

6



Foto: Cees Bol

# OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS (PSO)

# ¿QUE ES PSO? (PARTICLE SWARM OPTIMIZATION)



- ▶ Es una **metaheurística bioinspirada** que permite encontrar máximos o mínimos aproximados de una función objetivo determinada dentro de un espacio de búsqueda multidimensional.
- ▶ Esta técnica utiliza un conjunto de **partículas** que se desplazan y colaboran entre sí en un espacio n-dimensional.

# ¿EN QUE PROCESO BIOLÓGICO SE INSPIRA? (I)



- ▶ La metateurística PSO esta basada en el **comportamiento de algunas especies de aves y peces** las cuales poseen una conducta grupal que obedece a una serie de reglas que utilizan para encontrar **en forma óptima** alimento o refugio.



UBA - CEIA - Esp. Ing. Miguel Augusto Azar (2024)

Foto: Alain Delorme



Foto: <https://today.oregonstate.edu>



UBAfiuba



ERÍA

## ¿EN QUE PROCESO BIOLÓGICO SE INSPIRA? (II)



- ▶ La técnica PSO **imita el comportamiento social y comunicacional** que se establece en grupos y utilizan la inteligencia colectiva como recurso para hallar soluciones; esto significa que **ajustan sus movimientos** para evitar depredadores (ademas de la búsqueda de comida).



BAfiuba  
FACULTAD DE INGENIERÍA

## ¿QUIENES DESARROLLARON LA TÉCNICA PSO? (I)



- ▶ James Kennedy (Psicólogo social)
- ▶ Russell C. Eberhart (Ingeniero eléctrico)
- ▶ 1995



Kennedy



Eberhart

# ¿QUIENES DESARROLLARON LA TÉCNICA PSO? (II)



- ▶ Originalmente, Kennedy y Eberhart, comenzaron desarrollando **simulaciones** de software de bandadas de aves alrededor de fuentes de alimento.
- ▶ Luego descubrieron que sus **algoritmos** funcionaban muy bien en problemas de optimización.

# CARACTERÍSTICAS Y VENTAJAS DE PSO

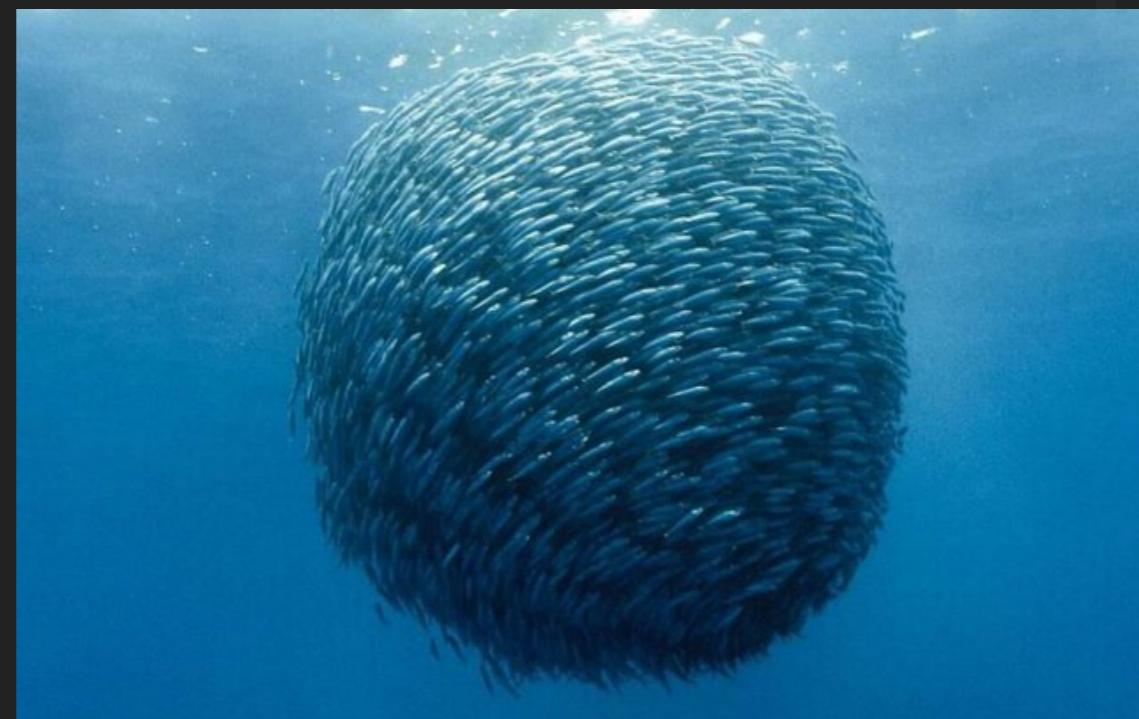


- ▶ Es simple, rápido y fácil de ser programado.
- ▶ Su requisito para el almacenamiento de memoria es mínimo.
- ▶ Tiene memoria, esto es, cada partícula recuerda su mejor solución (**personal best**), así como la mejor solución grupal (**global best**).
- ▶ Se mantiene la población inicial, por lo que no hay necesidad de aplicar operadores a la población, proceso que ralentiza notablemente la performance del sistema.
- ▶ Se basa en "cooperación constructiva" entre las partículas (**es cooperativo**), en contraste con los algoritmos genéticos, que se basan en "la supervivencia del mas apto" (**es competitivo**).

# INTELIGENCIA DE ENJAMBRES (SWARM INTELLIGENCE)



- ▶ El término **enjambre** se utiliza para representar un agrupamiento (de formas de vida, animales o insectos) que **trabajan colectivamente** para realizar sus tareas de una manera **inteligente** y eficiente los cuales interactúan entre sí y con su entorno.



# INTELIGENCIA DE ENJAMBRES (SWARM INTELLIGENCE)



- ▶ Los ejemplos naturales de Inteligencia de Enjambre incluyen **colonias de hormigas, bandadas de aves, crecimiento bacteriano, cardúmenes de peces, entre otros.**

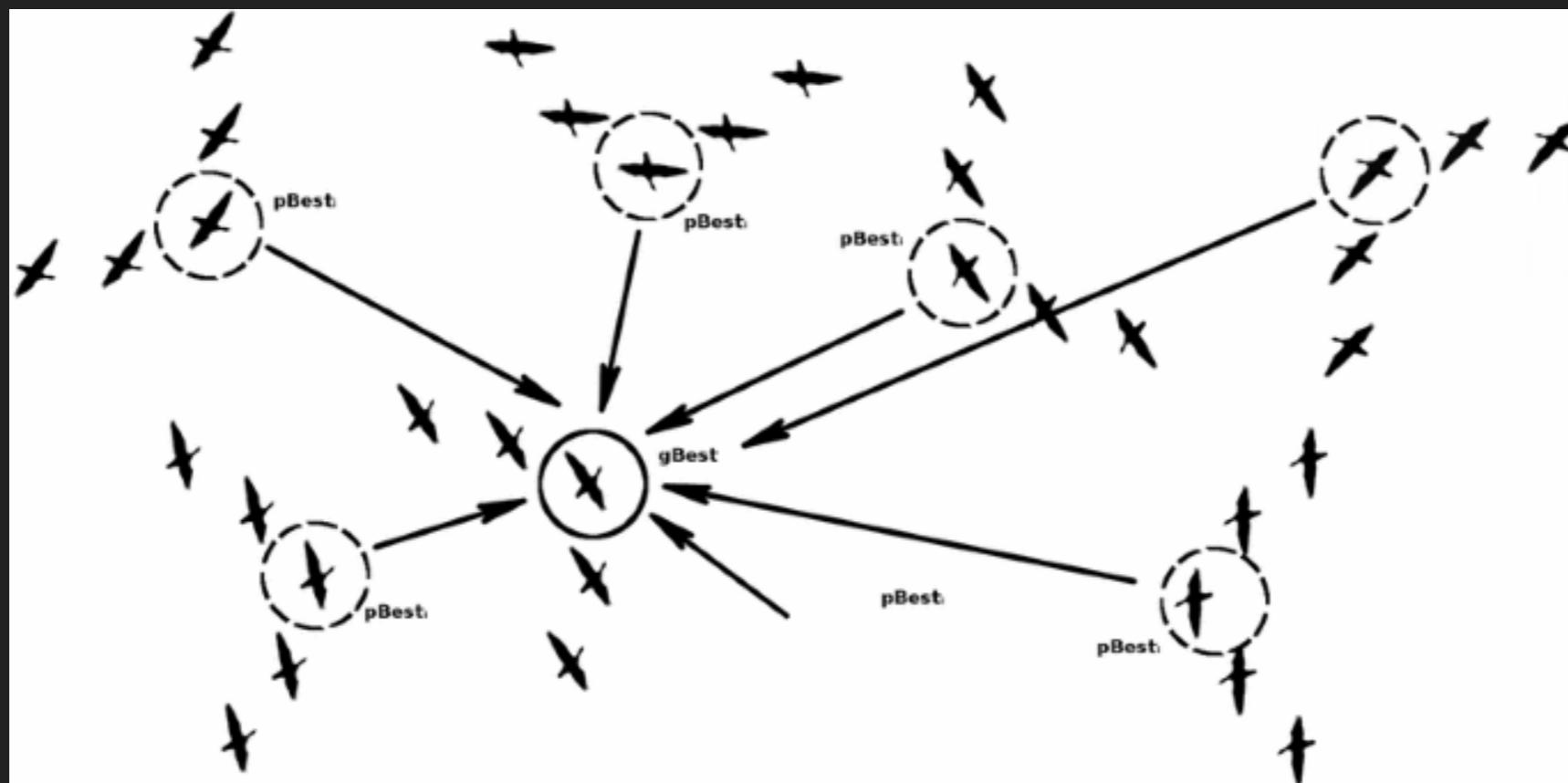


Afiuba  
FACULTAD DE INGENIERÍA

# ¿CÓMO SE MODELA UN ENJAMBRE? (I)



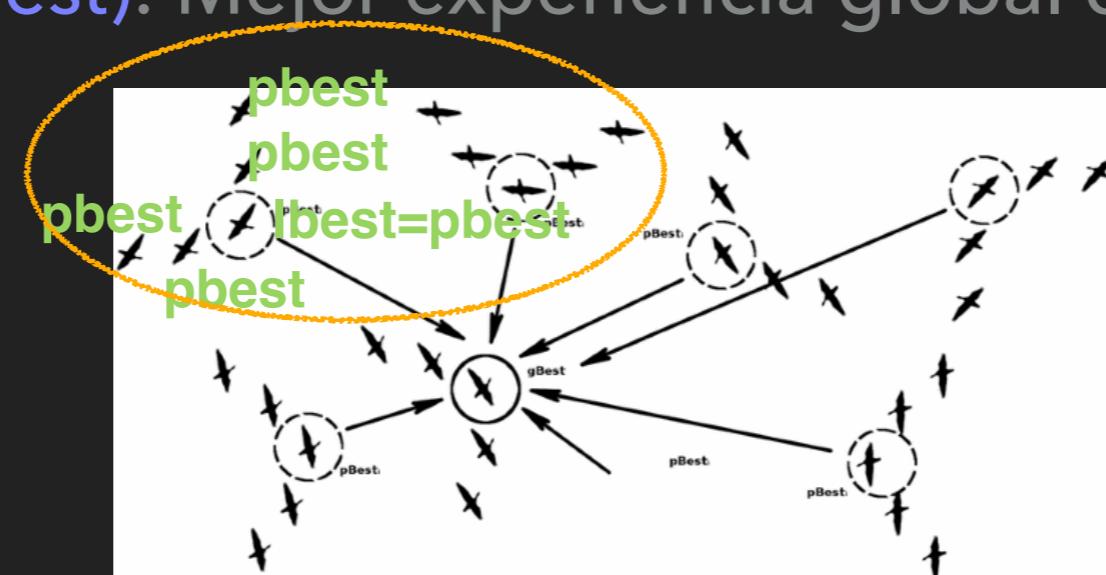
- ▶ Cada ave o pez se representa mediante una **partícula** que está siempre en continuo movimiento dentro del espacio de búsqueda; almacenando, e indirectamente, **comunicando** a todo el enjambre la **mejor solución** que han encontrado hasta el momento.



## ¿CÓMO SE MODELA UN ENJAMBRE? (II)



- ▶ pbest (personal best): Mejor experiencia (evaluación más cercana al óptimo de la función de adaptación) personal (o individual) de cada partícula.
- ▶ lbest (local best): Mejor experiencia local de un grupo de partículas pertenecientes a una vencindad o entorno.
- ▶ gbest: (global best): Mejor experiencia global de todo el cúmulo de partículas.



# ¿CÓMO SE MODELA UN ENJAMBRE? (III)

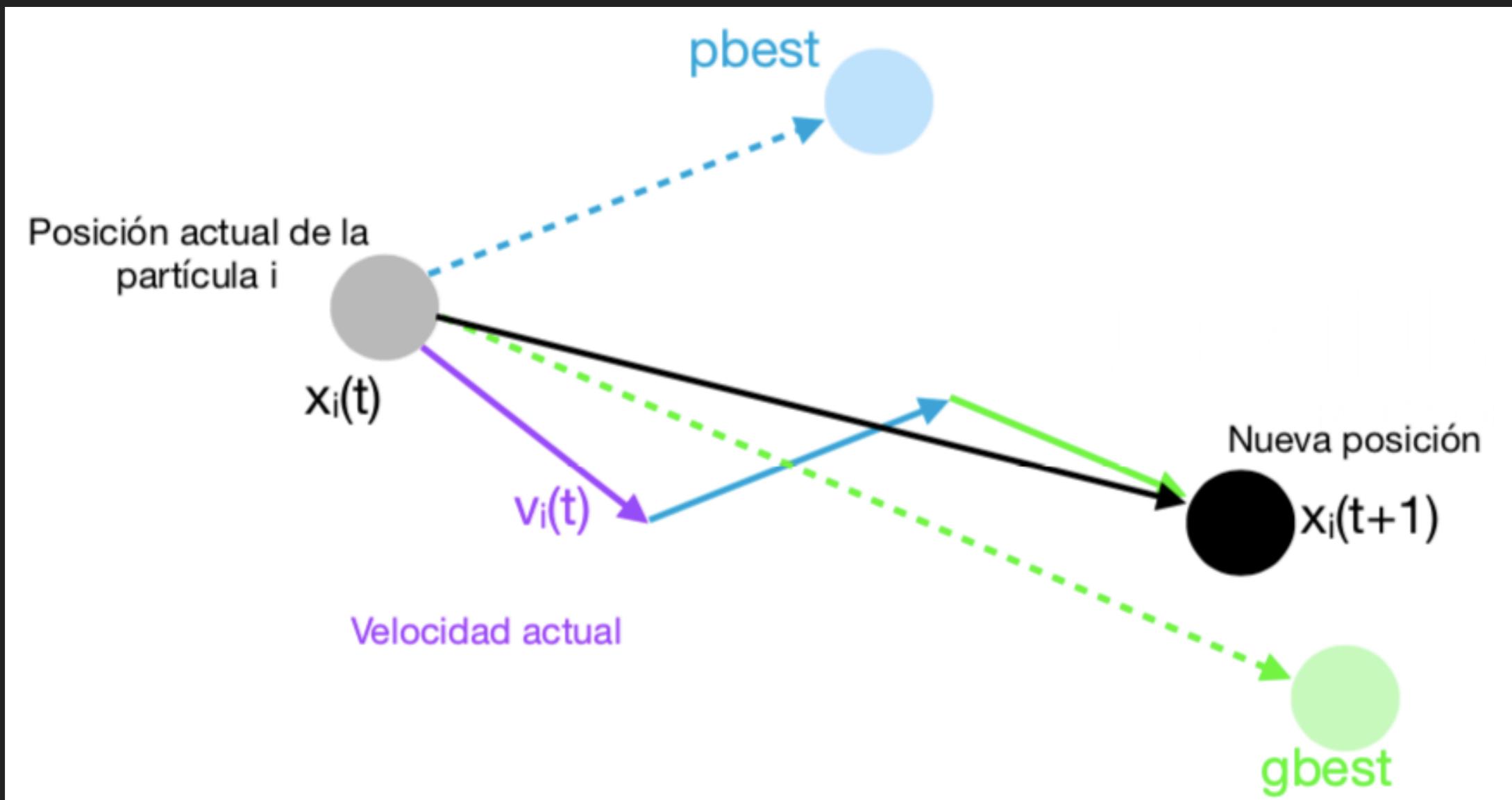


- ▶ Las **partículas**, "vuelan" a través del espacio de búsqueda hiperdimensional.
- ▶ Los cambios en la posición de las **partículas** dentro del espacio de búsqueda se basan en la tendencia sociopsicológica de los individuos a **emular el éxito de otros individuos**. (Engelbrecht, 2007)
- ▶ Los cambios (o el comportamiento) en una partícula dentro del enjambre están influenciados por la **experiencia o el conocimiento de sus vecinos**.
- ▶ La consecuencia de modelar este comportamiento social es que el proceso de búsqueda es tal que **las partículas regresan estocásticamente hacia regiones previamente exitosas** en el espacio de búsqueda.

# ¿CÓMO SE MUEVEN LAS PARTÍCULAS?



- ▶ Trayectoria de las partículas (Velocidad y Nueva posición)

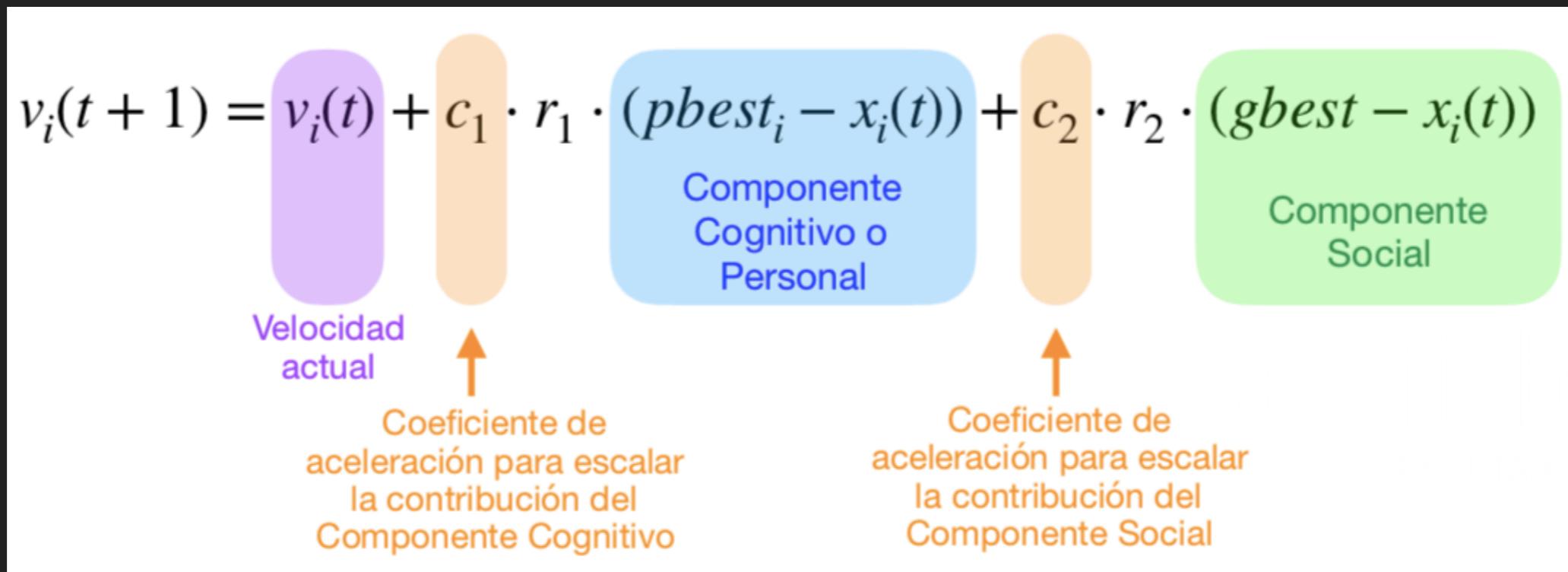


INGENIERÍA

# ¿CÓMO SE MUEVEN LAS PARTÍCULAS?



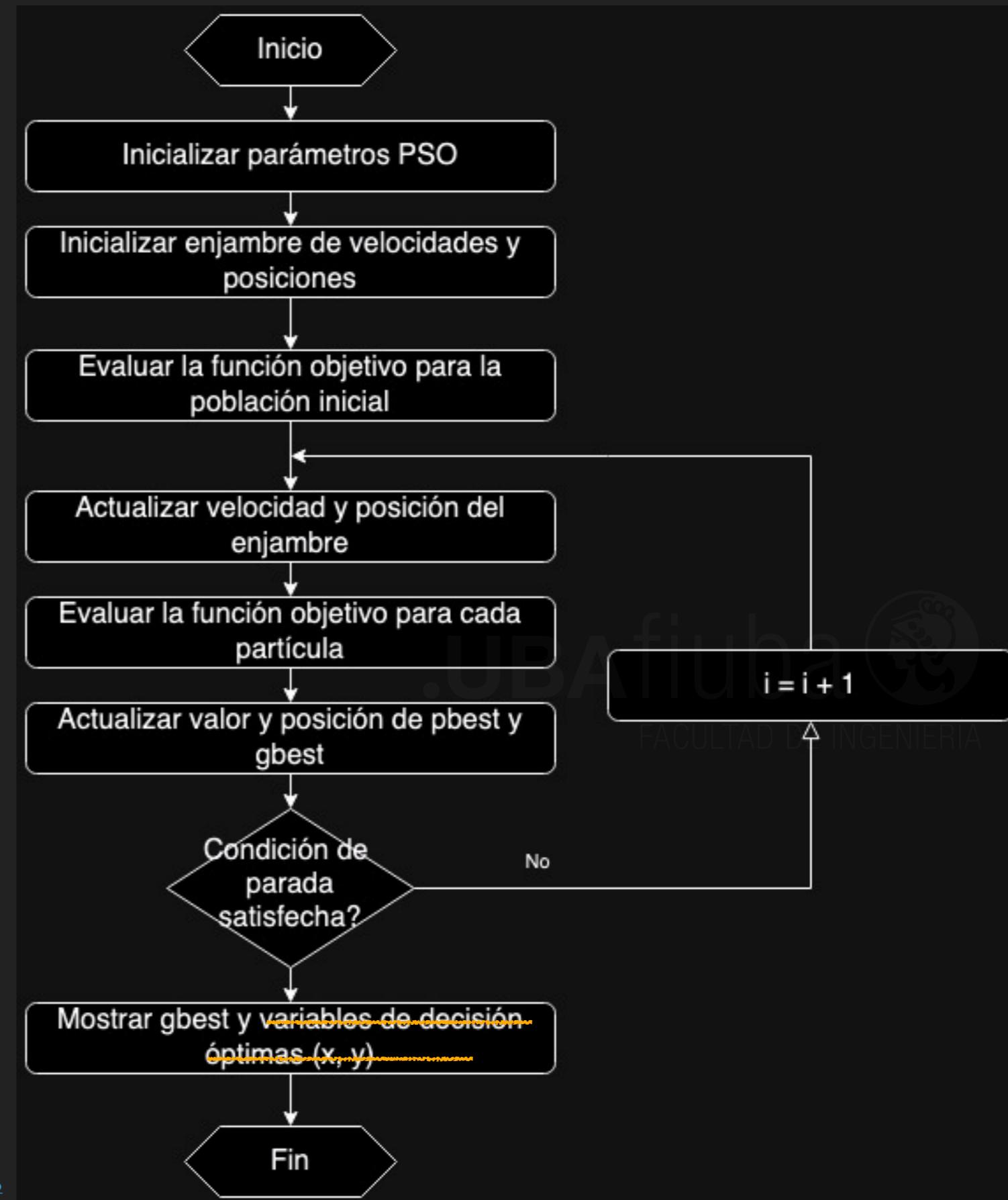
## ► Trayectoria de las partículas (**Velocidad**)



## ► Trayectoria de las partículas (**Nueva posición**)

$$x_i(t + 1) = x_i(t) + v_i(t + 1)$$

# ESTRUCTURA DEL ALGORITMO



# PSEUDOCÓDIGO DEL ALGORITMO PSO



1. Inicializar enjambre aleatoriamente (posiciones de las partículas  $x_i$ )
2. Evaluar cada partícula (según la función objetivo)
3. Comparar cada  $f(x_i)$  con  $f(pbest)$ , elegir el mejor de ambos y obtener  $gbest$
4. Cambiar el vector velocidad de cada partícula  $vel(x_i)$
5. Mover cada partícula  $x_i$  a una nueva posición
6. Ir al Paso 2 y repetir hasta condición de parada

## FACTOR O COEFICIENTE DE INERCIA W (I)



$$v_i(t + 1) = w \cdot v_i(t) + c_1 \cdot r_1 \cdot (pbest_i - x_i(t)) + c_2 \cdot r_2 \cdot (gbest - x_i(t))$$

- ▶ Su función principal es balancear la **exploración** y **explotación** del espacio de búsqueda.
- ▶ **w** reduce o aumenta a la velocidad de la partícula.
- ▶ Influencia en la rapidez con la que una partícula cambia de dirección. Esto permite que las **partículas mantengan parte de su velocidad anterior**, promoviendo **movimientos suaves** en el espacio de búsqueda.

## FACTOR O COEFICIENTE DE INERCIA W (II)



- ▶ **Exploración:**  $w$  alto => explorar una mayor área del espacio de búsqueda, ayudando a evitar que el enjambre se atasque en óptimos locales prematuros.
- ▶ **Explotación:**  $w$  bajo => las partículas se mueven mas lentamente y tienden a converger hacia las mejores soluciones encontradas hasta el momento.  
 $w(t+1) = w(t) / t$
- ▶ **W dinámico:** Al inicio, un  $w$  alto fomenta la exploración, y a medida que el algoritmo progresá, reducir  $w$  favorece la explotación y la **convergencia fina** hacia la solución óptima.

$$v_i(t + 1) = w \cdot v_i(t) + c_1 \cdot r_1 \cdot (pbest_i - x_i(t)) + c_2 \cdot r_2 \cdot (gbest - x_i(t))$$

# ESTRUCTURA DE UNA PARTÍCULA (I)



Un problema de optimización puede tener n dimensiones, esto es, n variables de decisión.

Ejemplo 1 (2 dimensiones):

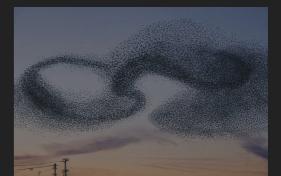
$$f(x, y) = \operatorname{seno}(x) + \operatorname{coseno}(y)$$

Ejemplo 2 (5 dimensiones):

$$f(x_1, x_2, x_3, x_4, x_5) = x_1^2 + 2x_3/x_5^2 - \operatorname{sen}(x_2/x_4)$$

particula i	0.43	1.14	0.92	0.57	1.06
	dimensión 1	dimensión 2	dimensión 3	dimensión 4	dimensión 5

# ESTRUCTURA DE UNA PARTÍCULA (I)



Cúmulo de 4 partículas para un problema de 5 dimensiones:

particula 1	0.43	1.14	0.92	0.57	1.06
particula 2	0.19	0.32	1.21	0.08	1.39
particula 3	1.28	0.68	0.96	0.71	0.42
particula 4	0.94	1.37	1.44	0.29	1.27
	dimensión 1	dimensión 2	dimensión 3	dimensión 4	dimensión 5



DEPARTAMENTO  
DE INGENIERÍA

## EJEMPLO PASO A PASO (II)



### ► 1. Definición del Problema

► Ejercicio: Obtener los valores de  $x$  e  $y$  que minimizan la función  $f(x, y) = x^2 + y^2$  en el intervalo  $[0, 100]$ .

### ► Función objetivo:

$$\text{minimizar} \quad f(x, y) = x^2 + y^2$$

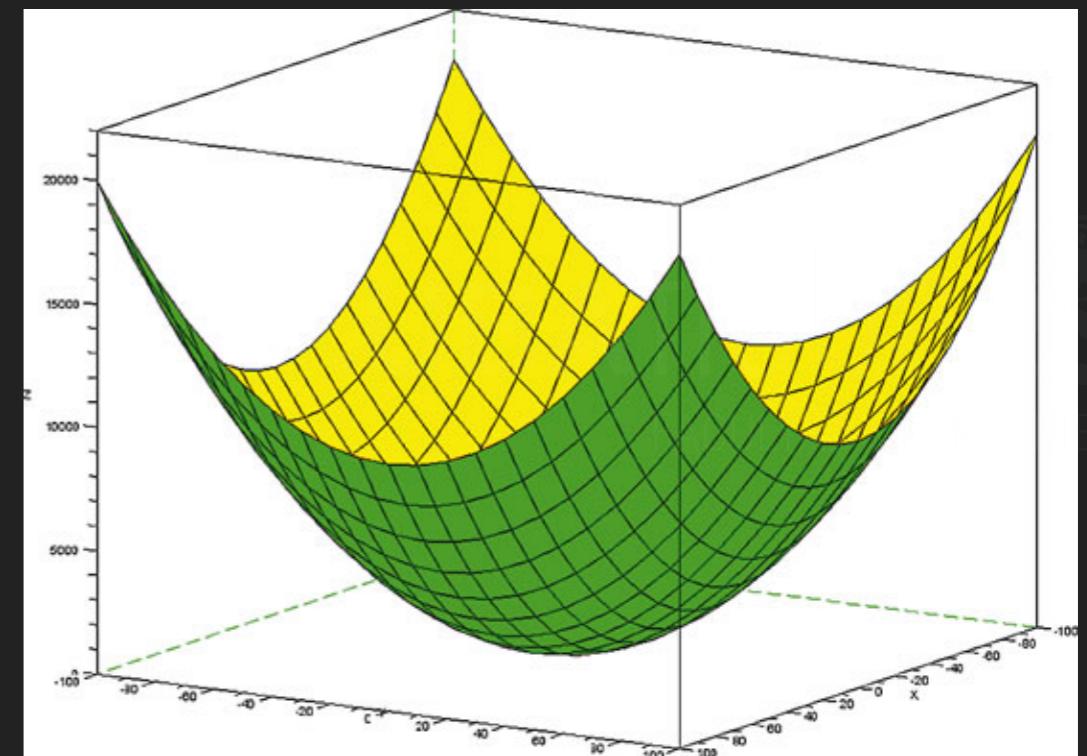
### ► Partícula:

Dimensión = 2, variables  $x, y \in \mathbb{R}$

$$0 \leq x \leq 100$$

$$0 \leq y \leq 100$$

Cada partícula está definida por el vector  $\text{particula} = [x, y]$



## EJEMPLO PASO A PASO (II)



- ▶ 2. Definición de parámetros y sus valores:

parámetro	valor
número de partículas	5
cant. iteraciones	4
dimensión	2
coef. aceleración	1.49

# EJEMPLO PASO A PASO (III)



## ► 3. Inicialización

particula	posicion		velocidad	v(i, 1)	v(i, 2)	pbest	x	y
	x	y						
1	65.81	67.76	1	0	0	1	65.81	67.76
2	98.73	61.00	2	0	0	2	98.73	61.00
3	87.18	42.88	3	0	0	3	87.18	42.88
4	78.87	2.71	4	0	0	4	78.87	2.71
5	72.00	25.22	5	0	0	5	72.00	25.22

## EJEMPLO PASO A PASO (IV)

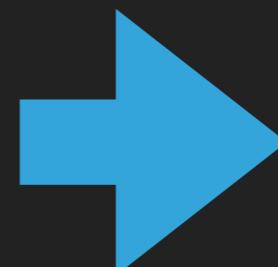


- ▶ 4. Búsqueda (Iteración 1)
- ▶ Actualizar velocidad para cada partícula

$$\text{velocidad}_i(t + 1) = \text{velocidad}_i(t) + c_1 \cdot r_1 \cdot (\text{pbest}_i - \text{posicion}_i(t)) + c_2 \cdot r_2 \cdot (\text{gbest} - \text{posicion}_i(t))$$

- ▶ **gbest** no es dato, por lo tanto lo obtengo del mínimo de todos los  $f(x, y)$ :

pbest	x	y	f(x, y)
1	65.81	67.76	8922.38
2	98.73	61.00	13468.61
3	87.18	42.88	9439.05
4	78.87	2.71	6227.82
5	72.00	25.22	5820.05



	x	y
gbest	72.00	25.22
f(x, y)	5820.05	

## EJEMPLO PASO A PASO (V)



- ▶ 4. Búsqueda (Iteración 1)
- ▶ Actualizar velocidad para cada partícula:

$$\text{velocidad}_i(t + 1) = \text{velocidad}_i(t) + c_1 \cdot r_1 \cdot (\text{pbest}_i - \text{posicion}_i(t)) + c_2 \cdot r_2 \cdot (\text{gbest} - \text{posicion}_i(t))$$

$$\text{velocidad}(1,1) = 0 + 1.49 * 0.34 * (65.81 - 65.81) + 1.49 * 0.22 * (72.00 - 65.81)$$

$$\text{velocidad}(1,1) = 2.03$$

$$\text{velocidad}(1,2) = 0 + 1.49 * 0.34 * (67.76 - 67.76) + 1.49 * 0.22 * (25.22 - 67.76)$$

$$\text{velocidad}(1,2) = -13.94$$

## EJEMPLO PASO A PASO (VI)



- ▶ 4. Búsqueda (Iteración 1)
- ▶ Actualizar velocidad para cada partícula:

velocidad	v(i, 1)	v(i, 2)
1	2.03	-13.94
2	-8.76	-11.72
3	-4.97	-5.78
4	-2.25	7.37
5	0	0

## EJEMPLO PASO A PASO (VII)



- ▶ 4. Búsqueda (Iteración 1)
- ▶ Actualizar posiciones de cada partícula:  $x_i(t + 1) = x_i(t) + v_i(t + 1)$
- ▶ nueva\_posicion = posicion\_actual + velocidad

particula	posicion	
	x	y
1	65.81	67.76
2	98.73	61.00
3	87.18	42.88
4	78.87	2.71
5	72.00	25.22



velocidad	v(i, 1)	v(i, 2)
1	2.03	-13.94
2	-8.76	-11.72
3	-4.97	-5.78
4	-2.25	7.37
5	0	0



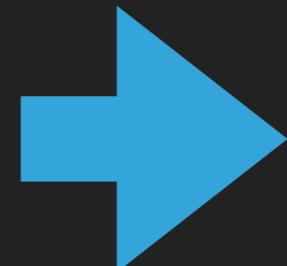
particula	posicion	
	x	y
1	67.84	53.82
2	89.96	49.27
3	82.2	37.09
4	76.62	10.09
5	72.00	25.22

## EJEMPLO PASO A PASO (VIII)



### ► 4. Búsqueda (Iteración 2)

pbest (iter 2)	x	y	f(x, y)
1	67.84	53.82	7428.24
2	89.96	49.27	10521.89
3	82.2	37.09	8133.24
4	76.62	10.09	5972.1
5	72.00	25.22	5820.05



	x	y
gbest	72.00	25.22
f(x, y)	5820.05	

# EJEMPLO PASO A PASO (IX)



## ► 4. Búsqueda (Iteración 2)

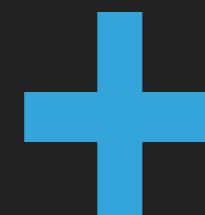
velocidad	v(i, 1)	v(i, 2)
1	1.33	-22.04
2	-13.85	-18.54
3	-7.86	-9.15
4	-3.56	4.85
5	0	0

# EJEMPLO PASO A PASO (X)



## ► 4. Búsqueda (Iteración 2)

particula	posicion	
	x	y
1	67.84	53.82
2	89.96	49.27
3	82.2	37.09
4	76.62	10.09
5	72.00	25.22



velocidad	v(i, 1)	v(i, 2)
1	1.33	-22.04
2	-13.85	-18.54
3	-7.86	-9.15
4	-3.56	4.85
5	0	0



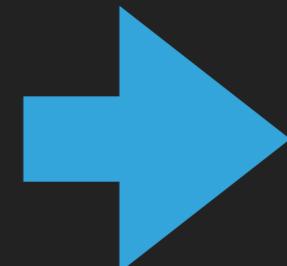
particula	posicion	
	x	y
1	69.17	31.78
2	76.12	30.73
3	74.34	27.94
4	73.06	14.93
5	72.00	25.22

## EJEMPLO PASO A PASO (XI)



### ► 4. Búsqueda (Iteración 3)

pbest (iter 3)	x	y	f(x, y)
1	69.17	31.78	5794.4
2	76.12	30.73	6738.68
3	74.34	27.94	6307.05
4	73.06	14.93	5560.61
5	72.00	25.22	5820.05



	x	y
gbest	73.06	14.93
f(x, y)	5560.61	

## EJEMPLO PASO A PASO (XII)



- ▶ Continuar iterando...

# BIBLIOTECA PYSWARM (I)



Instalación: `pip install pyswarm`

Declaración: `from pyswarm import pso`

Formato función y parámetros:

```
def pso(func: function, lb: array.pyi, ub: array.pyi, ieqcons: list = [],  
f_ieqcons: function = None, args: tuple = (), kwargs: dict = {},  
swarmsize: int = 100, omega: Any = 0.5, phip: Any = 0.5, phig: Any =  
0.5, maxiter: int = 100, minstep: Any = 1e-8, minfunc: Any = 1e-8,  
debug: bool = False) -> array.pyi
```

## BIBLIOTECA PYSWARM (II) - PARÁMETROS



**func:** La función objetivo que se desea minimizar.

**lb:** Un array que representa los límites inferiores para cada una de las variables de decisión.

**ub:** Un array que representa los límites superiores para cada una de las variables de decisión.

**ieqcons:** Una lista de funciones que representan las restricciones de igualdad

**f\_ieqcons:** Una función que agrega restricciones de igualdad y desigualdad.

**args:** Argumentos adicionales que se pasarán a la función objetivo func.

**kwargs:** Argumentos adicionales en formato diccionario que se pasarán a la función objetivo.

## BIBLIOTECA PYSWARM (III) - PARÁMETROS



**swarmsize:** El número de partículas en el enjambre.

**omega:** El factor de inercia ( $w$ ).

**phip:** El coeficiente de aceleración C1 (componente cognitiva).

**phig:** El coeficiente de aceleración C2 (componente social).

**maxiter:** El número máximo de iteraciones permitidas para el algoritmo

**minstep:** Tamaño mínimo de paso que las partículas pueden hacer entre iteraciones. Si todas las partículas se mueven menos que este valor entre iteraciones, el algoritmo se considera convergido.

**minfunc:** Cambio mínimo en la función objetivo que se considera significativo. Si el cambio en el valor de la función objetivo entre iteraciones es menor que este valor, el algoritmo puede detenerse.

**debug:** Un indicador booleano que, si está activado, permite la salida de información de depuración durante la ejecución del algoritmo, lo que puede ser útil para entender el comportamiento del PSO.

# BIBLIOTECA PYSWARM (IV) - EJEMPLO



```
from pyswarm import pso

# función objetivo
def funcion_objetivo(x):
    return x[0]**2 + x[1]**2

lb = [-100, -100] # límite inf
ub = [100, 100] # límite sup

num_particulas = 10 # numero de particulas
cantidad_iteraciones = 20 # numero maximo de iteraciones

# Llamada a la función pso
solucion_optima, valor_optimo = pso(funcion_objetivo, lb, ub, swarmsize=num_particulas, maxiter=cantidad_iteraciones, debug=True)

# Resultados
print("\nSolución óptima (x, y):", solucion_optima)
print("Valor óptimo:", valor_optimo)
```

# REFERENCIAS BIBLIOGRÁFICAS Y WEB



- ▶ J. Kennedy, R. Eberhart. (1995). Particle swarm optimization (in Neural Networks). Proceedings., IEEE International Conference on, vol. 4, pp. 1942 -1948 vol.4.
- ▶ Engelbrecht, A. P. (2007). Computational intelligence: an introduction. John Wiley & Sons.
- ▶ Clerc, M. (2010). Particle swarm optimization (Vol. 93). John Wiley & Sons.
- ▶ Shi, Y., & Eberhart, R. (1998). A Modified Particle Swarm Optimizer. Proceedings of the IEEE International Conference on Evolutionary Computation, 1998. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), Anchorage, AK, USA, 1998, pp. 69-73. DOI: 10.1109/ICEC.1998.699146.
- ▶ Bratton, D., & Kennedy, J. (2007, April). Defining a standard for particle swarm optimization. In 2007 IEEE swarm intelligence symposium (pp. 120-127). IEEE.
- ▶ Parsopoulos, K. E., & Vrahatis, M. N. (2002). Particle swarm optimization method for constrained optimization problems. Intelligent technologies-theory and application: New trends in intelligent technologies, 76(1), 214-220.