# Criptografía con RNA

## Trabajo Fin de Grado

Melisa Belmonte Jiménez

Facultad de Ciencias Matemáticas
Universidad Complutense de Madrid

# Índice

# Índice

# Introducción

- **Polivalencia** de Redes Neuronales $\longrightarrow$ Criptografía
- Métodos actuales: teoría de números.
- **Seguridad**: complejidad computacional alta de problemas $\longrightarrow$ Computación cuántica.
- Imitación de redes $\longrightarrow$ **Redes Neuronales Caóticas**

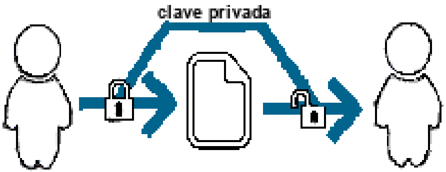
# Índice

# Criptografía

## Simétrica

Emisor y receptor usan la misma clave.

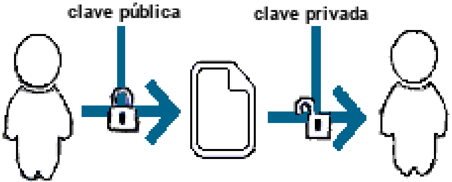
Seguridad basada en el secretismo de la clave.




## Asimétrica

Distintas claves para el cifrado y descifrado.

Seguridad basada en la estructura.



# Criptografía con caos

**Comportamiento aparentemente aleatorio** dado por funciones relativamente simples.

Usos principales han sido:

- Simular **ruido** en las comunicaciones
- Crear **claves** aleatorias
- Definir **funciones hash** más seguras.

| Característica caótica | Propiedad criptográfica | Descripción |
|---|---|---|
| Ergodicidad<br>Topológicamente transitivo | Confusión | Las salidas de distintas entradas no parecen tener relación |
| Sensibilidad a las condiciones iniciales | Difusión | Una pequeña diferencia en la entrada da una salida muy distinta. |
| Determinístico | Pseudo-aleatoriedad determinística | Resultados aparentemente aleatorios, que son determinísticos. |
| Complejidad | Complejidad algorítmica | Un algoritmo simple produce salidas complejas. |

### Definición

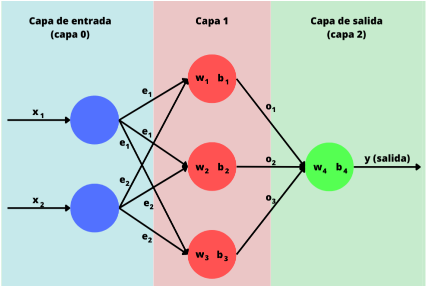
**Ergodicidad:** Propiedad de un sistema que sugiere que un punto del mismo visitará todas las partes del espacio en el que se mueve

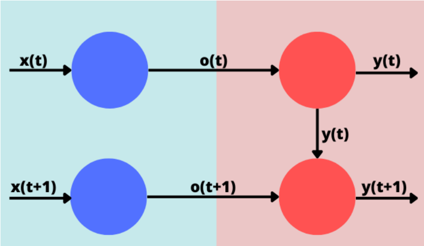
### Definición

Una función $f : X \mapsto Y$ es **topológicamente transitiva** si para todo par de conjuntos abiertos no vacíos $U, V \subset X$ existe un $n \in \mathbb{N}$ tal que $f^{-n}(U) \cap V \neq \emptyset$

# Índice

# Redes Neuronales




## Propagación hacia delante

- Transmisión lineal
- Estructura de capas

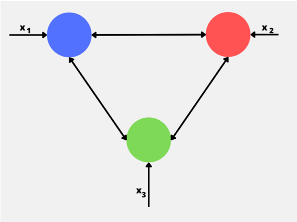
$F(x) = \sigma(W_N \cdot \sigma(...\sigma(W_1 \cdot x + b_1)...) + b_N)$

## Recurrentes

- Memoria
- Transmisión no lineal

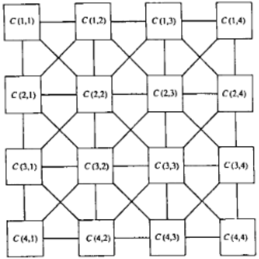
$\frac{\partial x(t)}{\partial t} = -x(t) + W\sigma(x(t)) + I(t)$

# RNA Recurrentes



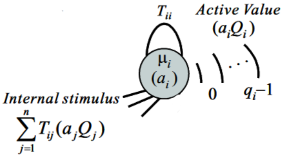

## Hopfield

. Aprenden unos patrones.
. Salida: patrón más similar a la entrada.




## Celulares

. Procesamiento paralelo.
. Localmente conectadas.




## Q'tron

. Minimización de función energía.
. Neuronas fijas y libres.

# Índice

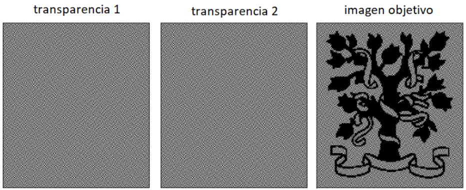
1. Introducción
2. Criptografía
3. Redes Neuronales
4. Criptografía con RNA no sincronizadas
   - Criptografía Visual
5. Criptografía con RNA sincronizadas
6. Conclusiones

# Criptografía con RNA no sincronizadas

Usos:

- Red o conjunto de entrenamiento como **clave privada**
  - Sistemas **robustos**
  - **Claves muy grandes**
- Imitar **funciones**
  - $+$ **rápido** que los métodos de aproximación
- Secuencias **pseudoaleatorias**
  - Con RN de Hopfield y RN Celulares

# Criptografía Visual




-

- $O$: **Imagen objetivo** - 1 Neurona fija/pixel
- $T_i$: **Transparencia i** - 1 Neurona libre/pixel
- $y_{ij}^{x}$ la salida de la neurona correspondiente al pixel en posición $ij$ en la imagen $x \in \{O, T_1, T_2\}$

-

$$\mathbb{E}(\Theta) = \frac{1}{2} \sum_{(o,t_1,t_2)\in\Theta} E(y_o, y_{t_1}, y_{t_2})$$

| $t_1$ | $t_2$ | $o$ | $E$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| | | 1 | 2, 25 |
| 0 | 1 | 1 | 0, 25 |
| | | 0 | 1 |
| 1 | 0 | 1 | 0, 25 |
| | | 0 | 1 |
| 1 | 1 | 1 | 0, 25 |
| | | 0 | 4 |

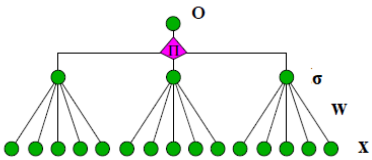
-

$$E(o, t_1, t_2) = (1{,}5o - (t_1 + t_2))^2$$

# Índice

# Criptografía con RNA sincronizadas

## Árbol de Paridad




$$o_i = \begin{cases} sign(x_i^T \cdot w_i) & si \quad x_i^T \cdot w_i \neq 0 \\ a & si \quad x_i^T \cdot w_i = 0 \end{cases}$$

Con $a \in \{-1, 1\}$ fijo
Salida final: $o = \prod_{i=1}^{K} o_i$

## RN Recurrentes

2 condiciones:

- $0 \leq \frac{\sigma_i(x)-\sigma_i(y)}{x-y} \leq h_i$ y $0 \leq \frac{\tilde{\sigma}_i(x)-\tilde{\sigma}_i(y)}{x-y} \leq k_i$ para $x \neq y$ y ciertos $h_i$ y $k_i$.

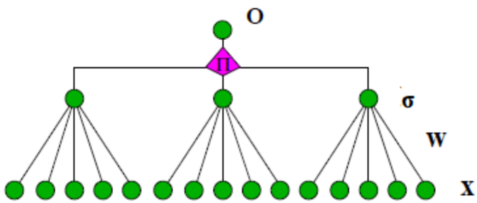
- $\tau(t) \geq 0$ función derivable que cumple $0 \leq \tau(t) \leq \mu < 1 \; \forall t$

$$\frac{\partial x(t)}{\partial t} = -Cx(t) + A\sigma(x(t)) + B\tilde{\sigma}(x(t-\tau(t)) + I$$

$$\frac{\partial y(t)}{\partial t} = -Cy(t) + A\sigma(y(t)) + B\tilde{\sigma}(y(t-\tau(t)) + \varepsilon \odot (y(t) - x(t))$$

# Árbol de Paridad (AP)

## Árbol de Paridad




$$o_i = \begin{cases} sign(x_i^T \cdot w_i) & si \quad x_i^T \cdot w_i \neq 0 \\ a & si \quad x_i^T \cdot w_i = 0 \end{cases}$$

.

.

Con $a \in \{-1, 1\}$ fijo
Salida final: $o = \prod_{i=1}^{K} o_i$

## Funciones de Activación:

- Hebbian:$\sigma(o_i) = o_i$
- Anti-Hebbian:$\sigma(o_i) = -o_i$
- Paseo Aleatorio (*Random Walk*):$\sigma(o_i) = 1$

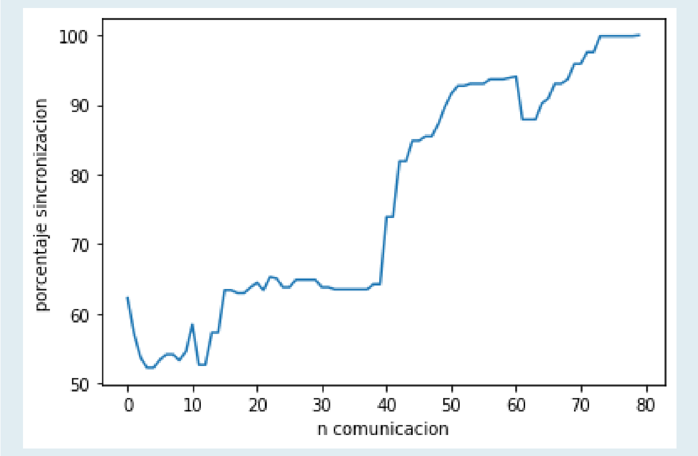
## Cáculo del peso

$$w_i^+ = w_i + \frac{\eta}{N} \cdot \sigma(o_i) \cdot x \quad \forall i \in \{1...K\}$$

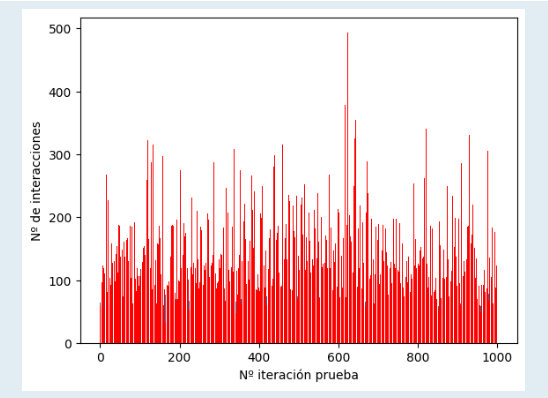
$\eta_i$: la tasa de aprendizaje
$N$: dimensión del peso
$f$: función de aprendizaje
$o_i$: salida obtenida en el perceptrón $i$
$o'$: salida de la otra red
$x$: entrada.

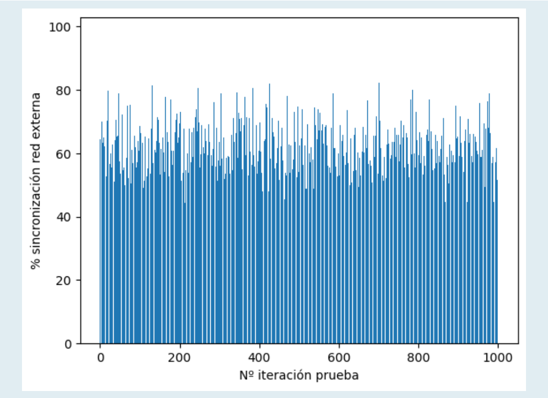
# Implementación

- 5 neuronas de entrada.
- Pesos de dimensión 10.
- Pesos acotados por 10.



# Ataque por fuerza bruta



# Ataque Probabilístico

- **Probabilístico**:
$$P(o_k = 1|o) = \frac{\sum_{(\alpha_1,\ldots,\alpha_k),,\ \prod_{i=1}^{k} \alpha_i = 0 y \alpha_k = 1} \prod_{i=1}^{k} p_i(\alpha_i)}{\sum_{(\alpha_1,\ldots,\alpha_k),,\prod_{i=1}^{k} \alpha_i = 0} \prod_{i=1}^{k} p_i(\alpha_i)}$$
- **Geométrico**: Hiperplanos $X_i : \sum_{j=1}^{N} x_{ij} z_j = 0$ en $U = \{-L, \ldots, L\}^N$, con los pesos como puntos. $o_i$ el lado del hiperplano $X_i$ en el que está $W_i$
- **Genético**: Usa una población de redes.

# Ataque Geométrico

- **Probabilístico**: $P(o_k = 1|o) = \frac{\sum_{(\alpha_1,\dots,\alpha_k),,\ \prod_{i=1}^{k} \alpha_i = 0 y \alpha_k = 1} \prod_{i=1}^{k} p_i(\alpha_i)}{\sum_{(\alpha_1,\dots,\alpha_k),,\prod_{i=1}^{k} \alpha_i = 0} \prod_{i=1}^{k} p_i(\alpha_i)}$
- **Geométrico**:

  .

  Hiperplanos $X_i : \sum_{j=1}^{N} x_{ij} z_j = 0$ en $U = \{-L, ..., L\}^N$

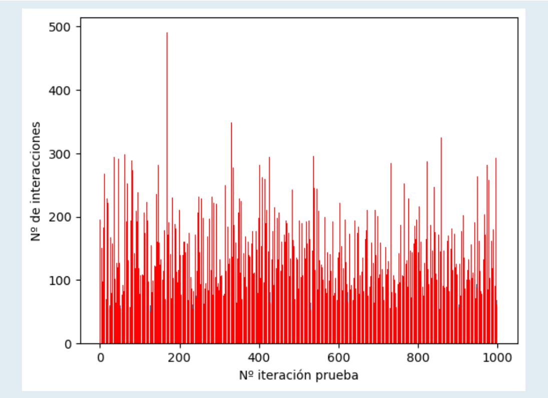
  $o_i$ el lado del hiperplano $X_i$ en el que está $W_i$

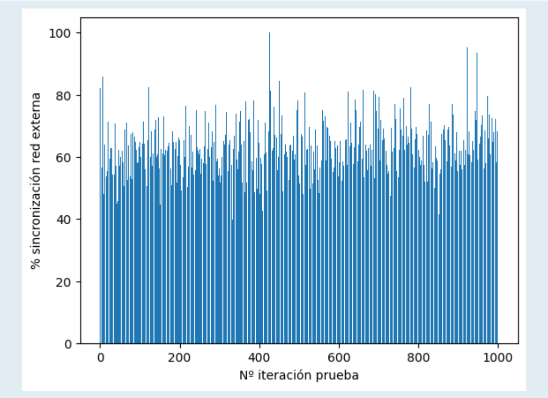
  .

  - $\mathbf{o^A} \neq \mathbf{o^B}$: A y B no cambian, luego C tampoco.
  - $\mathbf{o^A} = \mathbf{o^B} = \mathbf{o^C}$: Se actualiza C de la foma habitual.
  - $\mathbf{o^A} = \mathbf{o^B} \neq \mathbf{o^C}$: Se cambia $o_{i_0}^C = -o_{i_0}^C$ $(i_0 = indmin|\sum_{j=0}^{N} w_{ij}^C \cdot x_{ij}|)$ y luego se actualiza con $o^A$.

  .
- **Genético**: Usa una población de redes.

# Ataque Geométrico



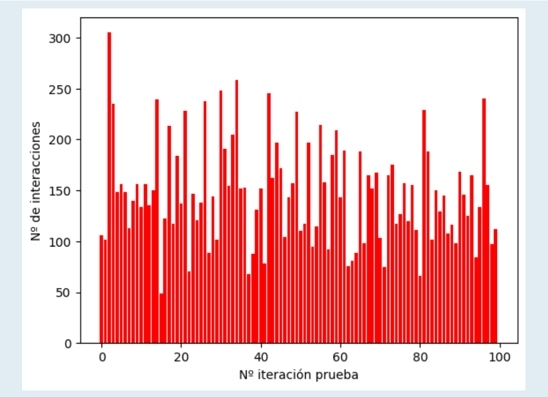



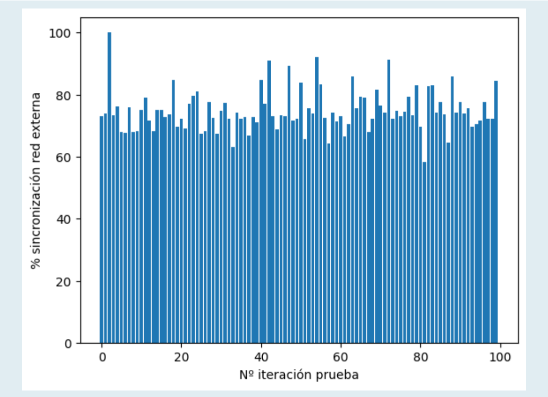
# Ataque Genético

- **Probabilístico**: $P(o_k = 1|o) = \frac{\sum_{(\alpha_1,\ldots,\alpha_k),,\ \prod_{i=1}^{k} \alpha_i = 0 y \alpha_k = 1} \prod_{i=1}^{k} p_i(\alpha_i)}{\sum_{(\alpha_1,\ldots,\alpha_k),,\prod_{i=1}^{k} \alpha_i = 0} \prod_{i=1}^{k} p_i(\alpha_i)}$
- **Geométrico**: Hiperplanos $X_i : \sum_{j=1}^{N} x_{ij} z_j = 0$ en $U = \{-L, ..., L\}^N$, con los pesos como puntos. $o_i$ el lado del hiperplano $X_i$ en el que está $W_i$
- **Genético**: Límite de M redes, inicia una población aleatoria reducida.

  .
  - $\mathbf{o}^{\mathbf{A}} \neq \mathbf{o}^{\mathbf{B}}$: A y B no cambian, luego la población tampoco.
  - $\mathbf{o}^{\mathbf{A}} = \mathbf{o}^{\mathbf{B}}$ y hay **menos de M redes**: Se eliminan las redes con salida distinta, se multiplican las iguales.
  - $\mathbf{o}^{\mathbf{A}} = \mathbf{o}^{\mathbf{B}}$ y hay **M redes o más**: Se eliminan las redes con salidas distintas, se actualizan el resto de la forma habitual.

# Ataque Genético



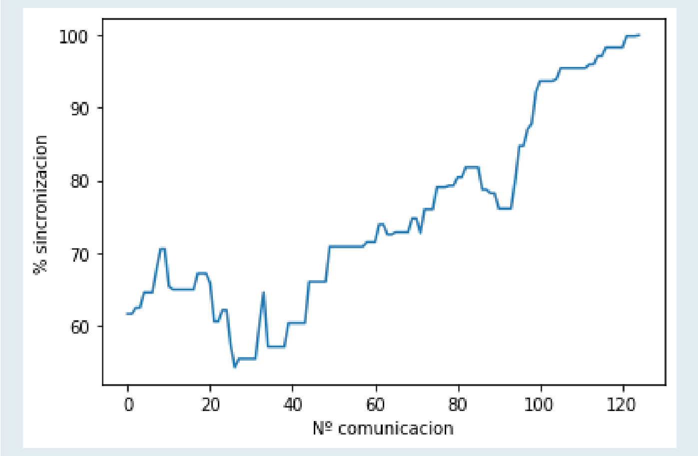



# Mejoras

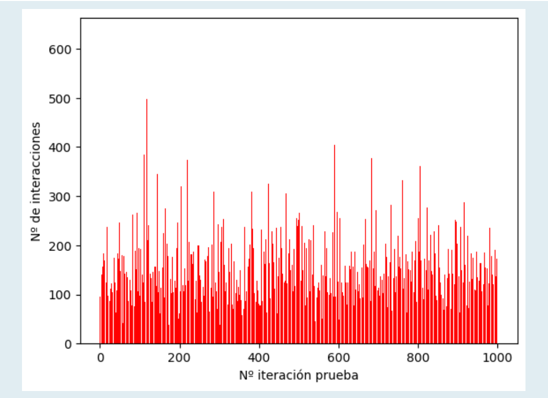
- Mapa caótico.
- Pesos discretos.
- Normalización de los pesos.
- Método para comprobar la sincronización.

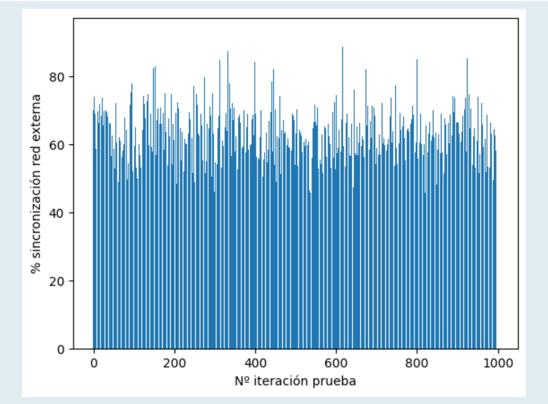
# Implementación

- 5 neuronas de entrada.
- Pesos de dimensión 10.
- Pesos acotados por 10.
- Mapa logístico.

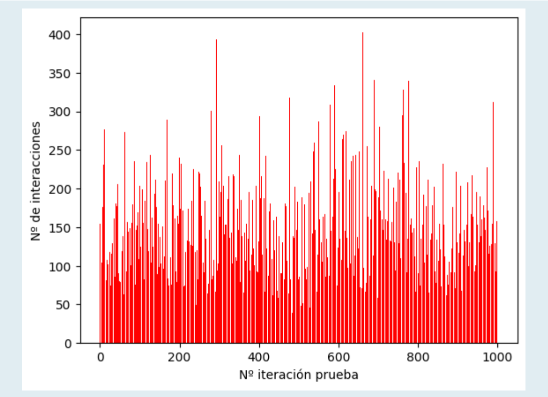


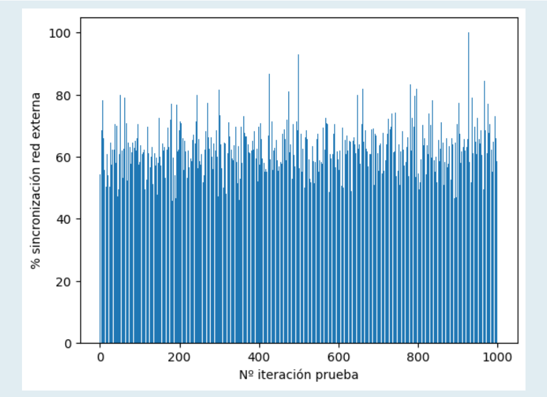
# Ataque por fuerza bruta



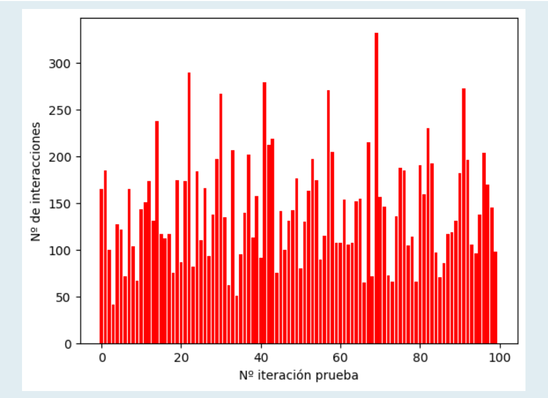



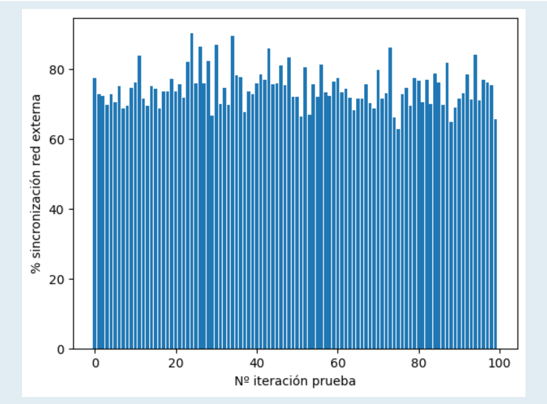
# Ataque Geométrico



# Ataque Genético



# Índice

# Conclusiones

Alternativa a los métodos dependientes de la potencia de los ordenadores.

Destaca el Árbol de Paridad

**Mejoras Árbol de Paridad:**

Método para comprobar la sincronización.

Cambios en su estructura para reducir interacciones.