# Autómata Off-Lattice: Bandadas de agentes autopropulsados

Camila Di Toro Kevin Catino Iván Chayer

Instituto Técnologico de Buenos Aires
[72.27] Simulación de Sistemas

# Contenidos

# Introducción

# Sistema Real

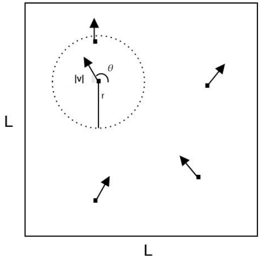
**Sistema Real**

Partículas auto-propulsadas

**Objetivo**

Investigar su auto-organización a partir de su interacción.

# Modelo de partículas auto-propulsadas




Reglas base del modelo:

- Cada partícula se desplaza en cada paso temporal
- Velocidad de módulo constante
- La dirección es un promedio de direcciones de velocidades vecinas en un radio de interacción "r" [a]
- Se adiciona ruido al cálculo de la dirección promedio

[a] El cálculo incluye el angulo de la propia partícula

# Modelo de partículas auto-propulsadas

Posición de la i-ésima partícula para cada tiempo $t$:

$$\boxed{x_i(t+1) = x_i(t) + v_i(t)\Delta t,\ \Delta t = 1} \tag{1}$$

La dirección de la velocidad se obtiene a partir de la expresión:

$$\boxed{\theta(t+1) = \langle\theta(t)\rangle_r + \Delta\theta} \tag{2}$$

## $\langle\theta(t)\rangle$ y $\Delta\theta$

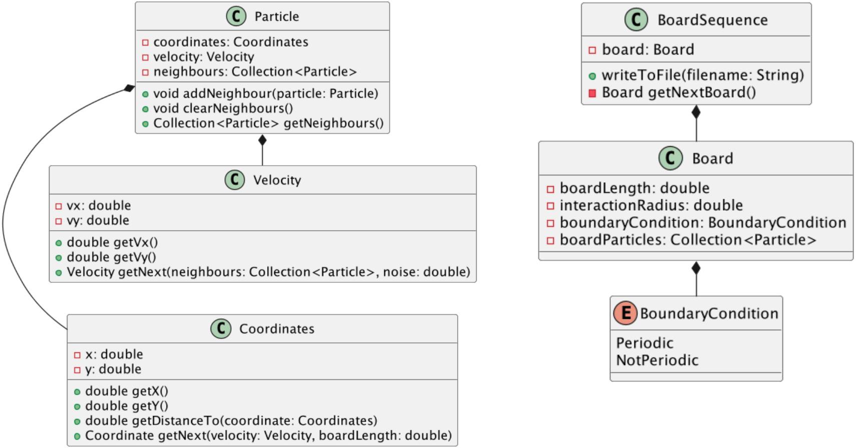
Cálculo del promedio de los ángulos:

$$\langle\theta(t)\rangle_r = atan2\left[\frac{\langle sin(\theta(t))\rangle_r}{\langle cos(\theta(t))\rangle_r}\right] \tag{3}$$

$\Delta\theta$ es el ruido y se obtiene de una distribución unifrome de intervalo $[-\frac{\eta}{2}, \frac{\eta}{2}]$

# Implementación

# Arquitectura



# Motor de simulación

Utiliza el método getNextBoard de la clase BoardSequence para avanzar en el tiempo y obtener el próximo Board.

Resumen de las operaciones realizadas:

- Cálculo de la nueva velocidad y posición
- Se actualiza la velocidad y posición de la partícula
- Se recalcula las celdas en las que se encuentran las partículas
- Se obtienen los vecinos utilizando Cell Index Method

# Actualización de la posición

```
Function getNext(v: Velocity, boardLength: Real) ->
    Coordinates:

    nextX = this.x + v.x
    nextY = this.y + v.y

    // se considera la condicion periodica de borde
    nextX = wrapAxis(nextX, boardLength)
    nextY = wrapAxis(nextY, boardLength)

    return Coordinates(nextX, nextY)
```

# Actualización de la velocidad

```
getNext(neighbours: list<particle>, noise: double) ->
    Velocity:
    noiseValue = RandomBetween(-noise/2, noise/2)

    angles = new list
    for each particle in neighbours:
        angle = arctan(particle.vy / particle.vx)
        angles.add(angle)

    selfAngle = arctan(this.velocity.y / this.velocity.x)
    angles.add(selfAngle)

    sinAvg = promedio de los senos en 'angles'
    cosAvg = promedio de los cosenos en 'angles'

    nextAngle = arctan(sinAvg / cosAvg) + noiseValue
    nextVx = cos(nextAngle) * modulo de v
    nextVy = sin(nextAngle) * modulo de v

    return Velocity(nextVx, nextVy)
```

# Simulaciones

# Modelo Propuesto

- Partículas puntuales en una celda de lado $L$ con condiciones periódicas.
- Módulo de velocidad constante $v = 0.03$.
- Direcciones $\theta$ aleatorias a $t = 0$, $\theta \in [0, 2\pi]$.
- Radio de interacción $r = 1$.
- Generación de $N$ partículas aleatoriamente a $t = 0$.

# Comportamiento del Sistema

- Velocidad promedio normalizada $v_a$ como observable.

$$v_a = \frac{1}{Nv} \left| \sum_{i=1}^{N} v_i \right|$$

- Parámetros de interés: ruido $\eta$ y densidad $\rho = N/L^2$.
- $v_a$ tiende a cero para desorden total y a 1 para partículas polarizadas.

## Simulaciones y Análisis

- Variación de $v_a$ en función del ruido ($\eta$).
- Variación de $v_a$ en función de la densidad ($\rho$)

## Parámetros

Comportamiento de $v_a$ con ruido

- $\eta \in [0, 5]$, $N \in \{40, 100, 400\}$
- Densidad constante $\rho = 4$, ajuste $L$ con $N$.

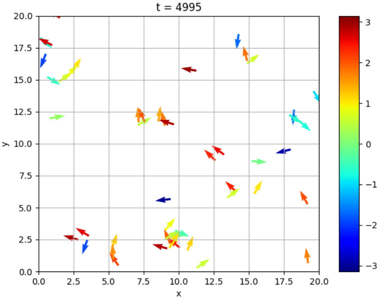
Comportamiento de $v_a$ con densidad

- $\rho \in [0, 10]$, $L = 20$, $\eta = 2.5$

# Cálculo de $v_a$ y Estado Estacionario

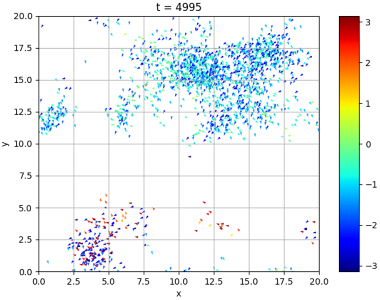
- Se calcula $v_a$ cuando sistema esté estable.
- Se determina el tiempo estacionario con pruebas.

# Baja densidad



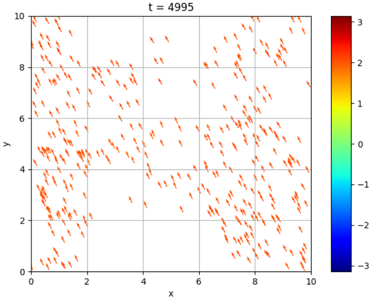

- $L = 20$
- $\eta = 2.5$
- $N = 200$
- $\rho = 0.5$

# Alta densidad



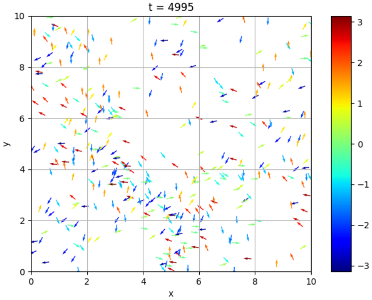

- $L = 20$
- $\eta = 2.5$
- $N = 2000$
- $\rho = 5$

# Bajo ruido




- $L = 10$
- $\eta = 0.1$
- $N = 400$
- $\rho = 4$

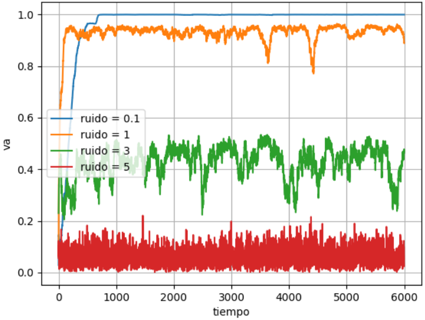
# Alto ruido




- $L = 10$
- $\eta = 5$
- $N = 400$
- $\rho = 4$

# Resultados

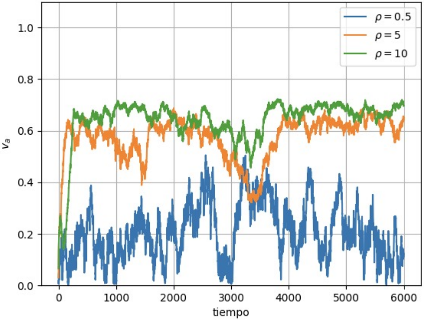
# $v_a$ en función del tiempo




- $L = 10$
- $N = 400$

- $\eta \in \{3, 5\}$: $v_a$ no se estabiliza.
- $\eta = 0.1$: $v_a$ se estabiliza desde $t \approx 700$.
- $\eta = 1$: $v_a$ se estabiliza desde $t \approx 4500$.

## $v_a$ en función del tiempo




- $L = 20$
- $\eta = 2.5$

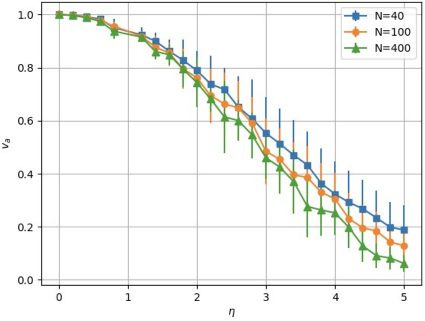
- $\rho \in \{5, 10\}$: $v_a$ se estabiliza desde $t \approx 4000$
- $\rho = 0.5$: $v_a$ no se estabiliza.

# Cálculo de $v_a$

Se decide utilizar el siguiente método:

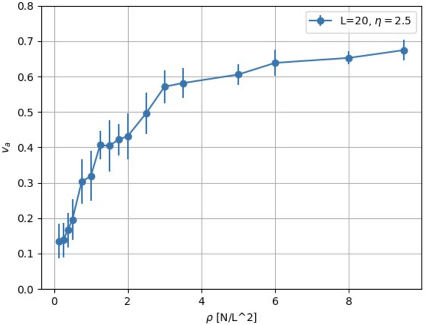
- Si la variación de $v_a$ en 10 iteraciones consecutivas es $< 0.01$, se toma ese promedio.
- Si al llegar a la iteración 5000 no se cumplió la condición anterior, se calcula $v_a$ como el promedio de las iteraciones 5001 a 6000.

# $v_a$ en función del ruido




- $\rho = 4$

## $v_a$ en función de la densidad




- $\eta = 2.5$
- $L = 20$

# Conclusiones

# Conclusiones

- $v_a$ es un indicador de polarización de las partículas del sistema.

## Relación entre $\eta$ y $v_a$

$$\uparrow \eta \implies \downarrow v_a$$

## Relación entre $\rho$ y $v_a$

$$\uparrow \rho \implies \uparrow v_a$$