

Algoritmos de optimización bioinspirados

Inteligencia Artificial
Segundo semestre 2019

Profesor: Jorge Maturana
jorge.maturana@inf.uach.cl

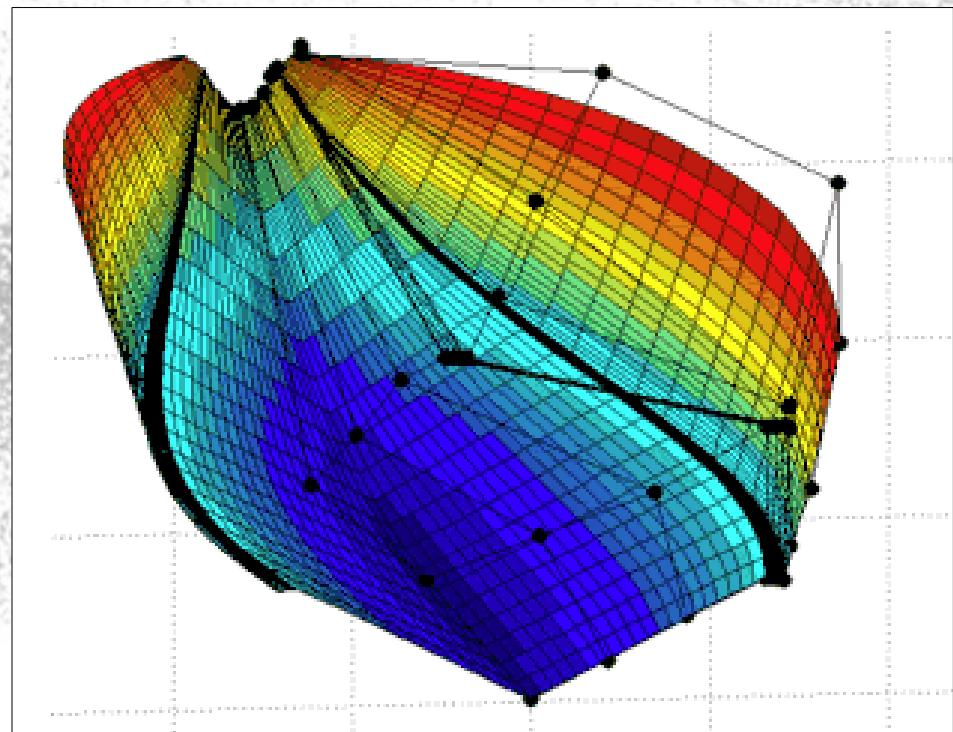
Introducción

- La Ingeniería está asociada al diseño de artefactos eficientes
- La eficiencia de estos artefactos depende de **parámetros** de diseño
 - Capacidad de carga
 - Grosor de vigas
 - Elección de materiales
 - etc.
- La correcta elección de estos parámetros, de acuerdo a un objetivo, es materia de la **optimización**

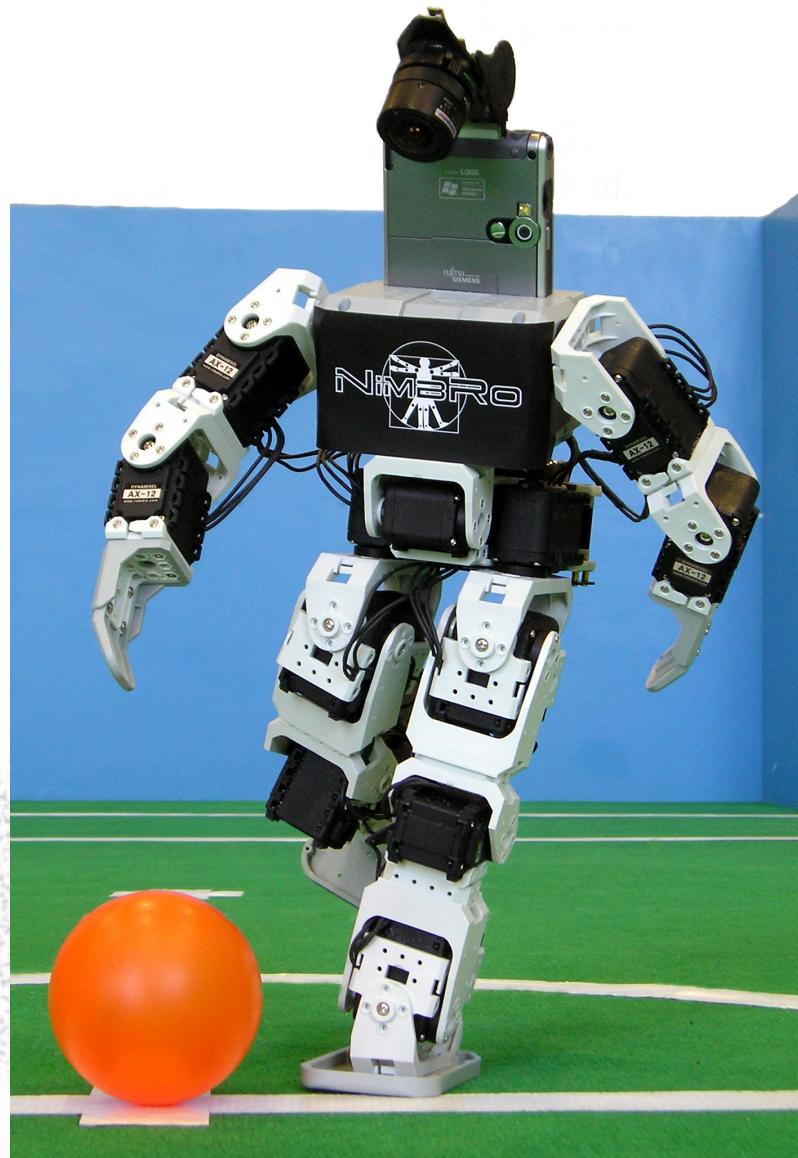
Ejemplo: Ingeniería Naval

Diseño naval

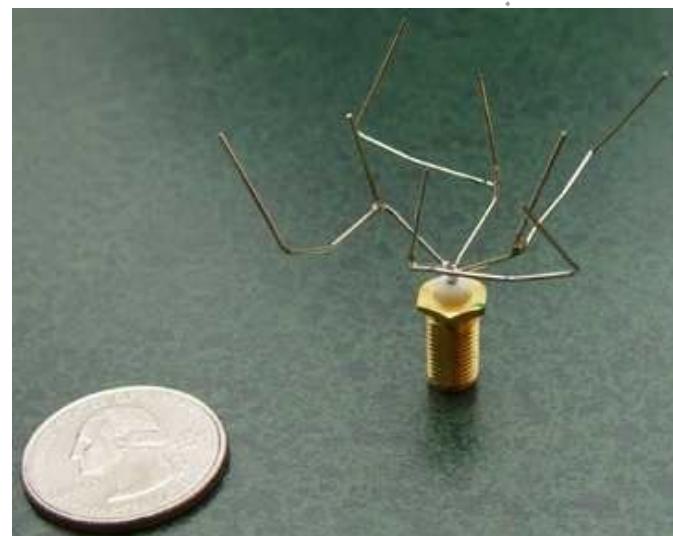
- Determinar parámetros que determinan forma del casco
- **Objetivo:** obtener buena estabilidad, maniobrabilidad, distribución de esfuerzos, velocidad, capacidad de carga, etc.



Ejemplo: Ing. Informática/Electrónica

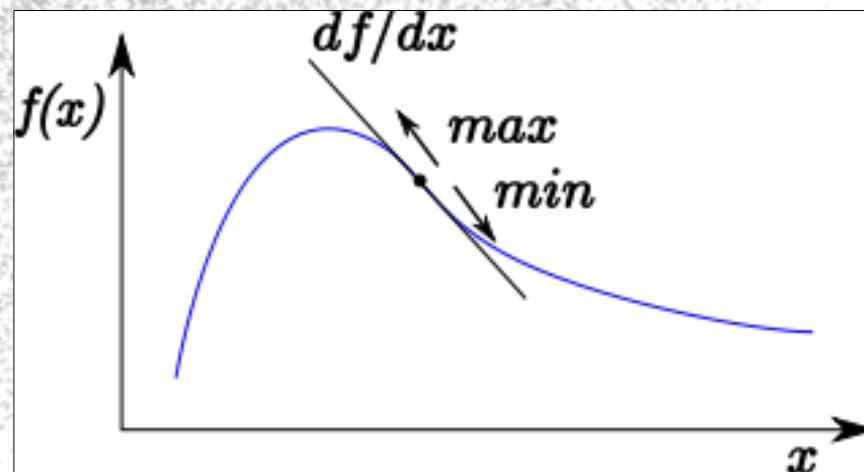


- Diseño de lógica
 - Circuitos
 - Programas
- Diseño de artefactos
 - Antenas



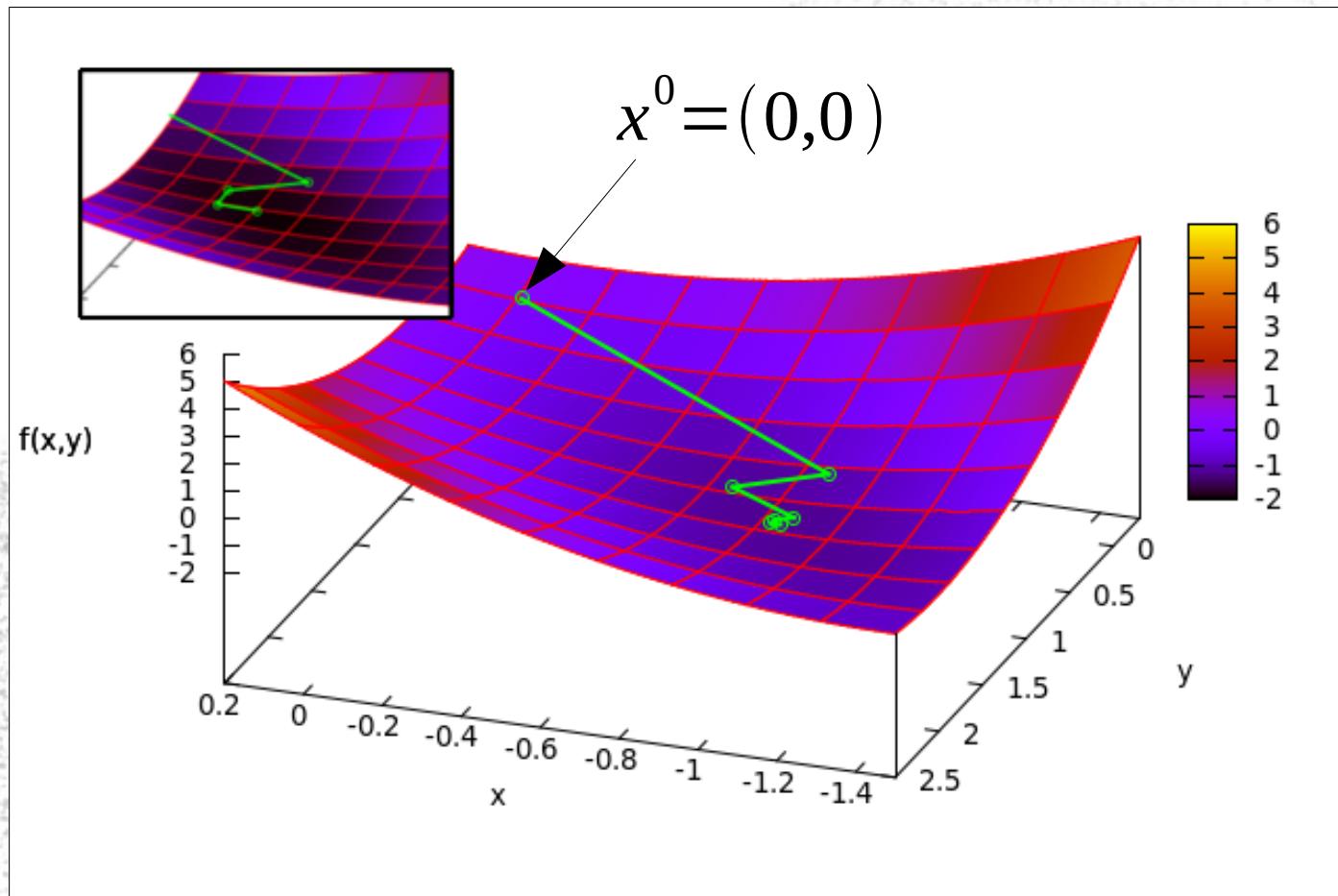
Métodos basados en el gradiente

- **Idea:** encontrar la dirección de mayor cambio (*max/min*) y avanzar en ese sentido
- A partir de un punto cualquiera, iterar hasta que se obtenga una *buen*a solución
- En 1 dimensión:



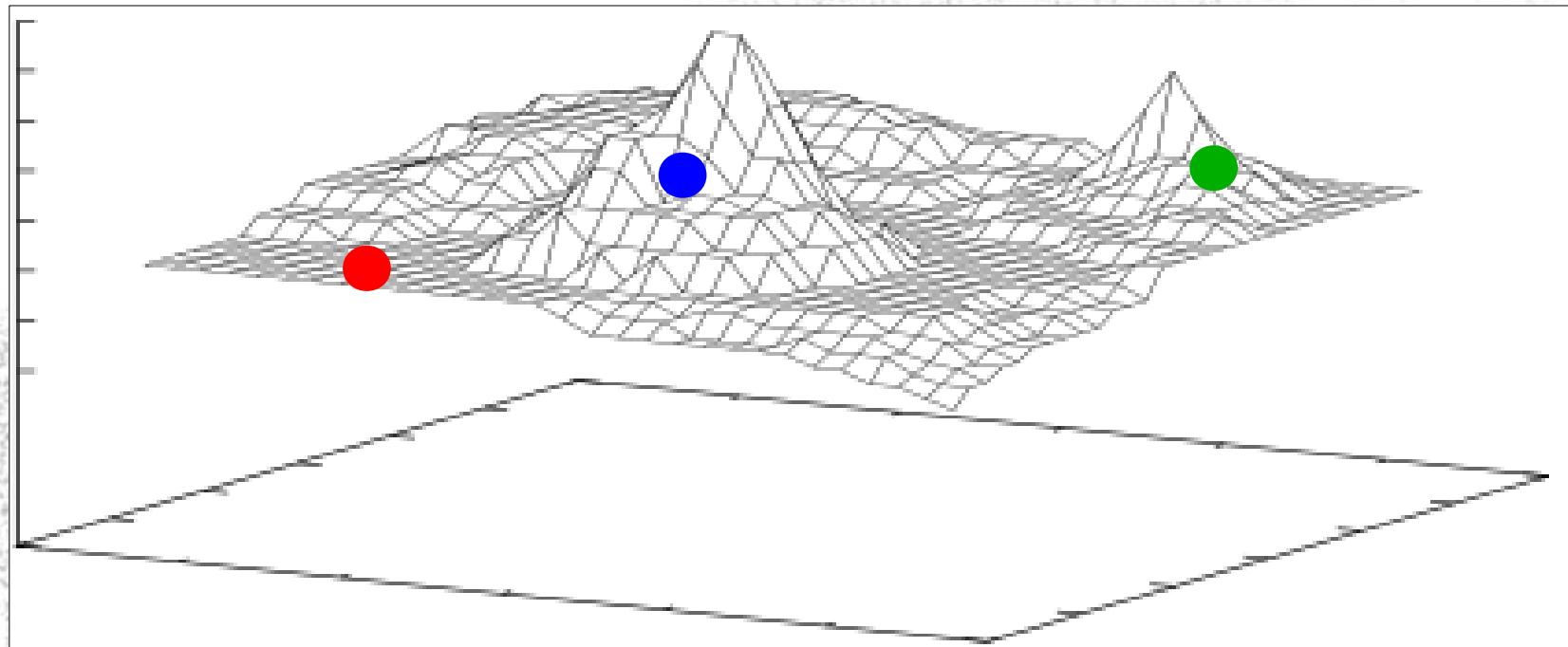
Ejemplo: Método de Cauchy

$$\min f(x, y) = x - y + 2x^2 + 2xy + y^2$$



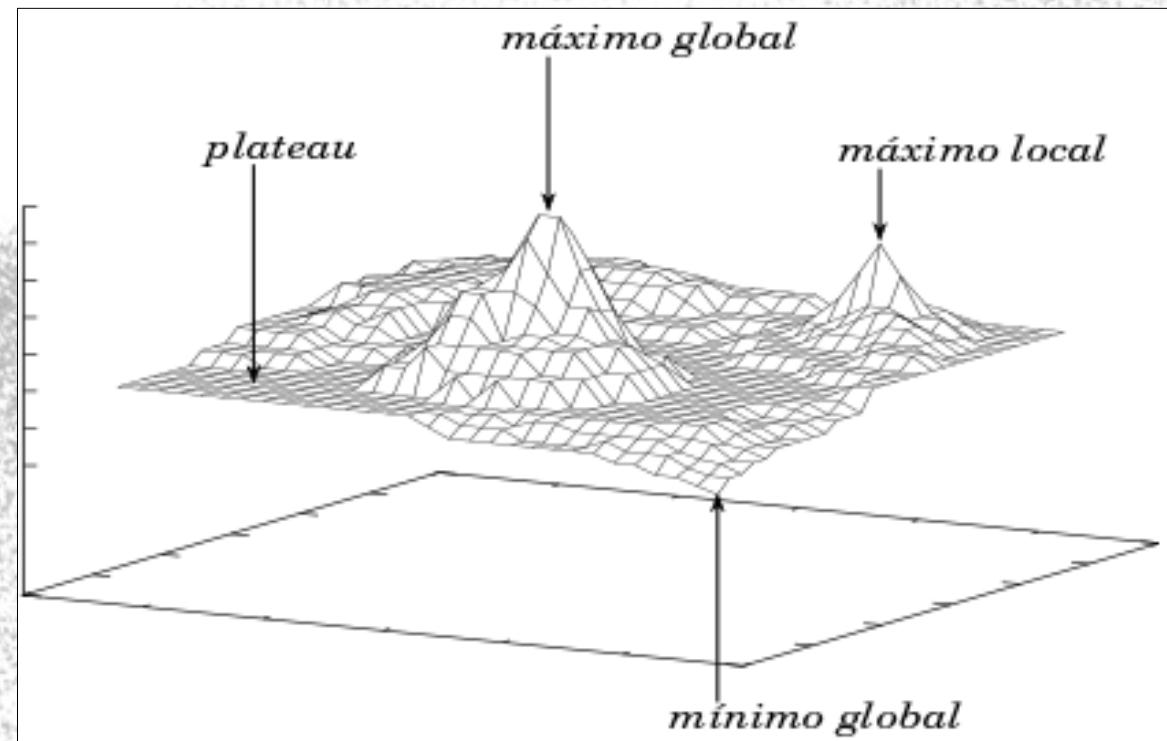
Limitaciones de Métodos de gradiente

- Qué sucede si se aplica un método basado en gradiente a los puntos de la figura?



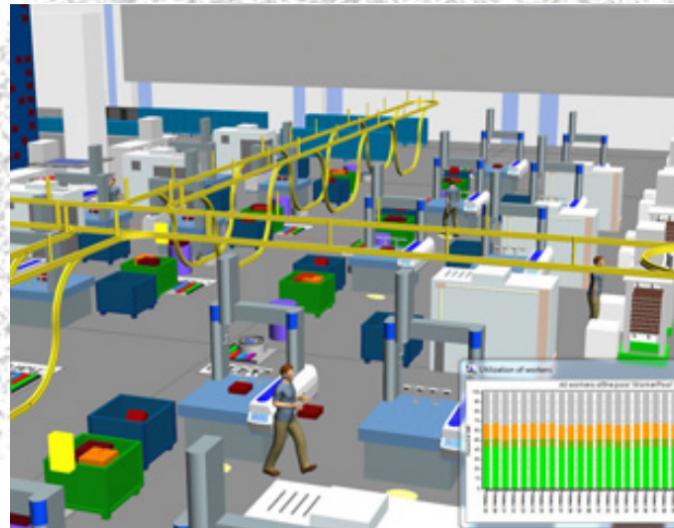
Optimización local y global

- **Espacio de búsqueda:** conjunto de posibles soluciones (dominio de la función objetivo)
- **Vecindad** (de un punto): conjunto de puntos “cercaos” a él en el espacio de búsqueda, definidos por un operador
- **Local vs. Global**

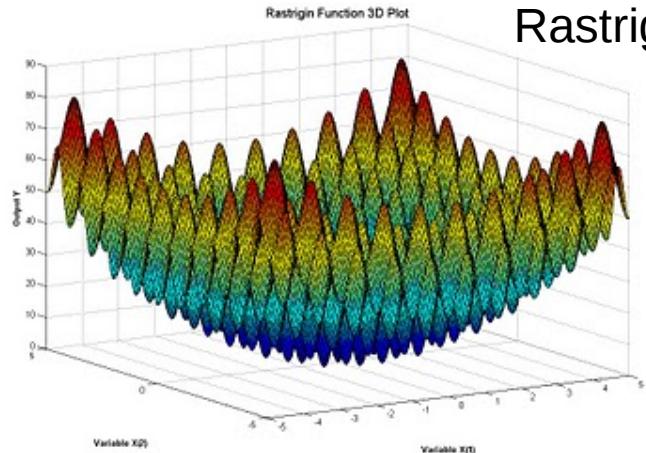


Límites para métodos matemáticos

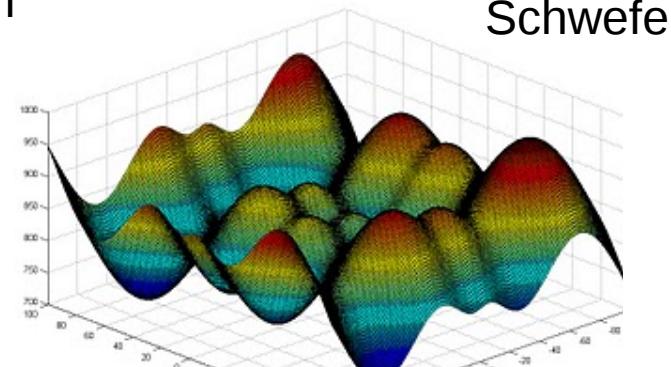
- Aparte de la localidad, normalmente se requiere que exista una función **continua y diferenciable**
- Qué sucede si la función no es continua, no es diferenciable, es desconocida o simplemente no existe?



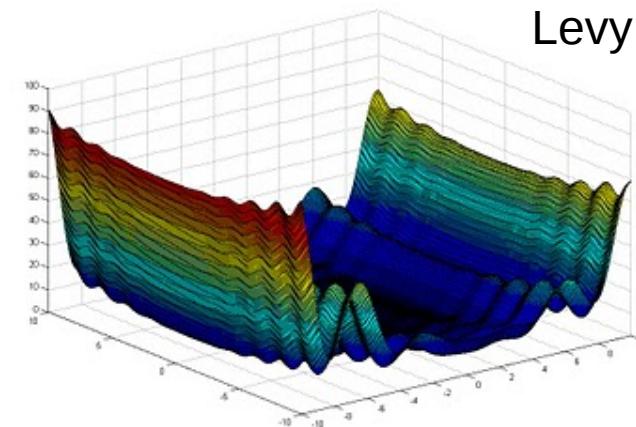
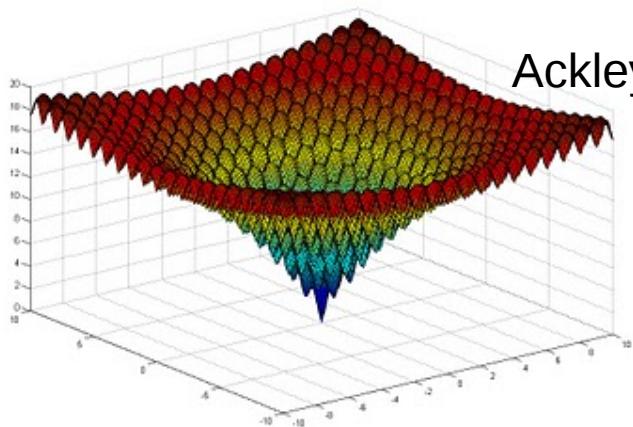
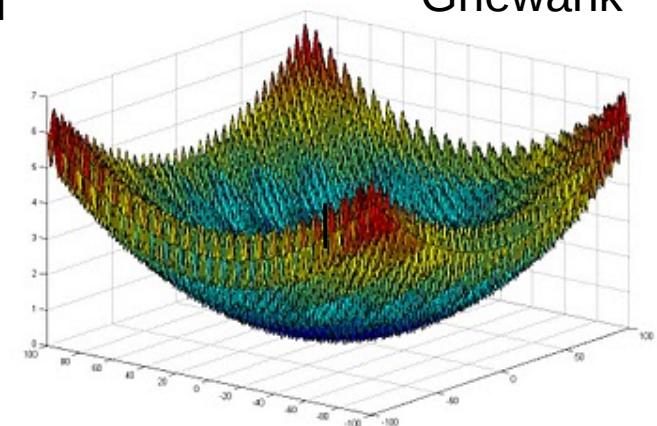
Multimodalidad



Rastrigin



Griewank



Ejemplo: Travelling Salesman Problem

Sean N ciudades de un territorio. El objetivo es encontrar una **ruta** que, comenzando y terminando en una ciudad concreta, pase **una sola vez** por **cada una** de las ciudades y minimice la distancia recorrida por el viajante.

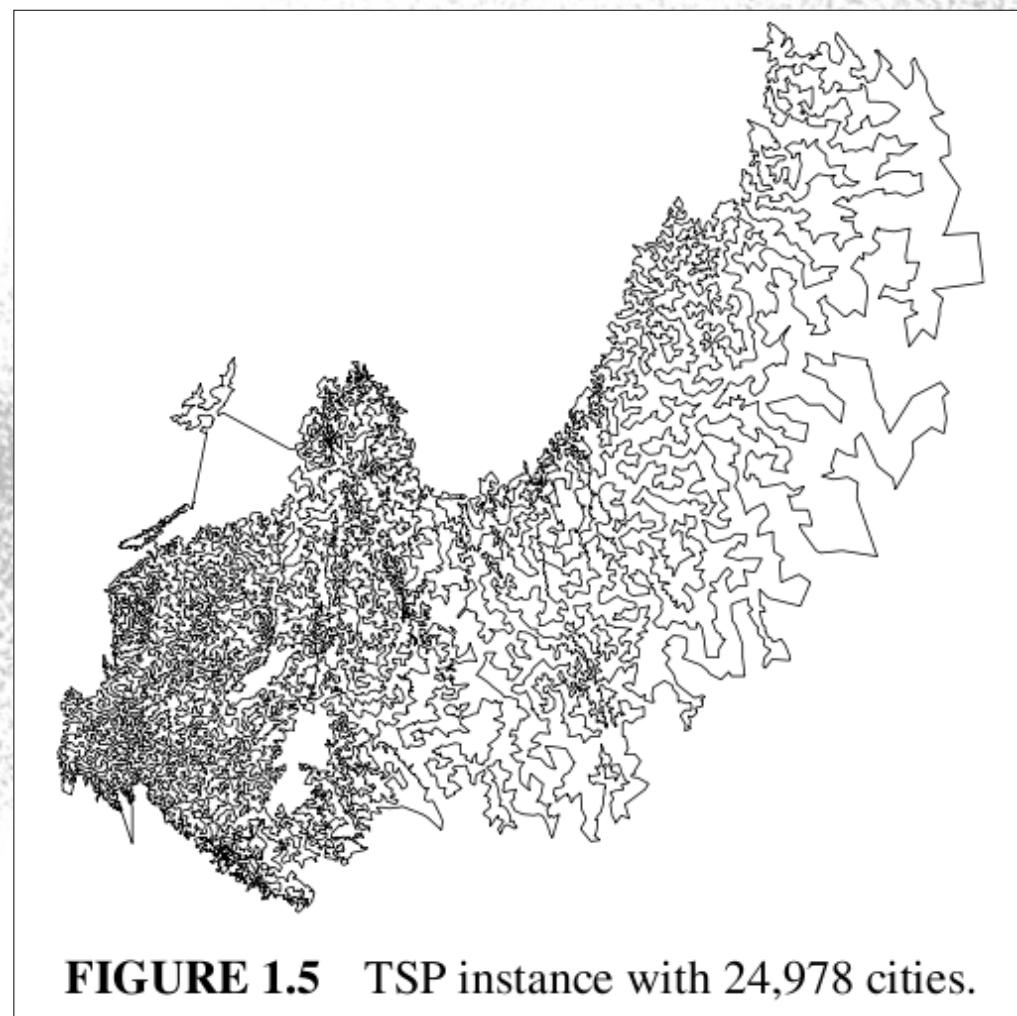


FIGURE 1.5 TSP instance with 24,978 cities.

Tiempo en supercomputador



Ciudades	Itinerarios	PC	Jaguar XT5
18	6.4E+15	24.7 días	32 s
19	1.2E+17	1.4 años	10.1 min
20	2.4E+18	25.7 años	3.4 horas
21	5.1E+19	540 años	3 días
22	1.1E+21	11.880 años	65 días
23	2.5E+22	273 mil años	4.1 años
24	6.2E+23	6.5 millones años	98.4 años

Maldición de la dimensionalidad

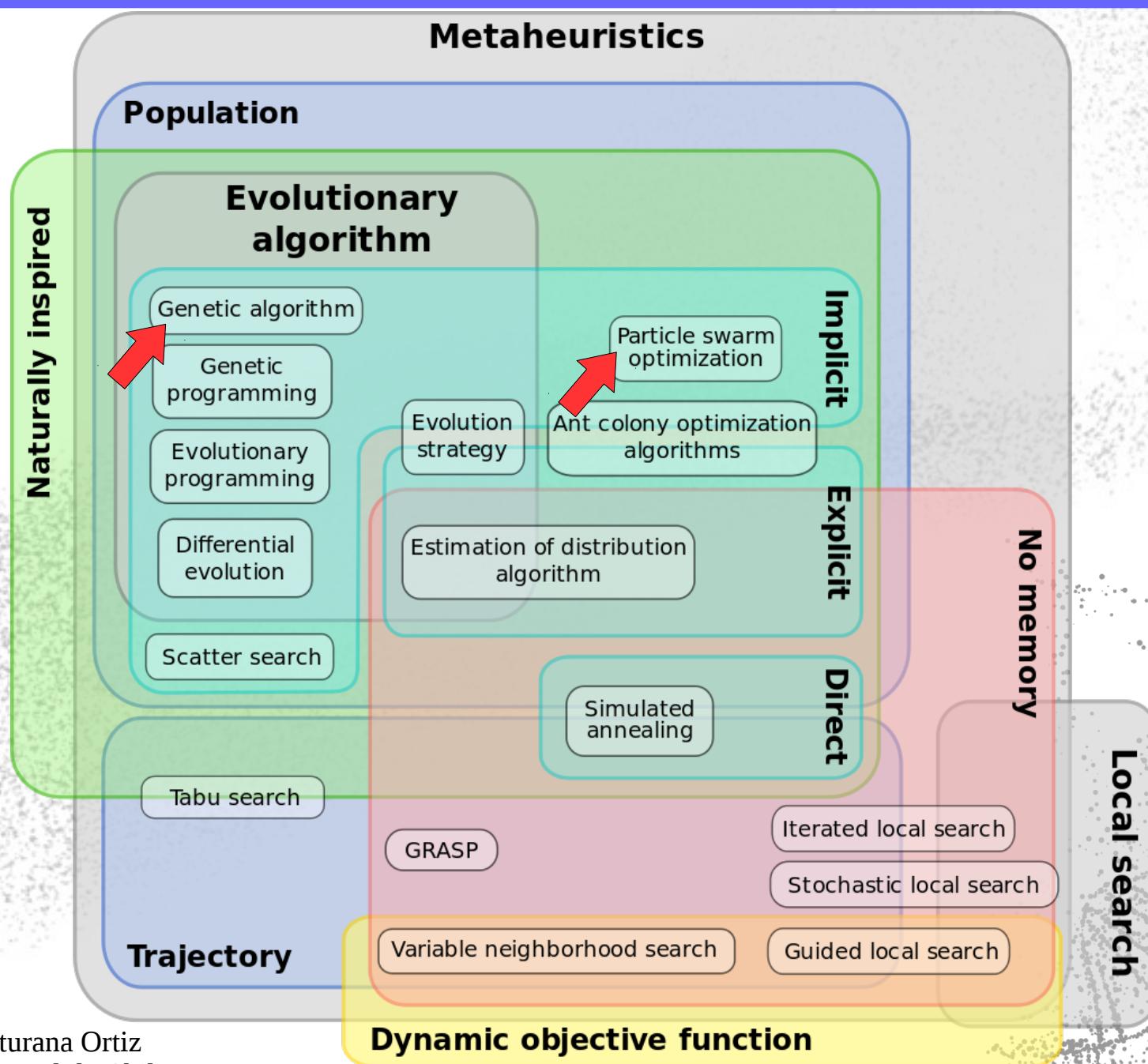
- Si el problema se **escala**, los recursos serán insuficientes
- Propio de sistemas complejos
 - Compuesto de muchos elementos
 - Ej: Problemas combinatorios



Metaheurísticas (MH)

- Son métodos computacionales utilizados en optimización
- Se desenvuelven bien en espacios de búsqueda grandes y complejos
- Exigen pocas condiciones de los espacios de búsqueda y pueden operar con poco conocimiento sobre los problemas
- No aseguran la convergencia hacia el óptimo global
- Muchas veces son estocásticos

Metaheurísticas



PSO: Particle Swarm Optimization

- Se inspira del comportamiento de enjambres
- Un comportamiento complejo emerge de la interacción de varios agentes simples



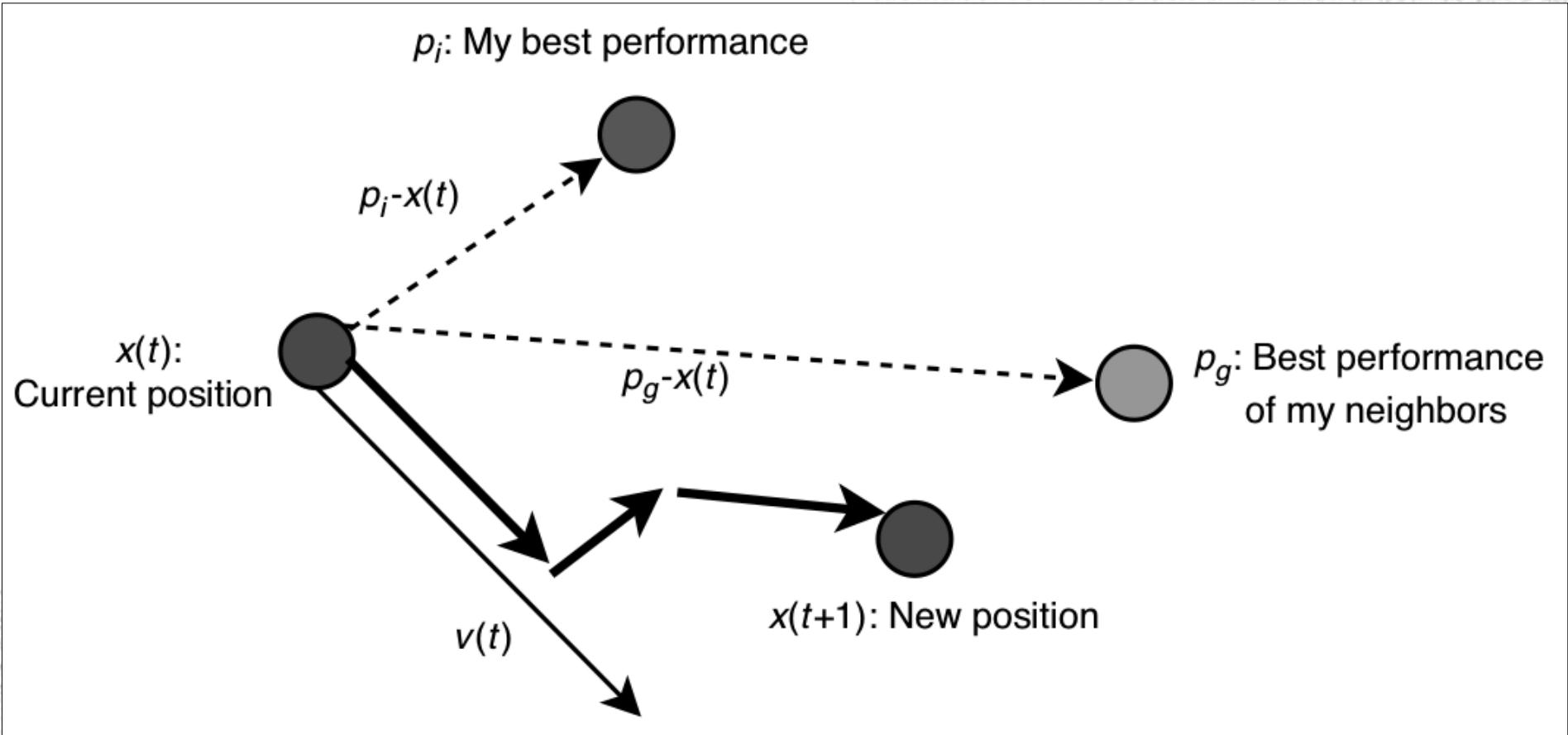
Videos:

- https://www.youtube.com/watch?v=V4f_1_r80RY
- <https://www.youtube.com/watch?v=15B8qN9dre4>

PSO: Particle Swarm Optimization

- Propuesto por Kennedy & Eberhart en 1995
- Un conjunto de partículas se mueven intentando llegar a zonas atractivas del espacio de búsqueda
- El desplazamiento de la partícula depende de:
 - Su inercia (rapidez, dirección)
 - La mejor solución que ha encontrado
 - La mejor solución encontrada globalmente

Movimiento de partículas



Dos formas de actualizar posición

- La actualización de la velocidad puede plantearse de dos formas:
 - Parámetros regulan tradeoff individual/social

$$v_i(t) = v_i(t - 1) + \rho_1 C_1 \times (p_i - x_i(t - 1)) + \rho_2 C_2 \times (p_g - x_i(t - 1))$$

- Parámetro regula inercia vs. Influencia de mejores valores encontrados

$$v_i(t) = w \times v_i(t - 1) + \rho_1 \times (p_i - x_i(t - 1)) + \rho_2 \times (p_g - x_i(t - 1))$$

- Actualización de posición

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

Algoritmo PSO

Algorithm 3.14 Template of the particle swarm optimization algorithm.

Random initialization of the whole swarm ;

Repeat

 Evaluate $f(x_i)$;

For all particles i

 Update velocities:

$$v_i(t) = v_i(t - 1) + \rho_1 \times (p_i - x_i(t - 1)) + \rho_2 \times (p_g - x_i(t - 1));$$

 Move to the new position: $x_i(t) = x_i(t - 1) + v_i(t)$;

If $f(x_i) < f(pbest_i)$ **Then** $pbest_i = x_i$;

If $f(x_i) < f(gbest)$ **Then** $gbest = x_i$;

 Update(x_i, v_i) ;

EndFor

Until Stopping criteria



- Processing permite programar sin mayor esfuerzo despliegues gráficos.
- Programación simple y versátil. Similar a C++
- (ver PSO en Processing)