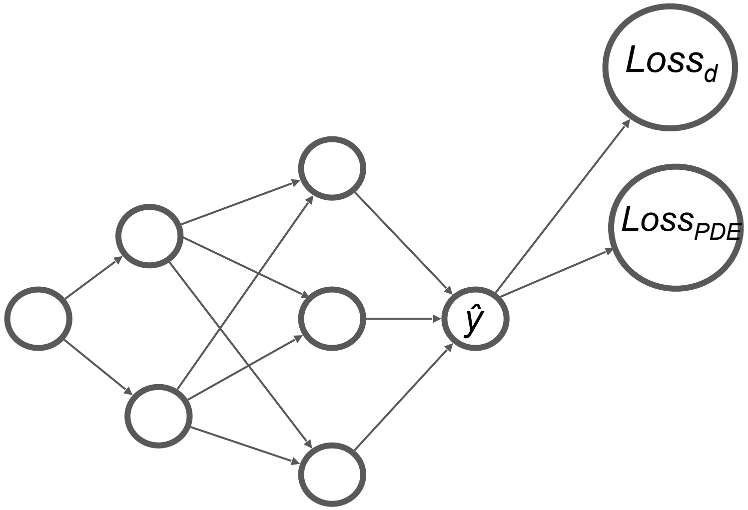


Características principales

Una PINN se obtiene cuando se dota a la **NN** de información proveniente de la **física del problema**, con el objetivo de:

- Ayudar al proceso de optimización
- Aprender con menos datos
- Mejorar la generalización
- Aprender de datos ruidosos.

¿Qué es una PINN?



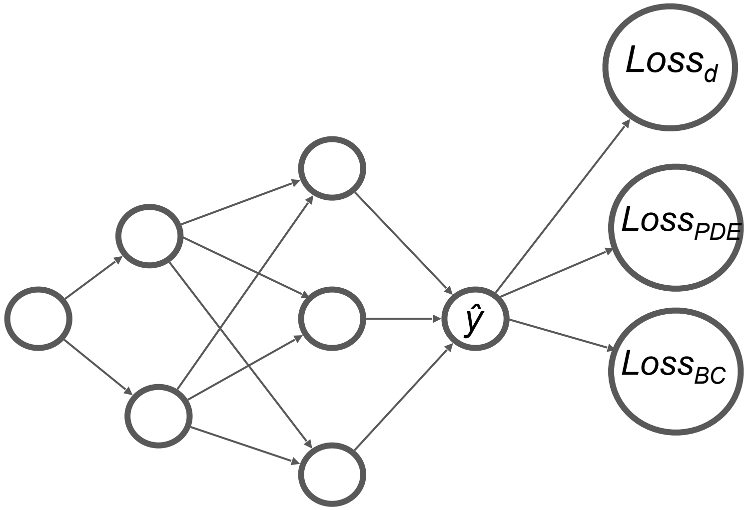
Características principales

Una PINN se obtiene cuando se dota a la **NN** de información proveniente de la **física del problema**, con el objetivo de:

- Ayudar al proceso de optimización
- Aprender con menos datos
- Mejorar la generalización
- Aprender de datos ruidosos.

$Loss_{PDE}$: Ecuación diferencial en derivadas parciales que gobierna la física del problema

¿Qué es una PINN?



Características principales

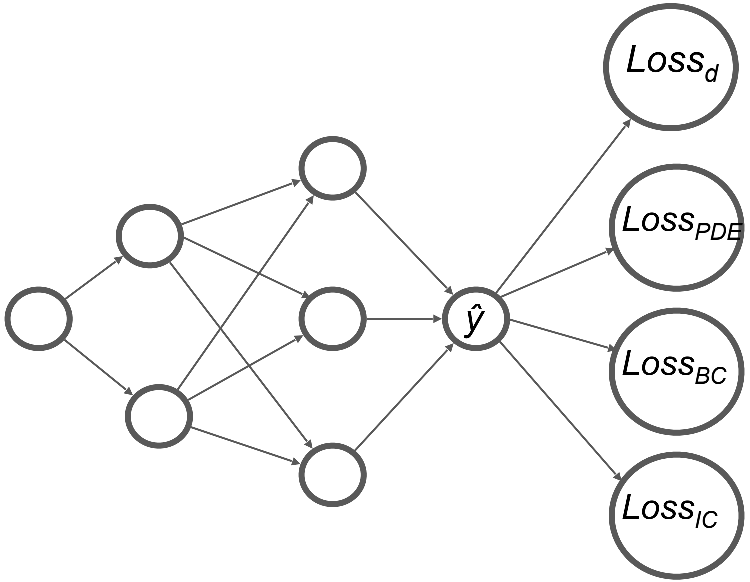
Una PINN se obtiene cuando se dota a la **NN** de información proveniente de la **física del problema**, con el objetivo de:

- Ayudar al proceso de optimización
- Aprender con menos datos
- Mejorar la generalización
- Aprender de datos ruidosos.

$Loss_{PDE}$: Ecuación diferencial en derivadas parciales que gobierna la física del problema

$Loss_{BC}$: Ecuación que fuerza a cumplir condiciones de borde.

¿Qué es una PINN?



Características principales

Una PINN se obtiene cuando se dota a la **NN** de información proveniente de la **física del problema**, con el objetivo de:

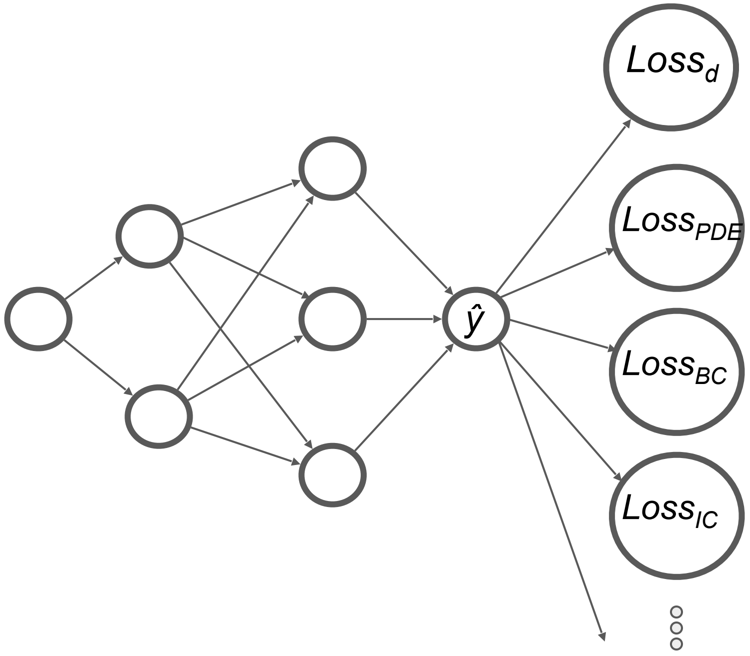
- Ayudar al proceso de optimización
- Aprender con menos datos
- Mejorar la generalización
- Aprender de datos ruidosos.

$Loss_{PDE}$: Ecuación diferencial en derivadas parciales que gobierna la física del problema

$Loss_{BC}$: Ecuación que fuerza a cumplir condiciones de borde.

$Loss_{IC}$: Ecuación que fuerza a cumplir con condiciones iniciales.

¿Qué es una PINN?



Características principales

Una PINN se obtiene cuando se dota a la **NN** de información proveniente de la **física del problema**, con el objetivo de:

- Ayudar al proceso de optimización
- Aprender con menos datos
- Mejorar la generalización
- Aprender de datos ruidosos.

$Loss_{PDE}$: Ecuación diferencial en derivadas parciales que gobierna la física del problema

$Loss_{BC}$: Ecuación que fuerza a cumplir condiciones de borde.

$Loss_{IC}$: Ecuación que fuerza a cumplir con condiciones iniciales.

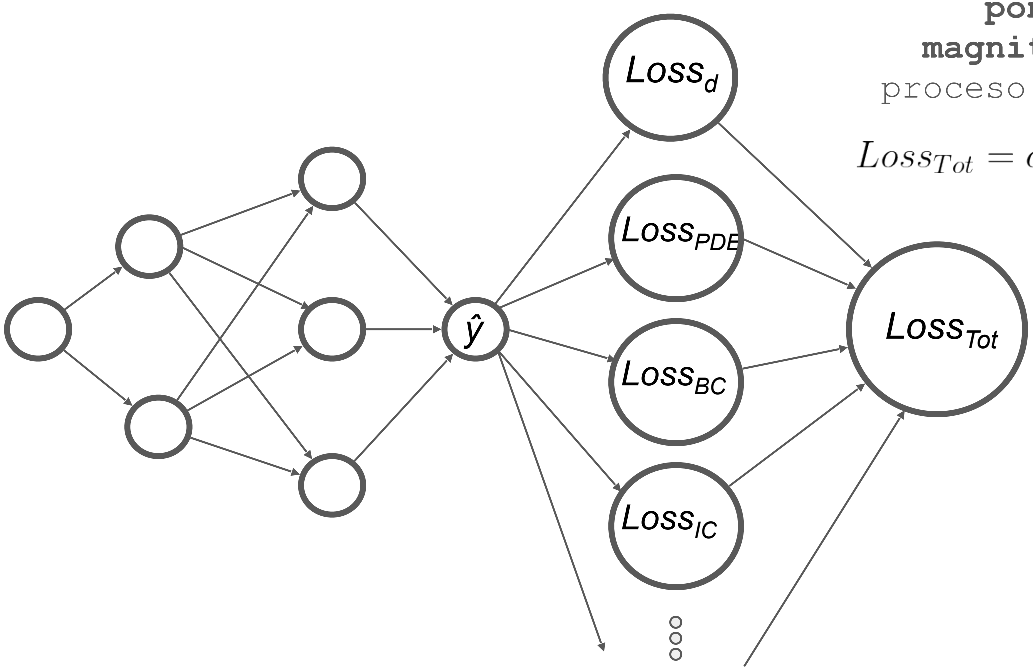
Otras Losses: Cualquier información que se conozca en alguna parte del problema puede colaborar con el proceso de optimización.

¿Qué es una PINN?

Características principales

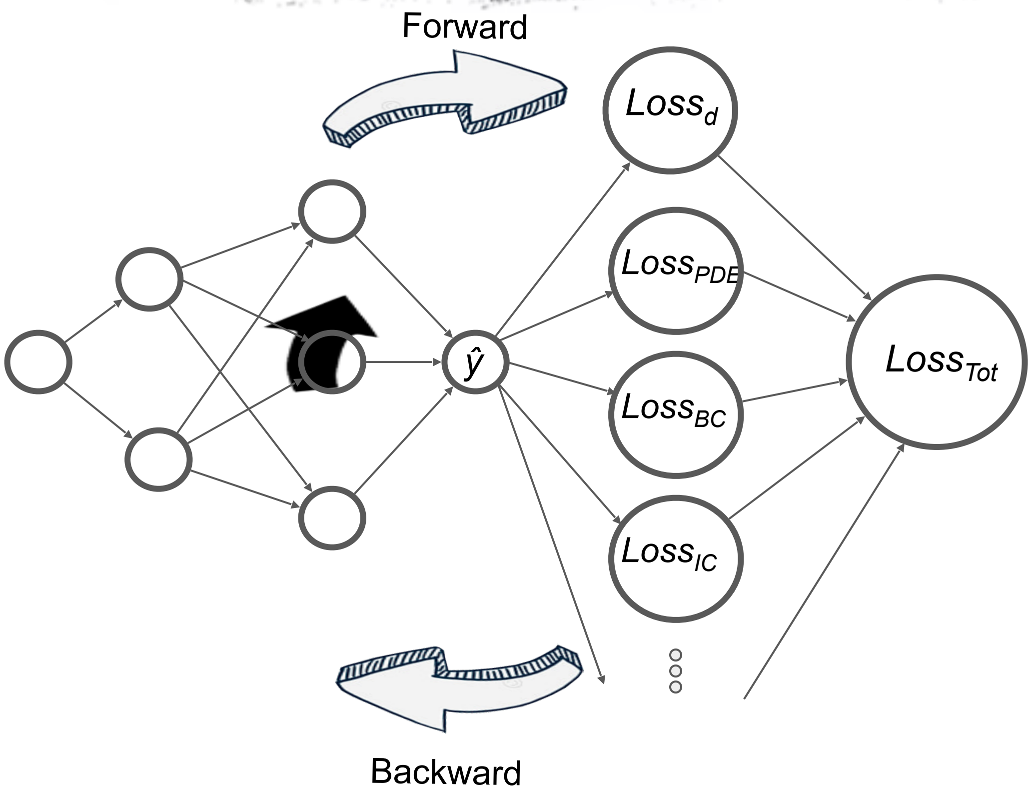
La Loss total se suele calcular como la suma **ponderada** de cada término, para **escalar las magnitudes** y **controlar la "importancia"** que el proceso de optimización le da a cada componente.

$$Loss_{Tot} = \alpha_d Loss_d + \alpha_{BC} Loss_{PDE} + \alpha_{BC} Loss_{BC} + \alpha_{IC} Loss_{IC} + \ldots$$



¿Qué es una PINN?

Entrenamiento



¿Qué es una PINN?

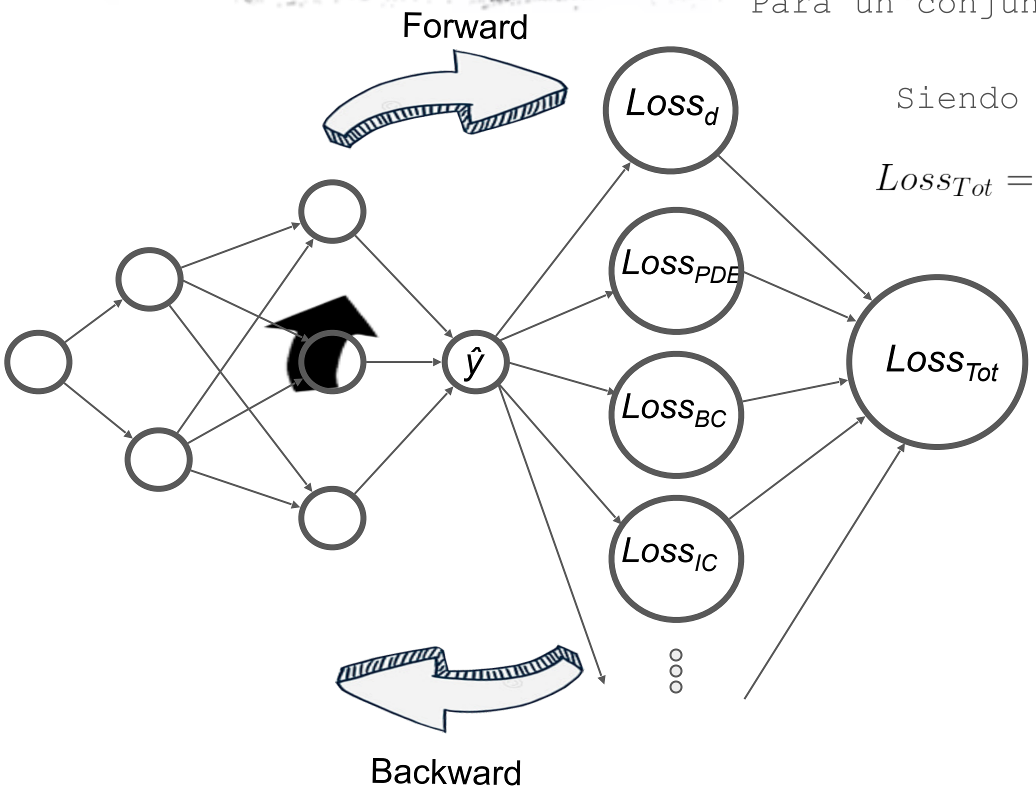
Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x}_d = \{\mathbf{x}_i \in \mathbf{R}^n : \mathbf{i} = 1, .. \mathbf{N}_d]$

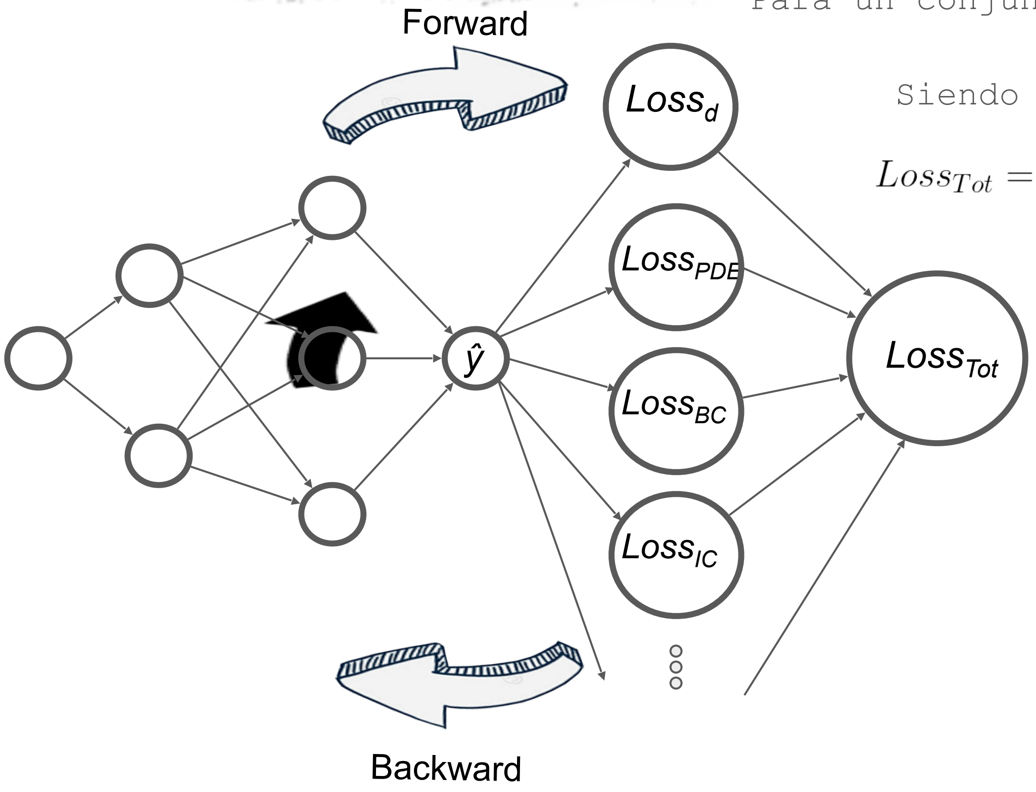
Se conoce $\mathbf{y}_d = \{\mathbf{y}_i \in \mathbf{R}^m : \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)\}$

Siendo $\mathbf{f}(\mathbf{x})$ la función a estimar con $\mathbf{f}(\mathbf{x}) : \mathbf{R}^n \rightarrow \mathbf{R}^m$

$$Loss_{Tot} = \alpha_d Loss_d + \alpha_{BC} Loss_{PDE} + \alpha_{BC} Loss_{BC} + \alpha_{IC} Loss_{IC} + \ldots$$



¿Qué es una PINN?



Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x}_d = \{\mathbf{x}_i \in \mathbb{R}^n : i = 1, \ldots N_d]$

Se conoce $\mathbf{y}_d = \{\mathbf{y}_i \in \mathbb{R}^m : \mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)\}$

Siendo $\mathbf{f}(\mathbf{x})$ la función a estimar con $\mathbf{f}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$

$$Loss_{Tot} = \alpha_d Loss_d + \alpha_{BC} Loss_{PDE} + \alpha_{BC} Loss_{BC} + \alpha_{IC} Loss_{IC} + \ldots$$

Resuelve iterativamente

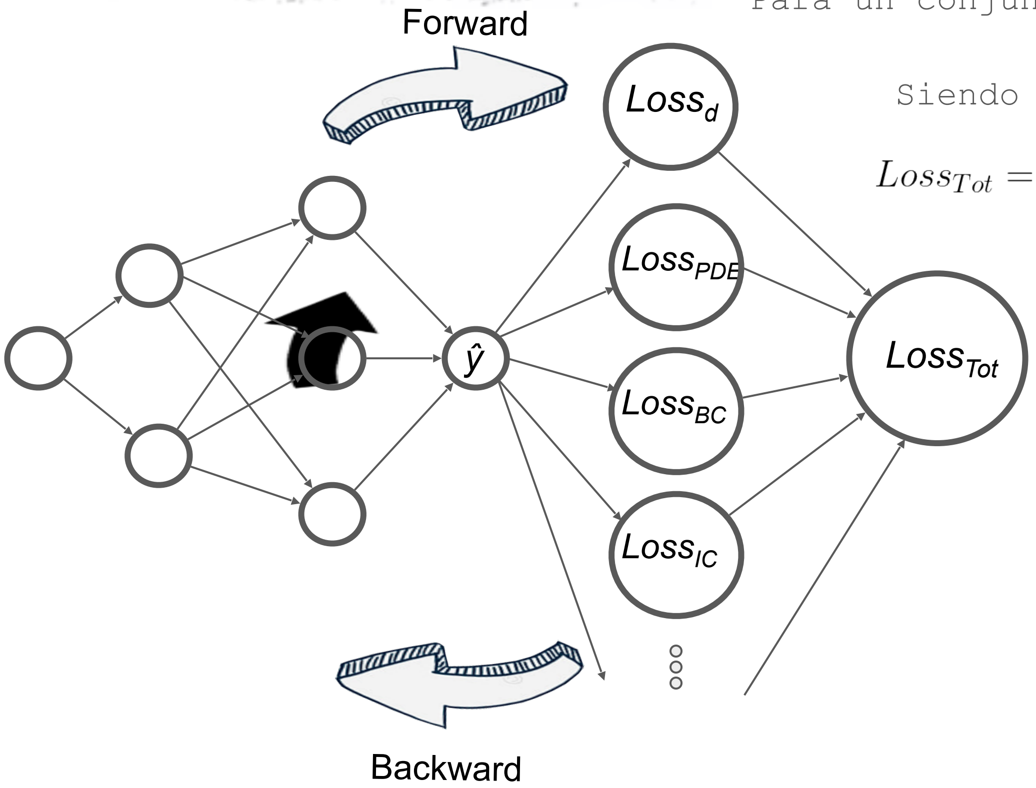
$$\arg\min_{w_{li}, b_l} \; Loss_{Tot}(\mathbf{x}, w_{li}, b_l)$$

Actualizando parámetros

$$\omega_{li}^{t+1} = \omega_{li}^{t} - \eta \frac{\partial Loss_{Tot}}{\partial \omega_{li}}$$

$$b_l^{t+1} = b_l^{t} - \eta \frac{\partial Loss_{Tot}}{\partial b_l}$$

¿Qué es una PINN?



Entrenamiento

Para un conjunto de N_d datos → $\mathbf{x_d} = \{\mathbf{x_i} \in \mathbf{R^n}: \mathbf{i} = 1, \ldots \mathbf{N_d}]$

Se conoce $\mathbf{y_d} = \{\mathbf{y_i} \in \mathbf{R^m}: \mathbf{y_i} = \mathbf{f(x_i)}\}$

Siendo $\mathbf{f(x)}$ la función a estimar con $\mathbf{f(x)}: \mathbf{R^n} \to \mathbf{R^m}$

$$Loss_{Tot} = \alpha_d Loss_d + \alpha_{BC} Loss_{PDE} + \alpha_{BC} Loss_{BC} + \alpha_{IC} Loss_{IC} + \ldots$$

Resuelve iterativamente

$$\arg\min_{w_{li}, b_l} Loss_{Tot}(\mathbf{x}, w_{li}, b_l)$$

θ

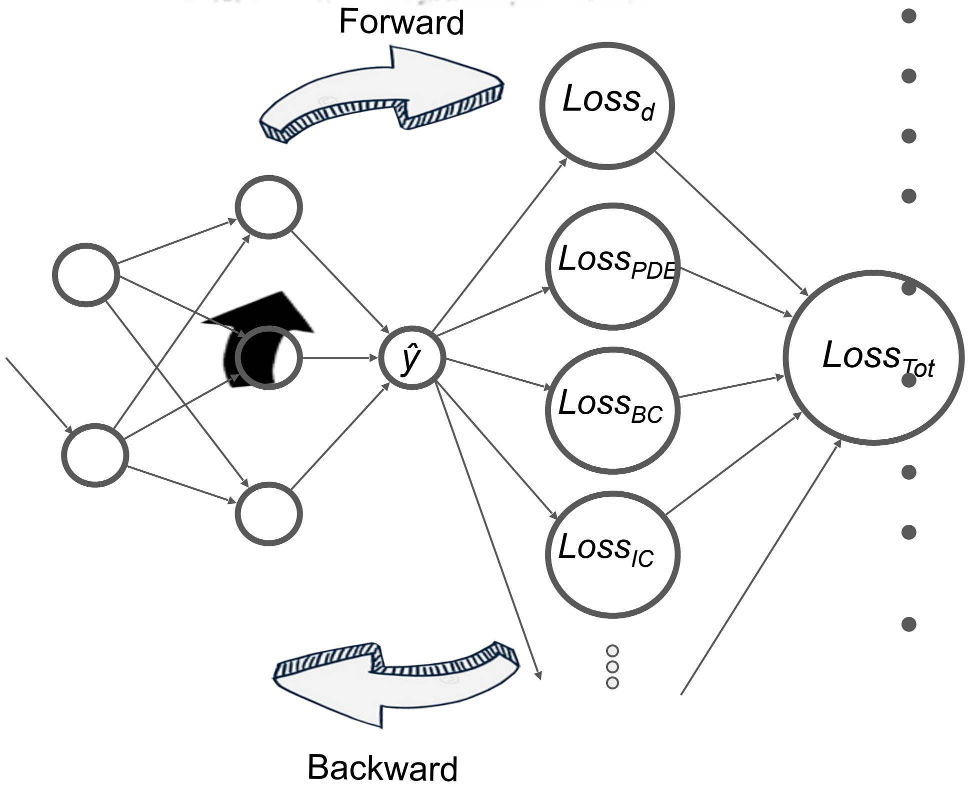
Actualizando parámetros

$$\omega_{li}^{t+1} = \omega_{li}^{t} \quad - \eta \frac{\partial Loss_{Tot}}{\partial \omega_{li}}$$

$$b_l^{t+1} = b_l^{t} \quad - \eta \frac{\partial Loss_{Tot}}{\partial b_l}$$

Otros parámetros

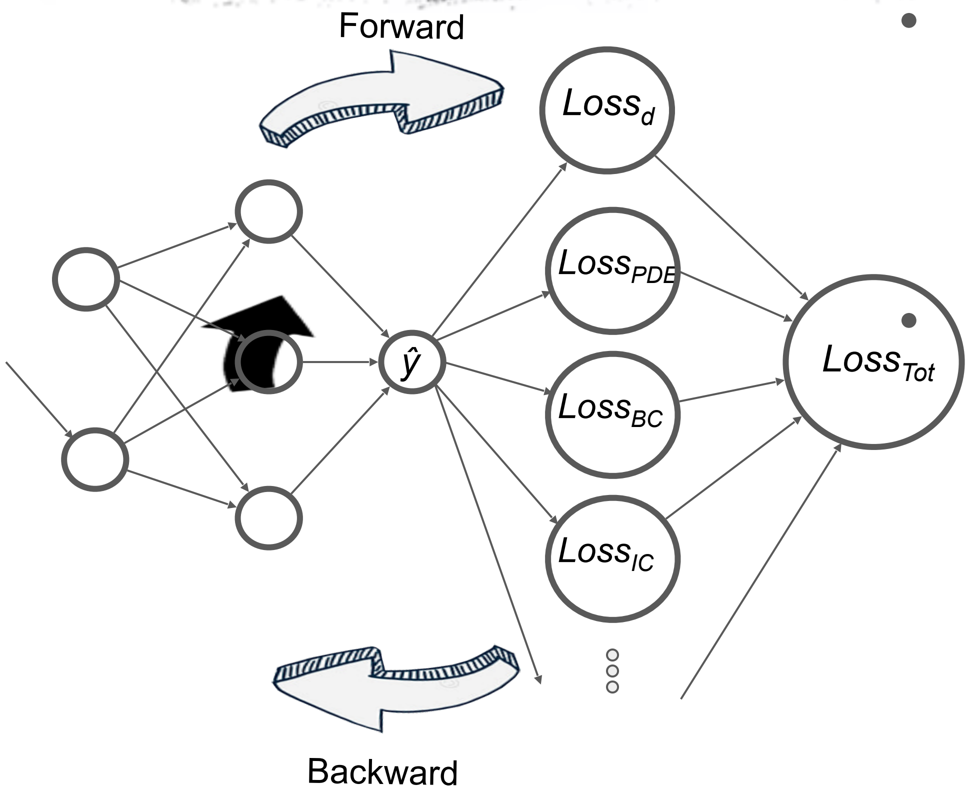
¿Qué es una PINN?



Ventajas

- Incorporación de información física
- Aprendizaje de datos ruidosos y escasos
- Alta generalización
- Flexibilidad en la formulación de problemas
- Menos (o nula) dependencia de mallas o discretización
- Capacidad para lidiar con EDPs no lineales
- Escalabilidad
- Potencial para resolver problemas inversos
- Integrabilidad con otros métodos

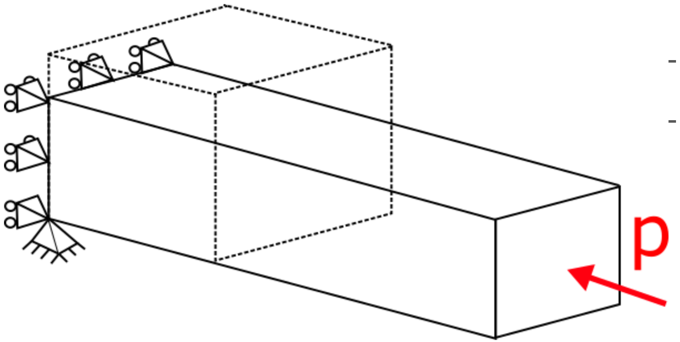
¿Qué es una PINN?



Algunas desventajas

- Su rendimiento depende de la elección adecuada de la **arquitectura de red,** la **configuración de hiperparámetros** y la cantidad de datos de entrenamiento disponibles.
- También pueden requerir **esfuerzo** y **experiencia** en su implementación y sintonización para obtener resultados óptimos en problemas específicos.

Aplicación a mecánica del sólido

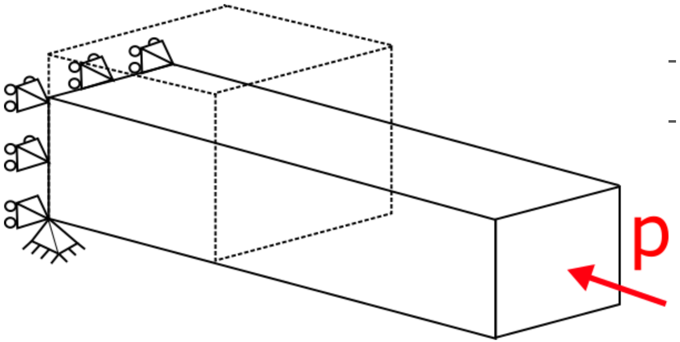


Objetivo

Resolver el **problema inverso** de identificación de propiedades mecánicas (K, G) a partir de conocer:

- Algunos datos de desplazamiento.
- La fuerza ejercida.

Aplicación a mecánica del sólido



Objetivo

Resolver el **problema inverso** de identificación de propiedades mecánicas (K, G) a partir de conocer:

- Algunos datos de desplazamiento.
- La fuerza ejercida.

Física del problema

$$\nabla . P + \mathbf{b} = 0$$ Ecuación de equilibrio

$$\mathbf{P} = \frac{\partial \Psi(\mathbf{F})}{\partial \mathbf{F}}$$ Ecuación constitutiva

$$\Psi(\mathbf{C}) = \frac{G}{2}(tr(\mathbf{C})-3) - G ln(J) + \left(\frac{K}{2} - \frac{G}{3}\right) ln(J)^2$$ Densidad de energía de deformación

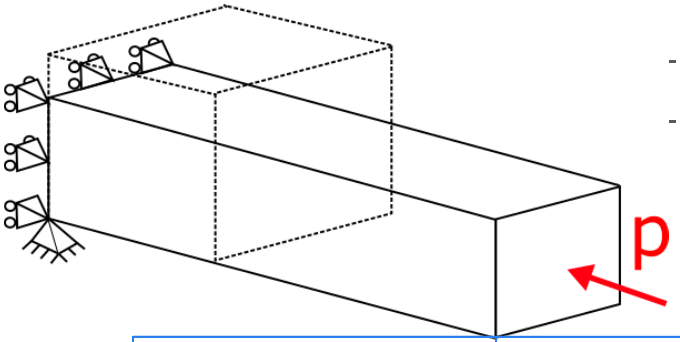
$$F = \nabla_m \chi \quad C = F^T F \quad J = \sqrt{|C|}$$ Relaciones cinemáticas

Aplicación a mecánica del sólido

Objetivo

Resolver el **problema inverso** de identificación de propiedades mecánicas (K, G) a partir de conocer:

- Algunos datos de desplazamiento.
- La fuerza ejercida.



Física del problema

$$\nabla . P + \mathbf{b} = 0$$ Ecuación de equilibrio

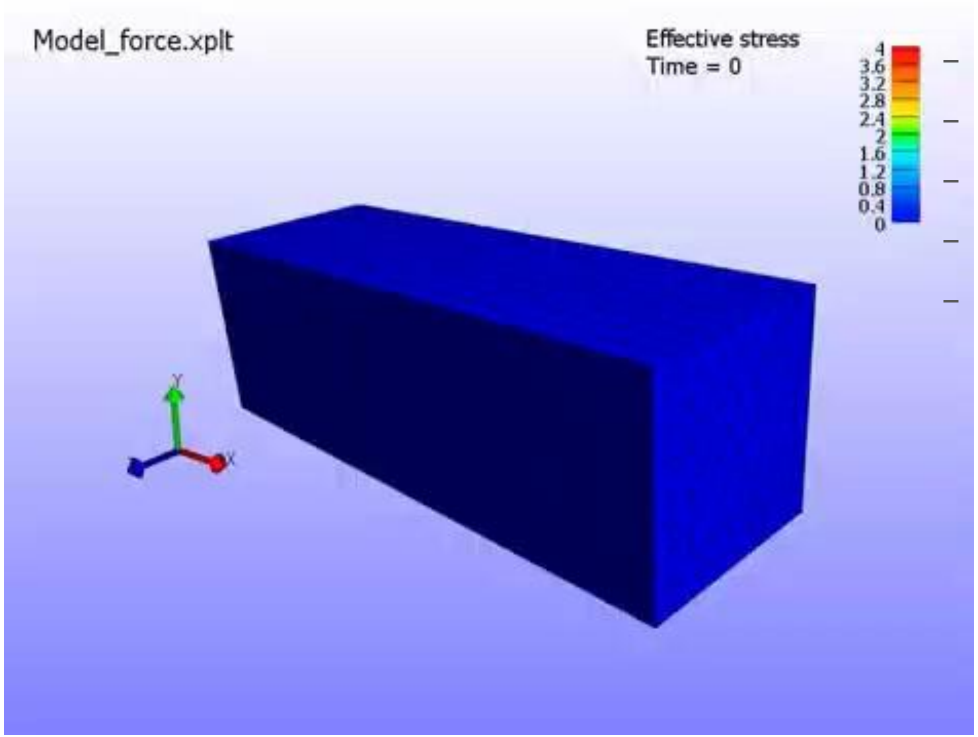
$$\mathbf{P} = \frac{\partial \Psi(\mathbf{F})}{\partial \mathbf{F}}$$ Ecuación constitutiva

Parámetros a identificar

$$\Psi(\mathbf{C}) = \frac{\boxed{G}}{2}(tr(\mathbf{C}) - 3) - \boxed{G} ln(J) + \left(\frac{\boxed{K}}{2} - \frac{\boxed{G}}{3}\right) ln(J)^2$$ Densidad de energía de deformación

$$F = \nabla_m \chi \quad C = F^T F \quad J = \sqrt{|C|}$$ Relaciones cinemáticas

Aplicación a mecánica del sólido



Creación de datos digitales

Utilizando un software de Elementos Finitos se resuelve el caso con:

Geometría (axhxl): 1x1x3mm

3000 elementos Hexaedros

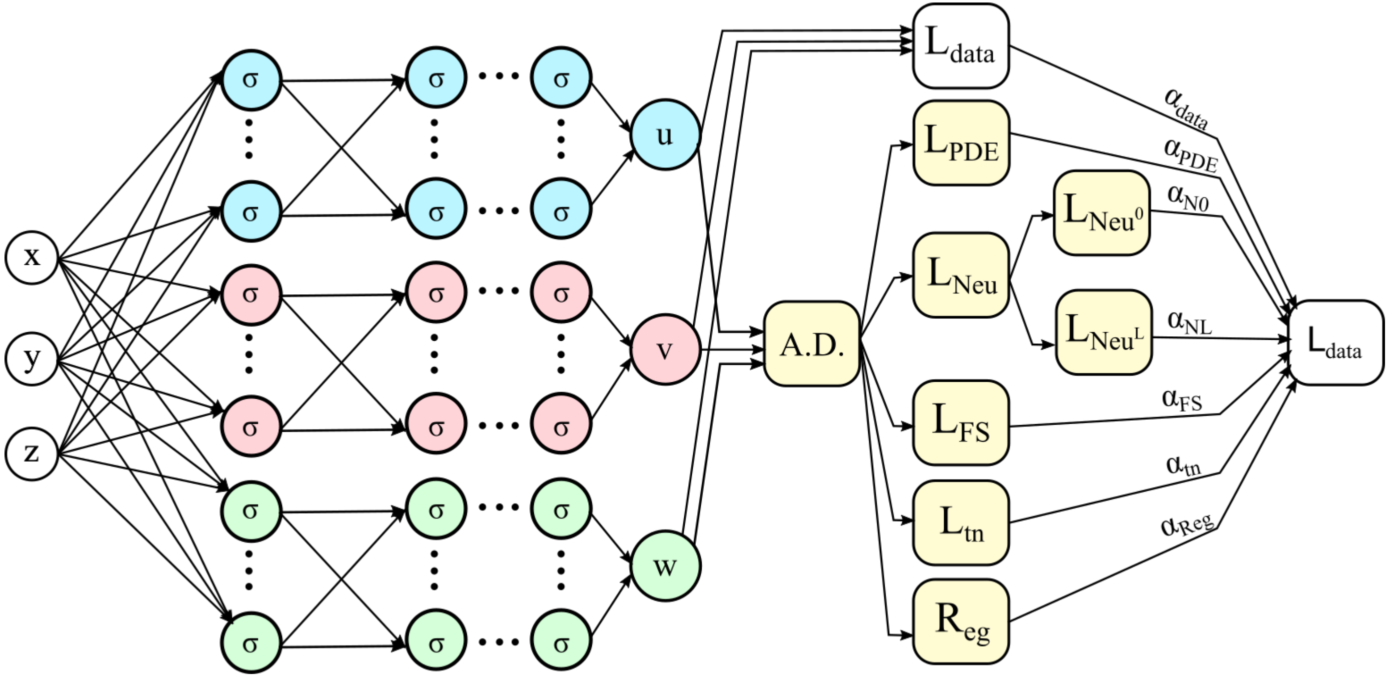
E = 4 MPa

v = 0.3

P = -5 N

Aplicación a mecánica del sólido

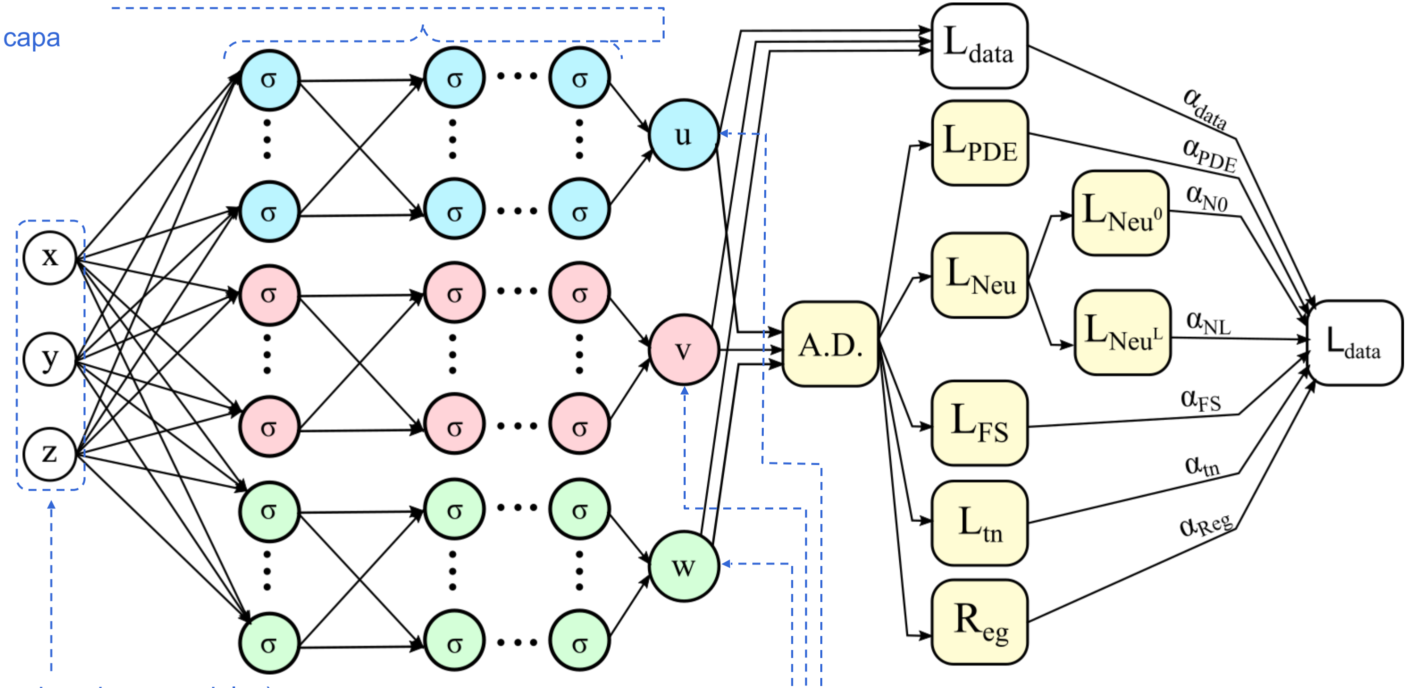
PINN propuesta



Aplicación a mecánica del sólido

PINN propuesta

3 Redes independientes
4 capas ocultas
50 neuronas por capa



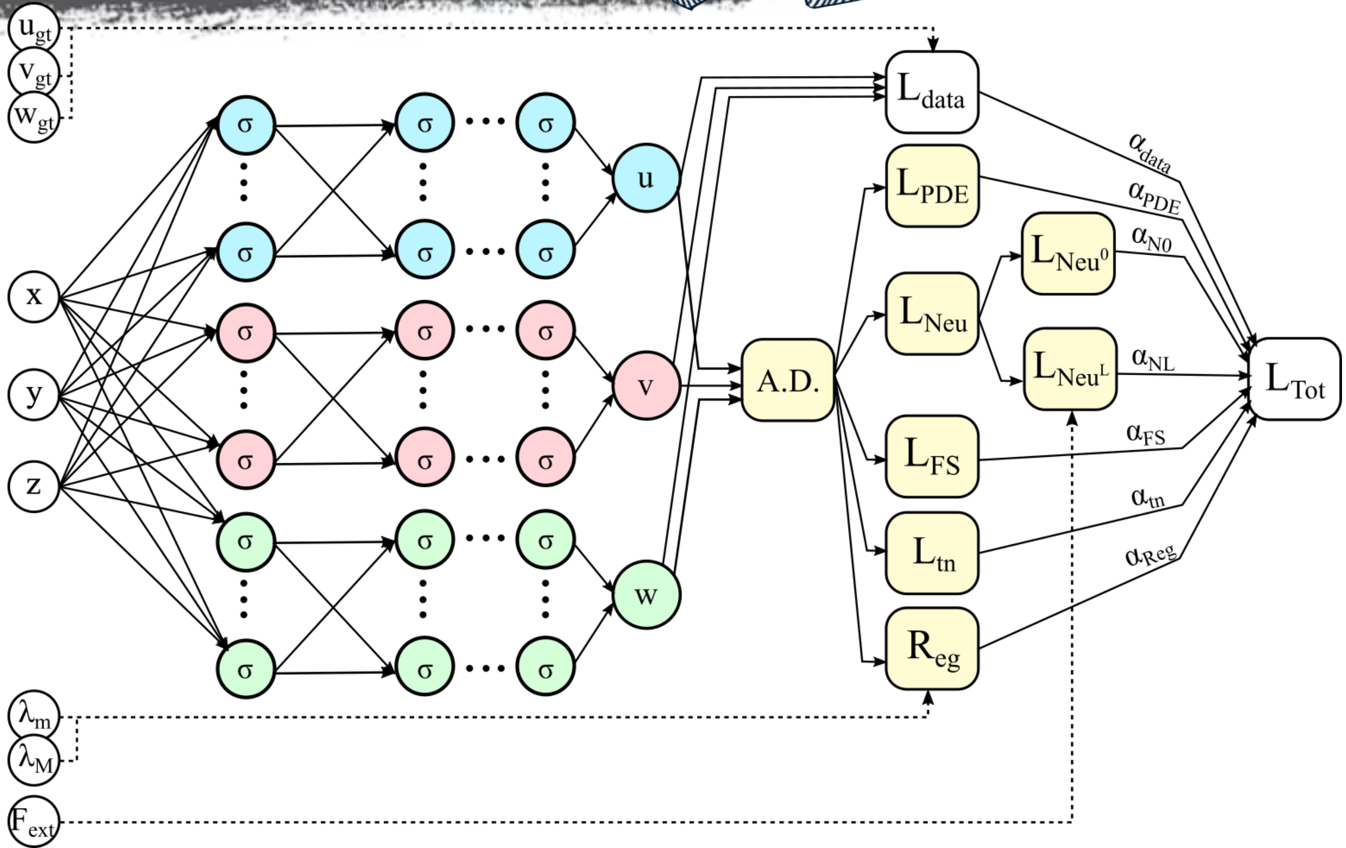
3 inputs (3D coordenadas espaciales)

1 output por red (desplazamientos)

Aplicación a mecánica del sólido

Forward

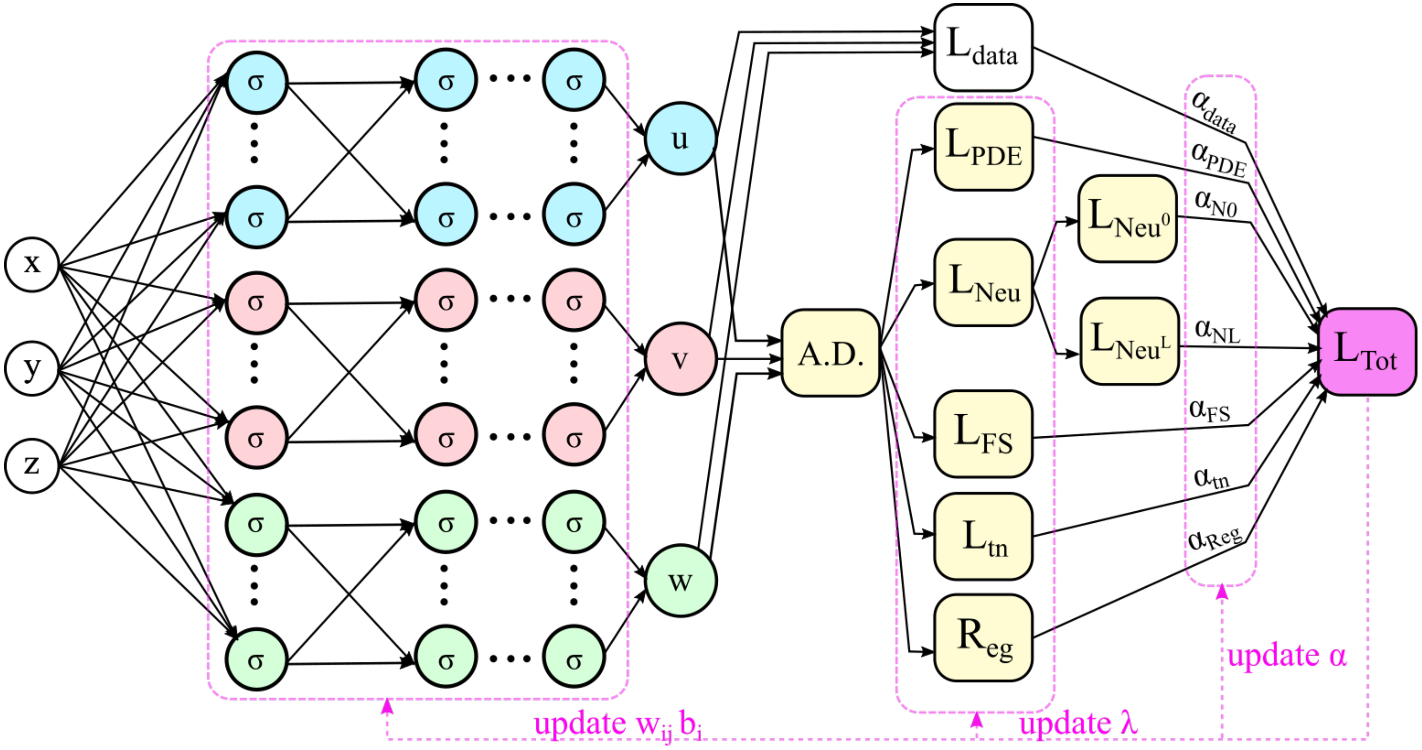
PINN propuesta



Aplicación a mecánica del sólido

Backward

PINN propuesta



Aplicación a mecánica del sólido

PINN propuesta

$$L_{Tot} = \begin{cases} \alpha_{data} L_{data} + & \longrightarrow \alpha_{dx} L_{dx} + \alpha_{dy} L_{dy} + \alpha_{dz} L_{dz} \\ \alpha_{PDE} L_{PDE} + & \longrightarrow \text{Minimiza el residuo de la PDE} \\ \alpha_{N0} L_{Neu^0} + & \longrightarrow \text{Aplica la condición de borde libre} \\ \alpha_{NL} L_{Neu^L} + & \longrightarrow \text{Aplica la condición de borde cargada} \\ \alpha_{FS} L_{FS} + & \longrightarrow \text{Aplica deslizamiento sin fricción en el contacto} \\ \alpha_{t_n} L_{t_n} + & \longrightarrow \text{Aplica condición de presión de contacto negativa} \\ \alpha_{Reg} R_{eg} & \longrightarrow \text{Restringe el rango posible de valores del parámetro material} \end{cases}$$

Aplicación a mecánica del sólido

PINN propuesta

$$L_{Tot} = \begin{cases} \alpha_{data} L_{data} + & \longrightarrow \alpha_{dx} L_{dx} + \alpha_{dy} L_{dy} + \alpha_{dz} L_{dz} \\ \alpha_{PDE} L_{PDE} + & \longrightarrow \text{Minimiza el residuo de la PDE} \\ \alpha_{N0} L_{Neu^0} + & \longrightarrow \text{Aplica la condición de borde libre} \\ \alpha_{NL} L_{Neu^L} + & \longrightarrow \text{Aplica la condición de borde cargada} \\ \alpha_{FS} L_{FS} + & \longrightarrow \text{Aplica deslizamiento sin fricción en el contacto} \\ \alpha_{t_n} L_{t_n} + & \longrightarrow \text{Aplica condición de presión de contacto negativa} \\ \alpha_{Reg} R_{eg} & \longrightarrow \text{Restringe el rango posible de valores del parámetro material} \end{cases}$$

Aplicación a mecánica del sólido

PINN propuesta

$$\alpha_{data} L_{data} = \alpha_{dx} L_{dx} + \alpha_{dy} L_{dy} + \alpha_{dz} L_{dz}$$

$$L_{dx} = \frac{1}{N_d} \sum_{i=1}^{N_d} ||u_{NN_i} - u_{gt_i}||^2$$

$$L_{dy} = \frac{1}{N_d} \sum_{i=1}^{N_d} ||v_{NN_i} - v_{gt_i}||^2$$

$$L_{dz} = \frac{1}{N_d} \sum_{i=1}^{N_d} ||w_{NN_i} - w_{gt_i}||^2$$

Cada loss es calculada como el MSE de las correspondientes componentes del desplazamiento.

Como los nodos con desplazamiento nulo corresponden al grupo de N_d puntos en los que se conocen los datos, la condición de borde de Dirichlet queda impuesta a través de esta los. Esta forma de aplicar las condiciones de borde se conoce como "soft constrain"

Aplicación a mecánica del sólido

PINN propuesta

$$\alpha_{PDE} L_{PDE} \longrightarrow$$ Minimiza el residuo de la ley que gobierna el sistema (PDE)

$$L_{PDE} = \frac{1}{N_c} \sum_{i=1}^{N_c} ||Res(\mathbf{x}_i)||^2$$ Mean square error del residuo

$$Res(\mathbf{x}_i) = \nabla . P(\mathbf{x}_i) + b(\mathbf{x}_i)$$ Equilibrio mecánico (forma fuerte)

$$P = FS$$ Primer tensor de tensiones de Piola-Kirchhoff

$$F = \nabla_m \chi = I + \nabla_m \mathbf{u}$$ Tensor del gradiente de la deformación

$$S = 2\frac{\partial \psi}{\partial C}$$ Tensor de tensiones de Cosserat

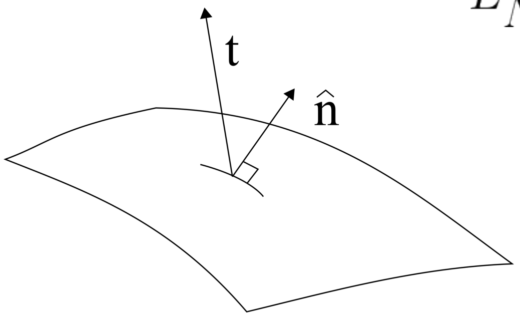
$$C = F^T F$$ Tensor de deformaciones de Cauchy-Green derecho

$$\psi(C) = \frac{G}{2}(tr(\mathbf{C}) - 3) - G ln(J) + \left(\frac{K}{2} - \frac{G}{3}\right) ln(J)^2$$ Función de densidad de energía de deformación (Neo-Hookean)

Aplicación a mecánica del sólido

PINN propuesta

$\alpha_{N0} L_{Neu^0}$ → Aplica la condición de borde sin carga



$$L_{Neu^0} = \frac{1}{N_{N0}} \sum_{i=1}^{N_{N0}} ||\mathbf{t}||^2$$

Mean square error de la norma de la tensión superficial

$$\mathbf{t} = \sigma.\hat{\mathbf{n}}$$

t: tensión sobre la superficie en el punto con normal **n**

$$\sigma = J^{-1} F S F^T$$

Tensor de tensiones de Cauchy

$$J = \sqrt{|C|}$$

Módulo de compresibilidad

Aplicación a mecánica del sólido

PINN propuesta

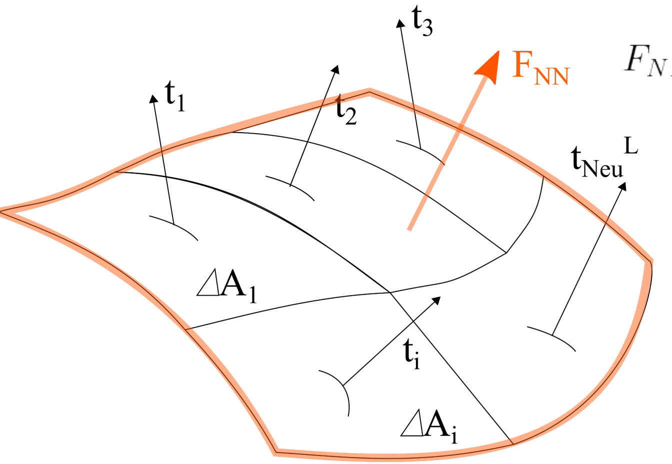
$\alpha_{NL} L_{Neu^L}$ → Aplica la condición de borde de carga

$$L_{Neu^L} = \left|\left| \frac{\mathbf{F}_{ext} - \mathbf{F}_{NN}}{||\mathbf{F}_{ext}||} \right|\right|^2$$

Equilibrio de fuerzas en la superficie

$$F_{NN} = \int_{\partial\Omega} \mathbf{t} dA \approx \sum_{i=1}^{N_{NL}} \mathbf{t}_i.\Delta A_i = \sum_{i=1}^{N_{NL}} \sigma_i.\hat{\mathbf{n}}_i.\Delta A_i$$

Aproximación de las fuerzas internas sobre la superficie cargada



Aplicación a mecánica del sólido

PINN propuesta

$\alpha_{FS} L_{FS}$ → Aplica el deslizamiento sin fricción en el contacto

$$L_{FS} = \frac{1}{N_{FS}} \sum_{i=1}^{N_{FS}} ||\mathbf{s}_i||^2$$

MSE de la tensión cortante **s** en cada nodo de N_{FS}

$$\sigma = (1/J) F S F^T \quad \rightarrow \quad \mathbf{t} = \sigma . \mathbf{n}$$

Tensor de tensiones de Cauchy → Tensión superficial

$$\mathbf{t}_a = \frac{\mathbf{t} \times \mathbf{n}}{||\mathbf{t} \times \mathbf{n}||} \qquad \mathbf{t}_b = \frac{\mathbf{n} \times \mathbf{t}_a}{||\mathbf{n} \times \mathbf{t}_a||}$$

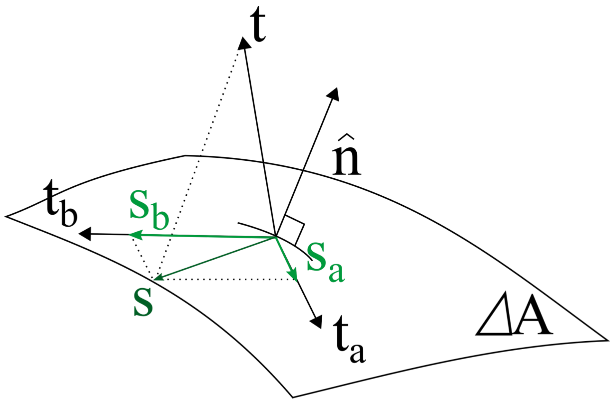
Vectores tangenciales a la superficie

$$\mathbf{s}_a = \mathbf{t} . \mathbf{t}_a \qquad \mathbf{s}_b = \mathbf{t} . \mathbf{t}_b$$

Vectores de tensión cortante

$$||\mathbf{s}|| = \sqrt{||\mathbf{s}_a||^2 + ||\mathbf{s}_b||^2}$$

Norma de la tensión cortante equivalente



Aplicación a mecánica del sólido

PINN propuesta

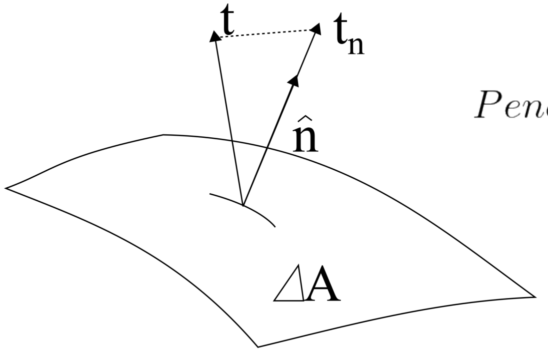
$\alpha_{t_n} L_{t_n}$ → Aplica la condición de negatividad de las tensiones en el contacto

$$L_{t_n} = \frac{1}{N_{t_n}} \sum_{i=1}^{N_{t_n}} ||Penalty(i)||^2$$

MSE de la función de penalización de la componente de tensión normal a la superficie en cada nodo N_{FS}

$$\mathbf{t}_n = \mathbf{t}.\mathbf{n}$$

Componente normal de la tensión superficial



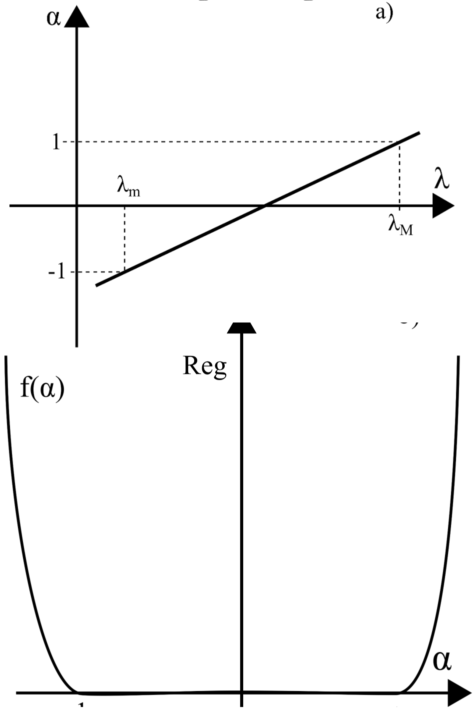
$$Penalty = \frac{1}{2}[1 + sg(\mathbf{t}_n)]\mathbf{t}_n$$

Función de penalización de la componente normal de la tensión superficial

Aplicación a mecánica del sólido

PINN propuesta

$\alpha_{Reg} L_{Reg}$ → Restinge el rango en el que puede variar el parámetro a ser identificado



Cconsiderando que λ es el parámetro a ser identificado
Se setean los valores mínimo y máximo que puede valer λ por el conocimiento del material aproximado → [λ_m , λ_M]
Luego α parametriza el rango de λ entre[-1 , 1]

$$\alpha = 2\frac{\lambda_{NN} - \lambda_m}{\lambda_M - \lambda_m} - 1$$

La regularización penaliza si la red predice valores de α mayores a 1 o menores a -1, segun:

$$Reg = \begin{cases} Ampl(min - \alpha)^2 & \text{si:}\alpha < \text{min} \\ 0.0 & \text{si:}\alpha \in \text{[-1;1]} \\ Ampl(\alpha - max)^2 & \text{si:}\alpha > \text{max} \end{cases}$$

Por lo tanto penaliza valores de λ mayores que λ_M o menores que λ_m

Aplicación a mecánica del sólido

PINN propuesta

$$L_{Tot} = \begin{cases} \boxed{\alpha_{data}} L_{data} + \\ \alpha_{PDE} L_{PDE} + \\ \alpha_{N0} L_{Neu^0} + \\ \alpha_{NL} L_{Neu^L} + \\ \alpha_{FS} L_{FS} + \\ \alpha_{KKT} L_{KKT} + \\ \alpha_{Reg} L_{Reg} \end{cases}$$

Existen numerosas técnicas para seleccionar o calcular los pesos de las losses, aquí tenemos implementado 4:

1. **Coefficient of Variance** (Ajusta los pesos según la varianza acumulada de las losses) **No buenos resultados**

2. **Adaptive Weights** (The network aprende los pesos de la misma manera que con los parámetros de la red) **Buenos resultados en algunos casos**

3. **Residual-Based attention** (Ajusta los èsps de acuerdo al error correspondiente en cada loss relativo al maximo) **Buenos resultados en algunos casos**

4. **Self-Adaptative Residual Minimization** (Maximiza los pesos de acuerdo al gradiente de la loss con momentum activado) **Los Mejores resultados hasta ahora**

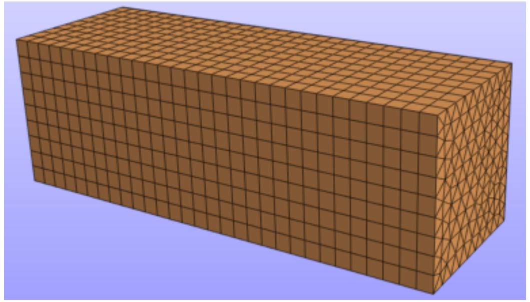
Resultados de caso

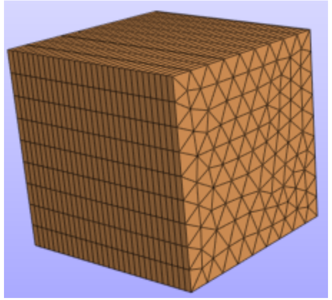
Datos digitales

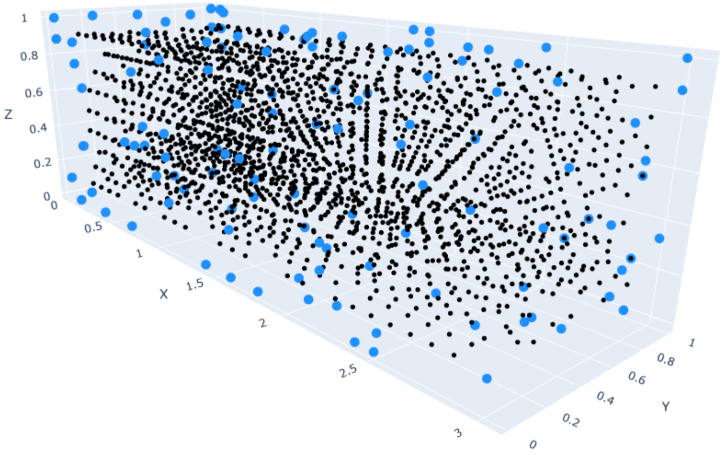
Resultados de caso

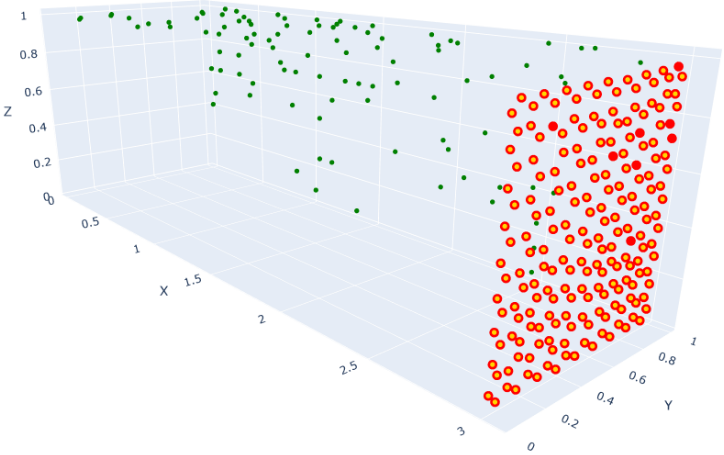
Datos digitales

Lr = 1e-1
L-BGFS optimizer
120 data points only on surfaces(25x4 + 10x2)
2000 collocation points
100 points on no-loaded surfaces
190 points on contact surfaces
198 points on loaded surface





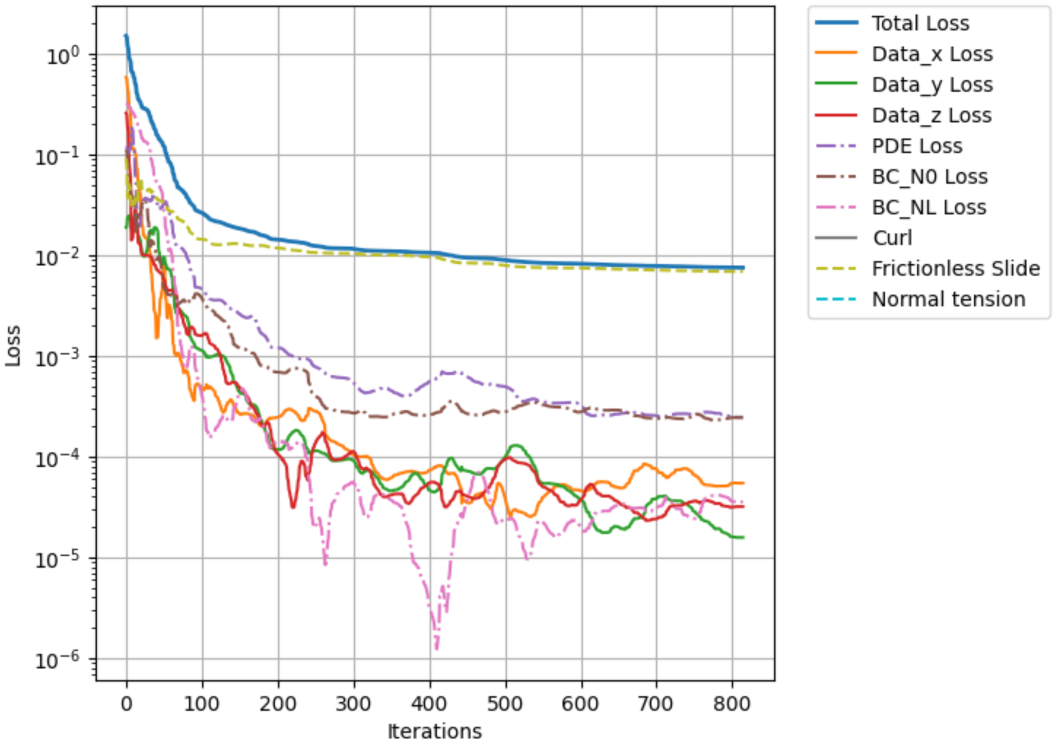


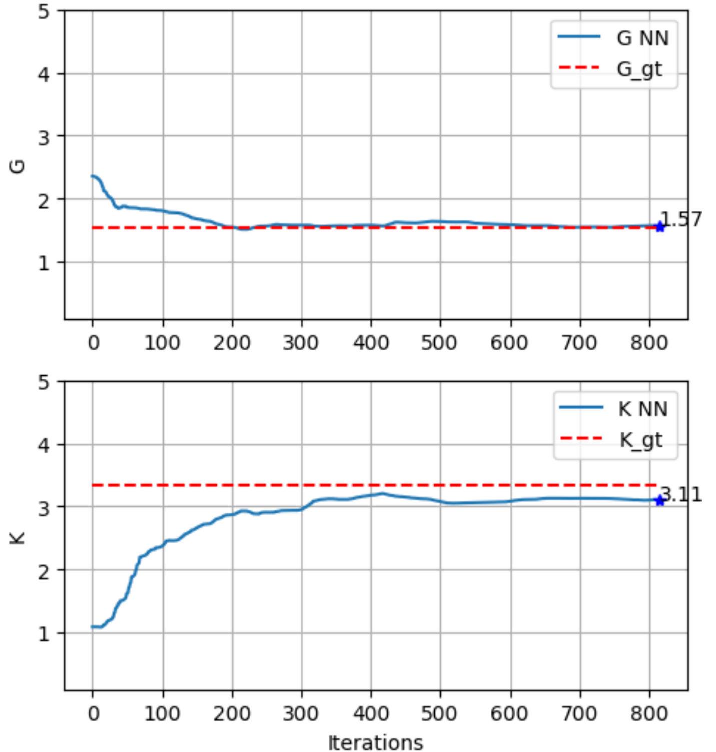


Resultados de caso

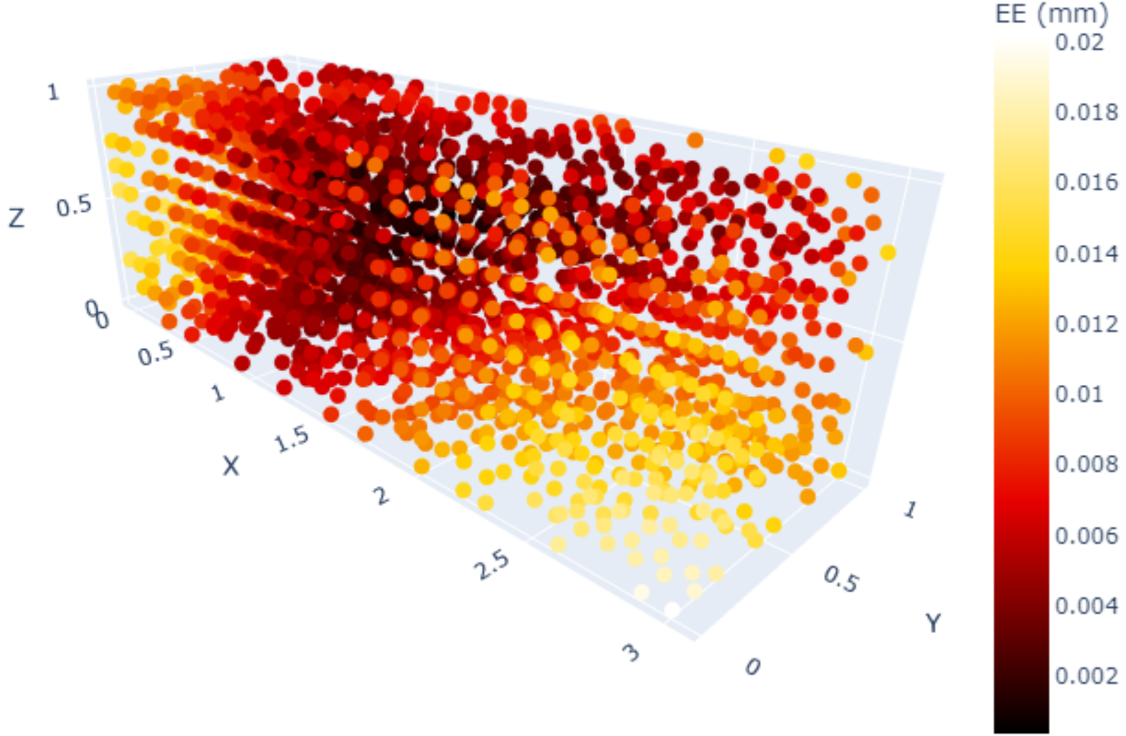
Identificación de K y G

G_{real} = 1.538 | G_{NN} = 1.570 (2.1% de error)
K_{real} = 3.333 | K_{NN} = 3.107 (6.8 % de error)





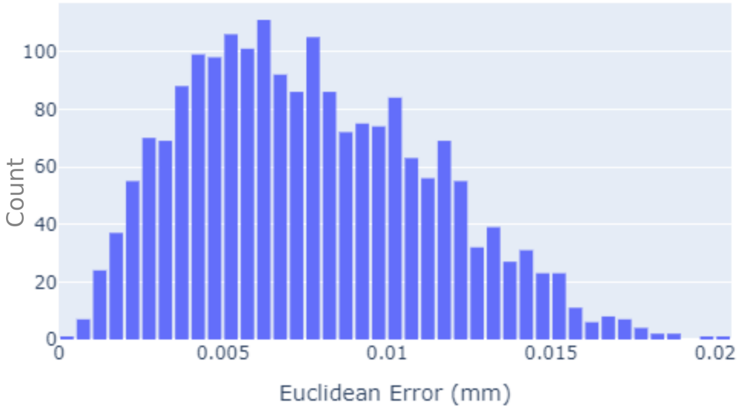
Resultados de caso



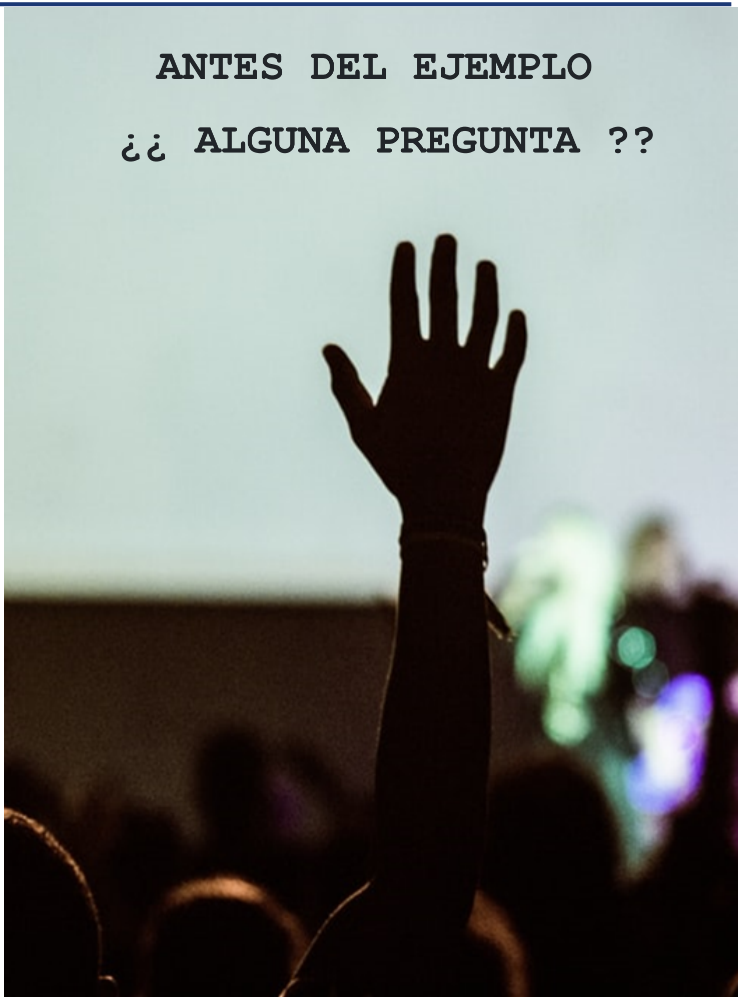
Otras métricas de error

Usando 2000 puntos de prueba:

MEE: 0.00765mm
MREE: 1.5 %
99.95 % de los ptos tienen EE<0.02mm
MAE: [0.00577 0.00187 0.00351] mm
MRAE: [0.337 0.726 1.364] %



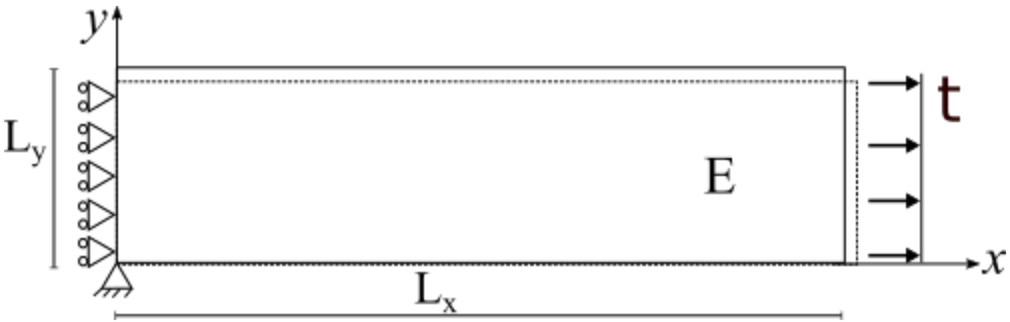
Ejemplo



Ejemplo

Características del ejemplo

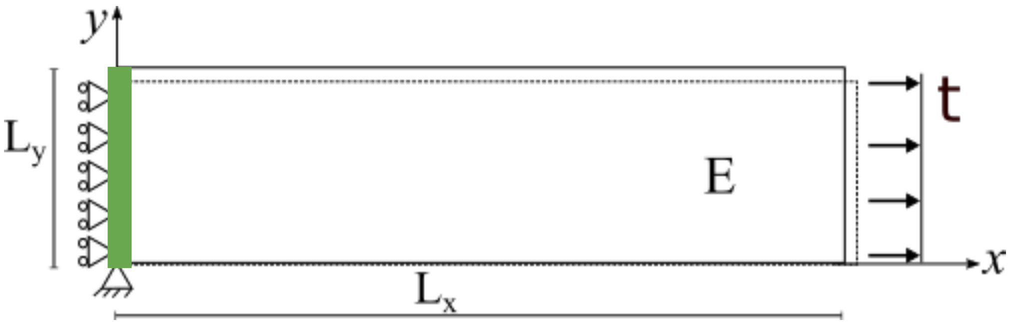
Placa 2D de Lx = 30mm, Ly = 10mm.
Elástico isotrópico com: E =10 MPa, v = 0.3



Ejemplo

Características del ejemplo

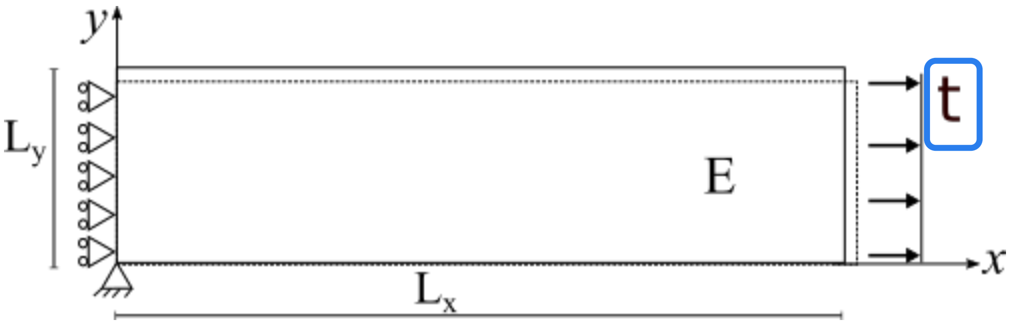
Placa 2D de Lx = 30mm, Ly = 10mm.
Elástico isotrópico com: E =10 MPa, v = 0.3
Extremo izquierdo fijo deslizante



Ejemplo

Características del ejemplo

Placa 2D de Lx = 30mm, Ly = 10mm.
Elástico isotrópico com: E =10 MPa, v = 0.3
Extremo izquierdo fijo deslizante
Fuerza en el extremo: T = 1N

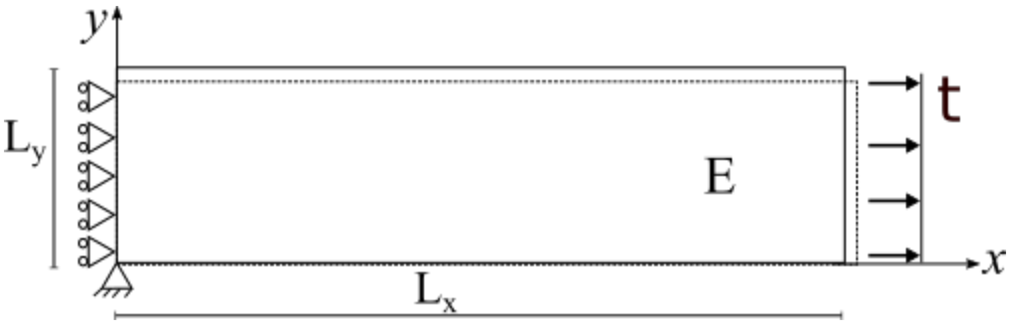


Ejemplo

Características del ejemplo

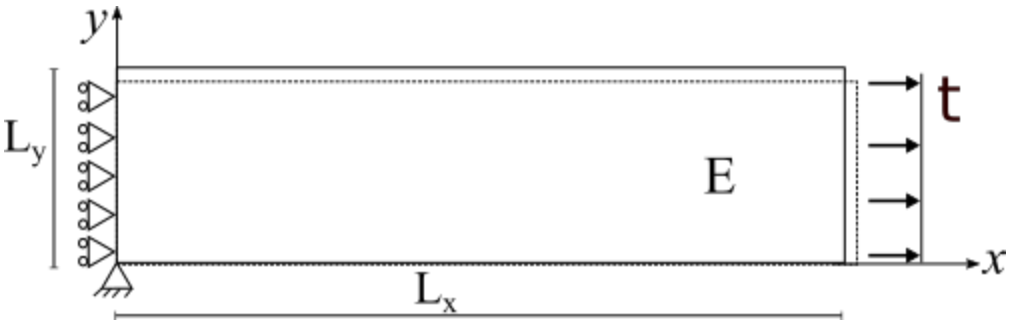
$$Res(\mathbf{x}_i) = \nabla . P(\mathbf{x}_i) + b(\mathbf{x}_i)$$

Placa 2D de Lx = 30mm, Ly = 10mm.
Elástico isotrópico com: E =10 MPa, v = 0.3
Extremo izquierdo fijo deslizante
Fuerza en el extremo: T = 1N
Aplicando equilibrio puntual



Ejemplo

Placa 2D de Lx = 30mm, Ly = 10mm.
Elástico isotrópico com: E =10 MPa, v = 0.3
Extremo izquierdo fijo deslizante
Fuerza en el extremo: T = 1N
Estado plano de tensiones



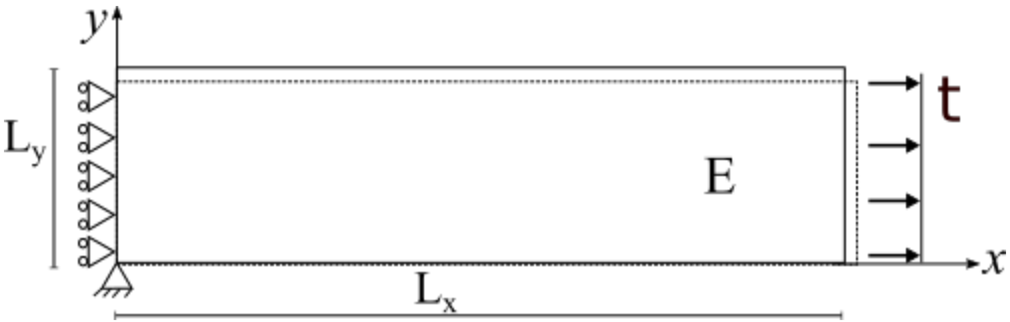
Características del ejemplo

$$Res(\mathbf{x}_i) = \nabla . P(\mathbf{x}_i) + b(\mathbf{x}_i)$$

$$\mathbf{P} = G \begin{pmatrix} \frac{2}{1-\nu}(u_x + \nu v_y) & u_x + v_y \\ u_x + v_y & \frac{2}{1-\nu}(v_y + \nu u_x) \end{pmatrix}$$

Ejemplo

Placa 2D de Lx = 30mm, Ly = 10mm.
Elástico isotrópico com: E =10 MPa, v = 0.3
Extremo izquierdo fijo deslizante
Fuerza en el extremo: T = 1N
Aplicando gradiente y despreciando *b*



Características del ejemplo

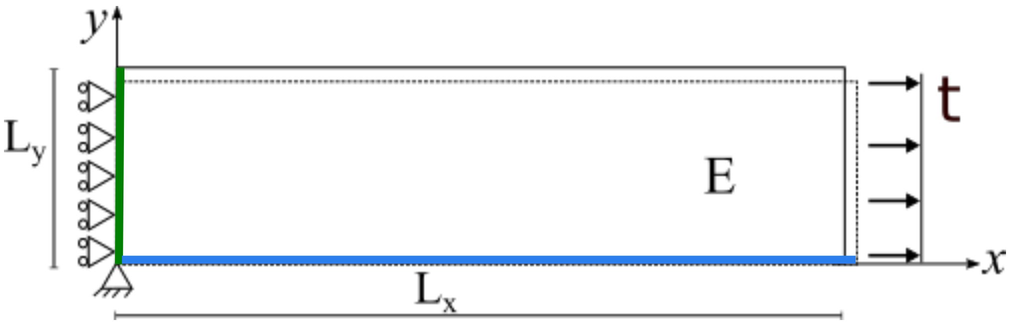
$$Res(\mathbf{x}_i) = \nabla . P(\mathbf{x}_i) + b(\mathbf{x}_i)$$

$$\mathbf{P} = G\begin{pmatrix} \frac{2}{1-\nu}(u_x + \nu v_y) & u_x + v_y \\ u_x + v_y & \frac{2}{1-\nu}(v_y + \nu u_x) \end{pmatrix}$$

$$\nabla . P = G\begin{pmatrix} \frac{2}{1-\nu}(u_{xx} + \nu v_{yx}) + u_{xy} + v_{yy} \\ \frac{2}{1-\nu}(v_{yy} + \nu u_{xy}) + u_{xx} + v_{yx} \end{pmatrix} = 0$$

Ejemplo

Placa 2D de Lx = 30mm, Ly = 10mm.
Elástico isotrópico com: E =10 MPa, v = 0.3
Extremo izquierdo fijo deslizante
Fuerza en el extremo: T = 1N
Condición de borde de desplazamiento nulo



Características del ejemplo

$$Res(\mathbf{x}_i) = \nabla.P(\mathbf{x}_i) + b(\mathbf{x}_i)$$

$$\mathbf{P} = G\begin{pmatrix} \frac{2}{1-\nu}(u_x + \nu v_y) & u_x + v_y \\ u_x + v_y & \frac{2}{1-\nu}(v_y + \nu u_x) \end{pmatrix}$$

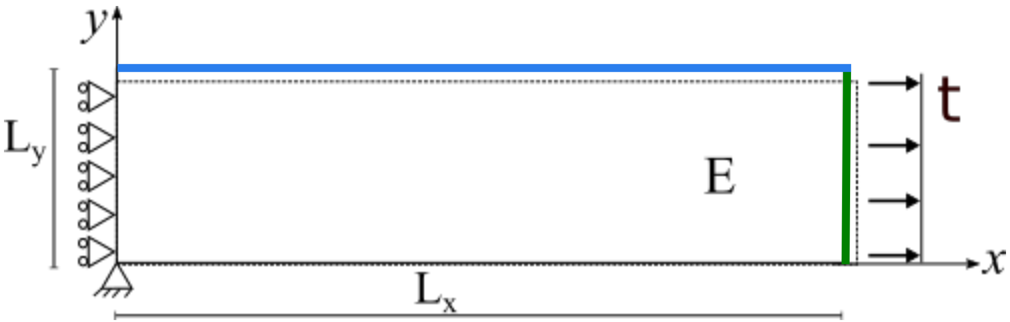
$$\nabla.P = G\begin{pmatrix} \frac{2}{1-\nu}(u_{xx} + \nu v_{yx}) + u_{xy} + v_{yy} \\ \frac{2}{1-\nu}(v_{yy} + \nu u_{xy}) + u_{xx} + v_{yx} \end{pmatrix} = 0$$

$$\mathbf{u}(\mathbf{x}) \cdot (\mathbf{e}_1) = 0 \qquad \forall\, \mathbf{x} \in \partial\Omega_X$$

$$\mathbf{u}(\mathbf{x}) \cdot (\mathbf{e}_2) = 0 \qquad \forall\, \mathbf{x} \in \partial\Omega_Y$$

Ejemplo

Placa 2D de Lx = 30mm, Ly = 10mm.
Elástico isotrópico com: E =10 MPa, v = 0.3
Extremo izquierdo fijo deslizante
Fuerza en el extremo: T = 1N
Condición de borde de carga



Características del ejemplo

$$Res(\mathbf{x}_i) = \nabla . P(\mathbf{x}_i) + b(\mathbf{x}_i)$$

$$\mathbf{P} = G\begin{pmatrix} \frac{2}{1-\nu}(u_x + \nu v_y) & u_x + v_y \\ u_x + v_y & \frac{2}{1-\nu}(v_y + \nu u_x) \end{pmatrix}$$

$$\nabla . P = G\begin{pmatrix} \frac{2}{1-\nu}(u_{xx} + \nu v_{yx}) + u_{xy} + v_{yy} \\ \frac{2}{1-\nu}(v_{yy} + \nu u_{xy}) + u_{xx} + v_{yx} \end{pmatrix} = 0$$

$$\mathbf{u}(\mathbf{x}) \cdot (\mathbf{e}_1) = 0 \qquad \forall\, \mathbf{x} \in \partial\Omega_X$$
$$\mathbf{u}(\mathbf{x}) \cdot (\mathbf{e}_2) = 0 \qquad \forall\, \mathbf{x} \in \partial\Omega_Y$$

$$\mathbf{P}(\mathbf{x})[\mathbf{e}_1] = \mathbf{t} \qquad \forall\, \mathbf{x} \in \partial\Omega_{L_x}$$
$$\mathbf{P}(\mathbf{x})[\mathbf{e}_2] = \mathbf{0} \qquad \forall\, \mathbf{x} \in \partial\Omega_{\mathbf{L_Y}}$$

Ejemplo

UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

IIMPI
INSTITUTO DE
INGENIERIA MECANICA
Y PRODUCCION INDUSTRIAL

Seminario en redes neuronales basadas en la Física

PINNs

[Physics-Informed Neural Networks]

GRACIAS

MSc. Ing. Christian Díaz-Cuadro