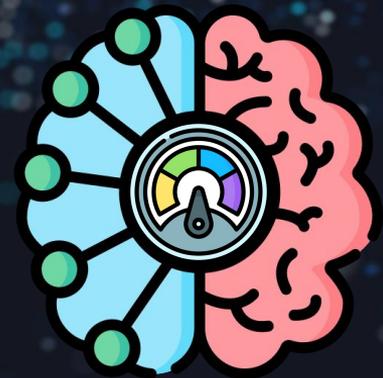


Optimización & Inteligencia Artificial



Optimización & Inteligencia Artificial

- Campos separados
- IA **usa** Optimización
 - **Algoritmo de Descenso de Gradiente** para **Redes Neuronales**
- IA **es** Optimización
 - **Algoritmos de Grafos** para **Caminos Mínimos** en el **Problema del Vendedor Viajante**



Optimización en IA basada en datos

- Entrenamiento = Aprendizaje = Optimización
 - Función de error E
 - Ejemplos
 - Datos de entrada x_i
 - Datos de salida y_i
 - Buscar *parámetros óptimos* en base a *ejemplos* y E

Ejemplos

Entrada	Salida
2	1
5	3.2
7	4.5
9	6

Aprendizaje =
Optimización de E

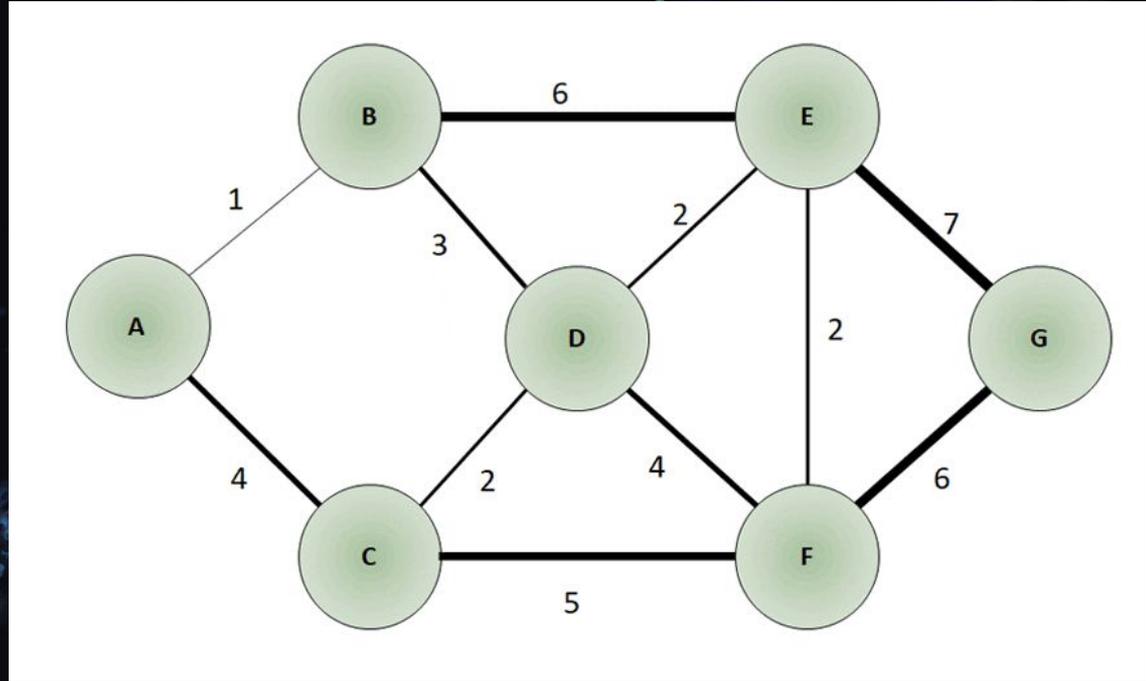


Modelo

*Parámetros
óptimos*

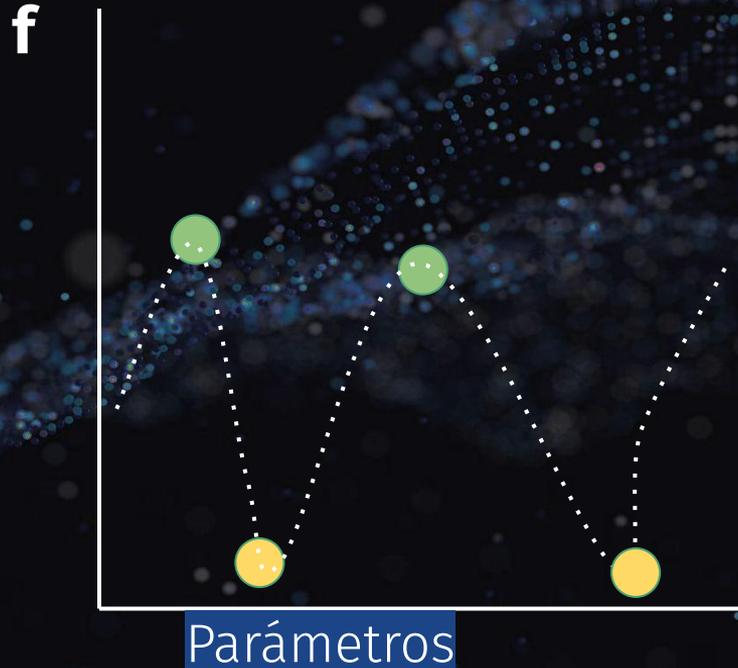
Optimización en IA 'clásica'

- Resolver una tarea
 - = Optimizar una función
- Ejemplo
 - Búsqueda de Caminos Mínimos en un Grafo
 - Algoritmo de Dijkstra
 - Solución exacta



Optimización de funciones

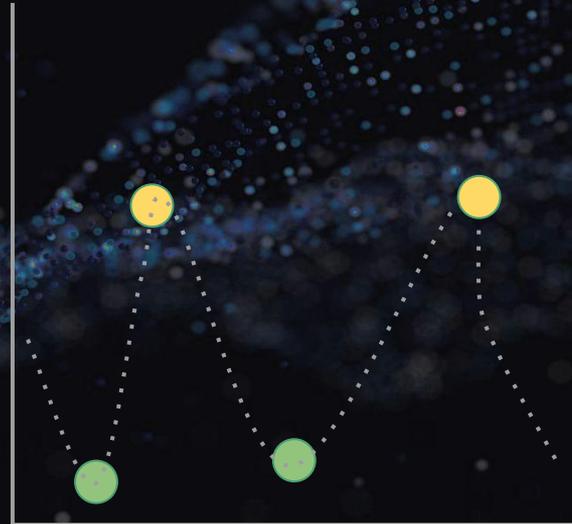
- ¿Qué significa **optimizar** una función f ?
 - Buscar algún **mínimo**
 - Variando sus **parámetros**



Optimización de funciones

- ¿Por qué *minimizar*?
 - Maximizar f es equivalente a minimizar $-f$
 - Puedo buscar **máximos** de f en los **mínimos** de $-f$

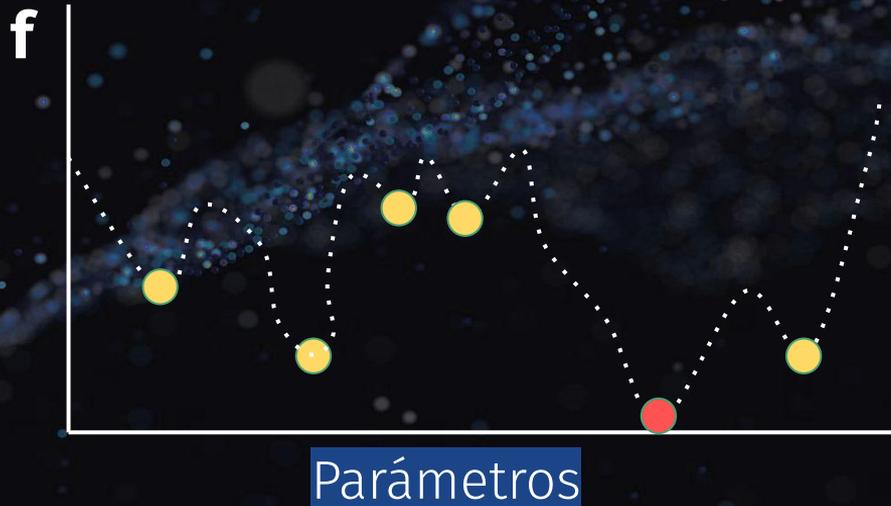
$-f$



Parámetros

Optimización de funciones

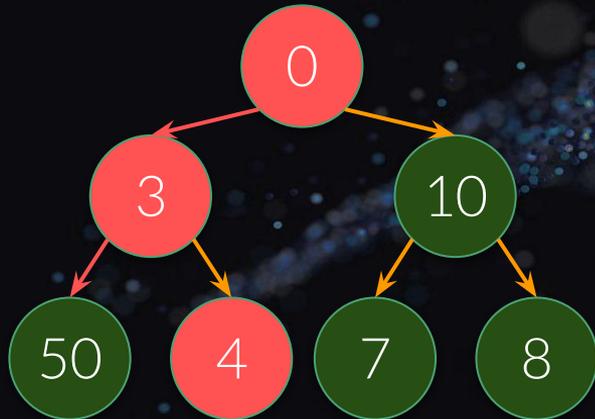
- Mínimos **Locales**
 - Mejor solución en una región
- Mínimos **Globales**
 - Mejor solución entre todos los parámetros



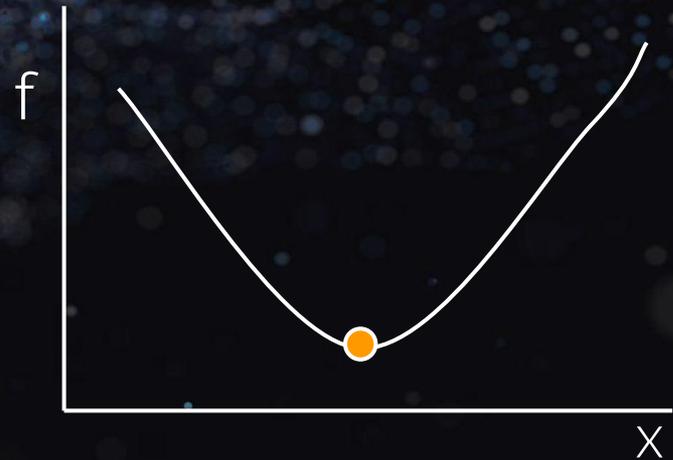
Tipos de Problemas de Optimización

Problemas de Optimización

Variables Discretas



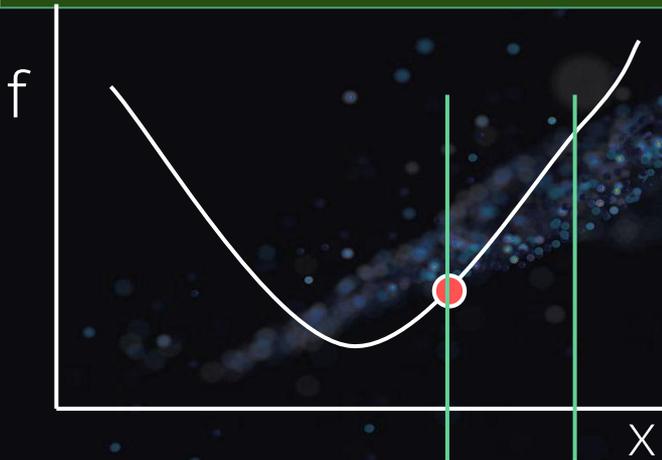
Variables Continuas



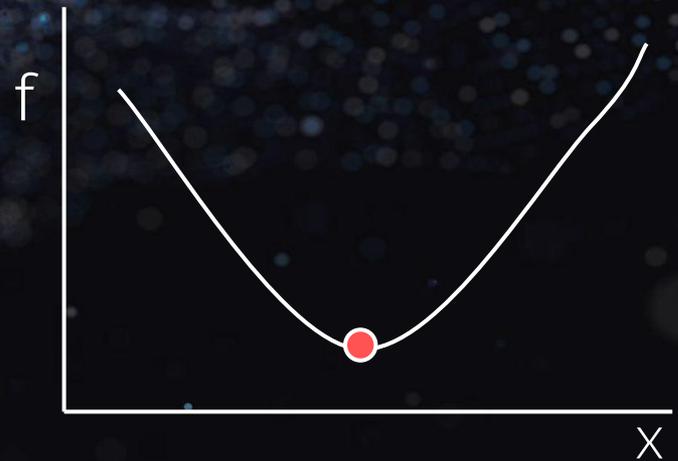
Tipos de Problemas de Optimización

Problemas de Optimización

Con restricciones



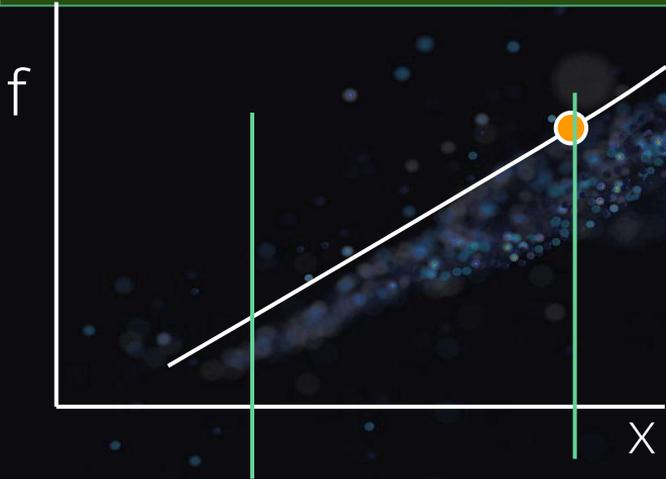
Sin restricciones



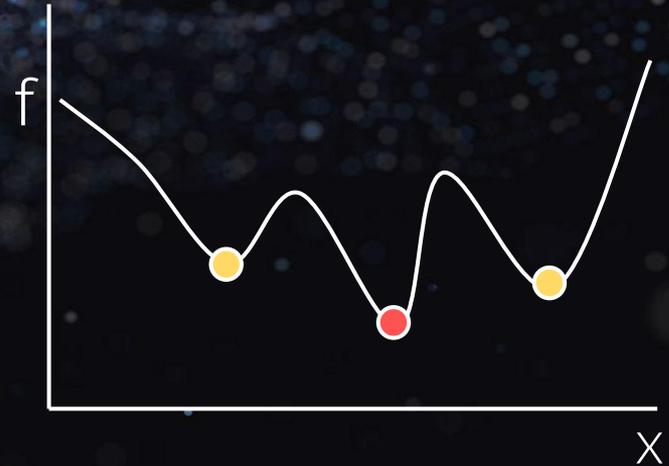
Tipos de Problemas de Optimización

Problemas de Optimización

Lineales



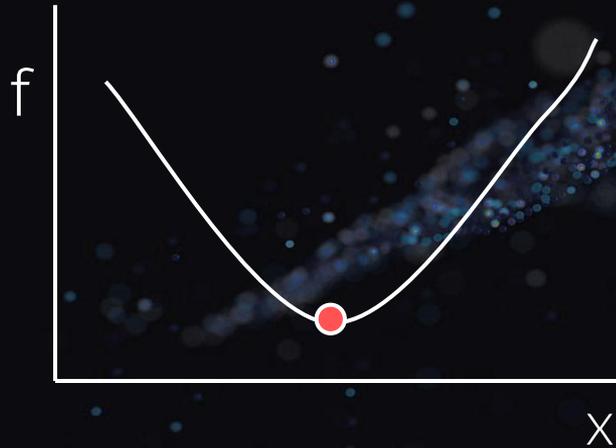
No Lineales



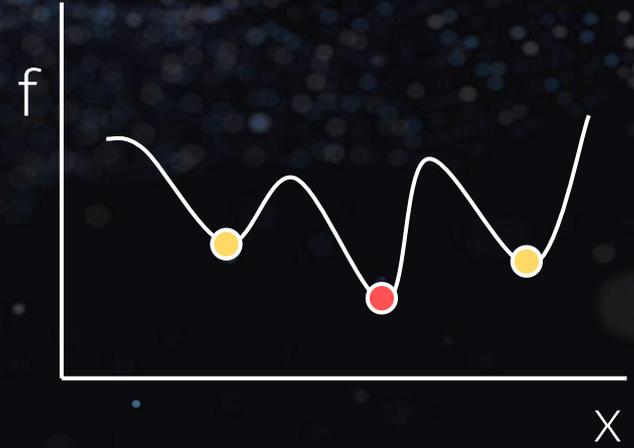
Tipos de Problemas de Optimización

Problemas de Optimización

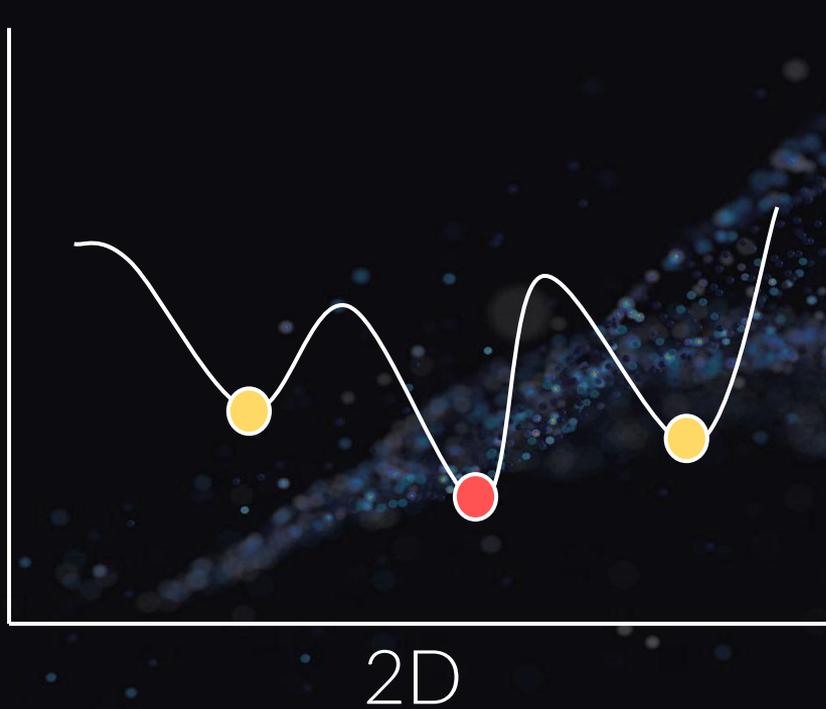
Función convexa



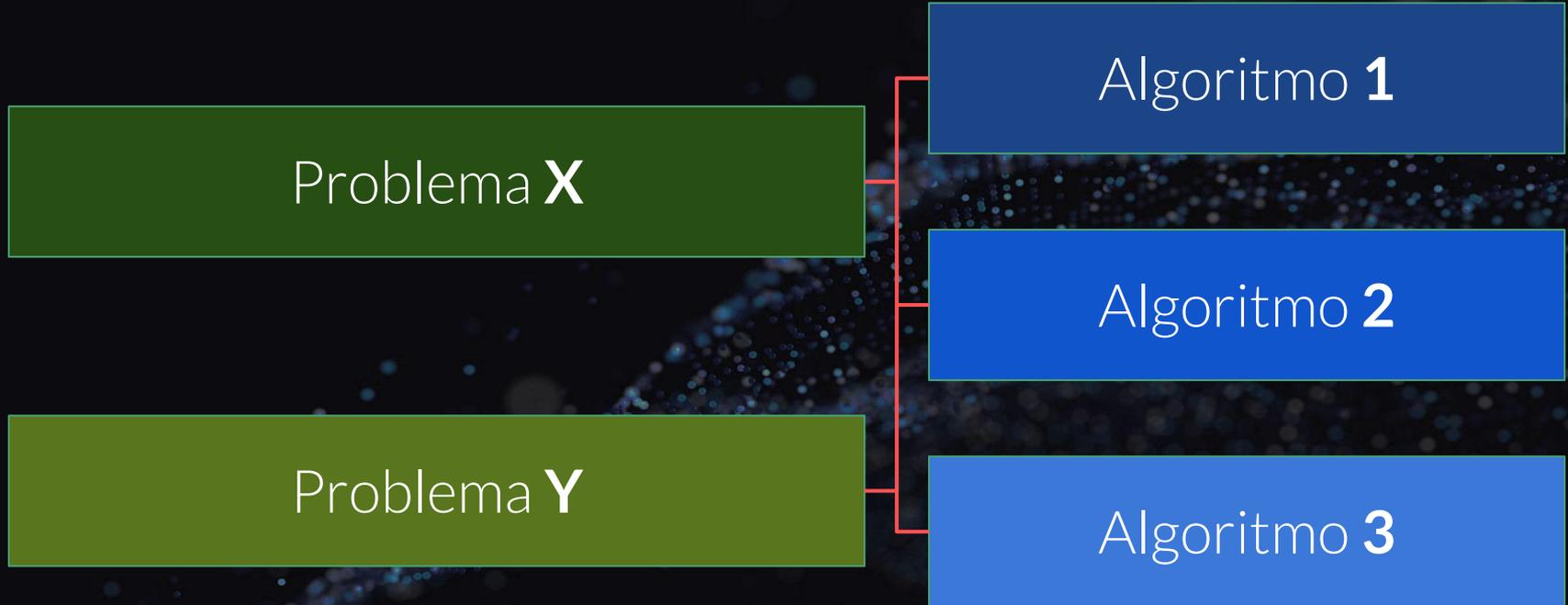
Función no convexa



Alta dimensionalidad



Problemas vs Algoritmos de Optimización



Algoritmos de Optimización



Generales



Especializados

Algoritmos de Optimización Generales



- Pocas asunciones
- Pocas garantías
- Mayor tiempo de ejecución
- Ejemplos
 - Fuerza bruta
 - Búsqueda aleatoria

Optimización con Fuerza Bruta

- Fuerza bruta
 - Probar “todos” los valores
 - Quedarse con el mejor
 - ¿Parámetros continuos?
 - ¿Muchos parámetros?

f



Optimización con Búsqueda Aleatoria

- Búsqueda aleatoria
 - Evaluar valores aleatorios
 - Quedarse con el mejor
 - ¿Cómo generar valores aleatorios?

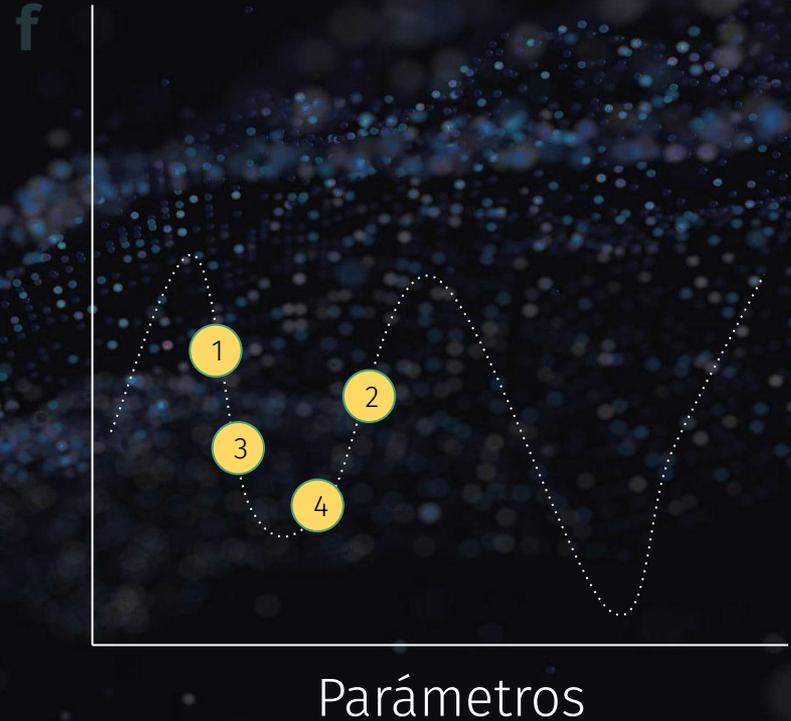
f



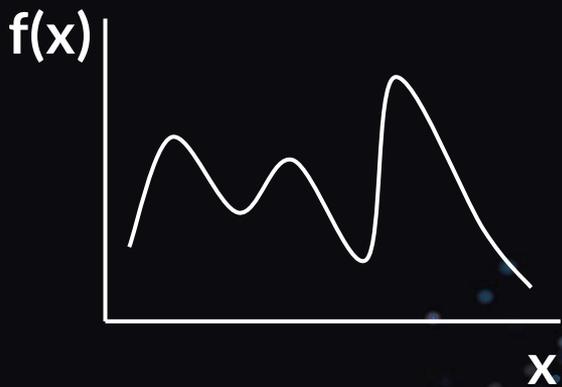
Parámetros

Dificultades de Búsqueda Aleatoria

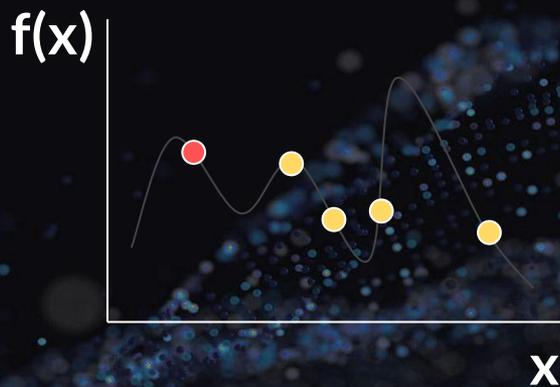
- Coste computacional de evaluar la función de error
- Criterio de convergencia
 - Convergencia = terminar
 - Explotación vs exploración
 - 4 evaluaciones
 - ¿continuar o parar?
 - Varios mínimos



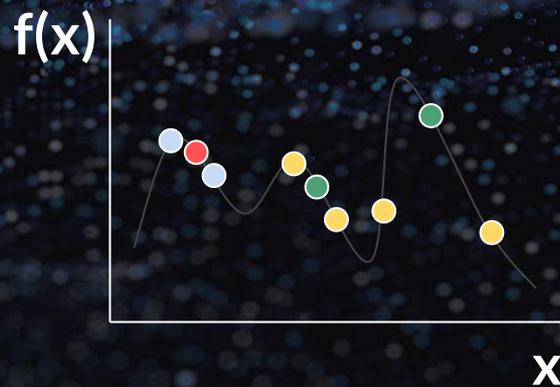
Exploración vs Explotación



Función Completa



Explorado hasta ahora



Explorar?
Explotar?

Algoritmos de Optimización Especializados

- No siempre existen
- No siempre son aplicables
- Pueden ser exactos pero costosos
- Ejemplos
 - Camino mínimo en un grafo
 - Dijkstra, Bellman Ford
 - Optimización de funciones convexas y derivables
 - Descenso de gradiente



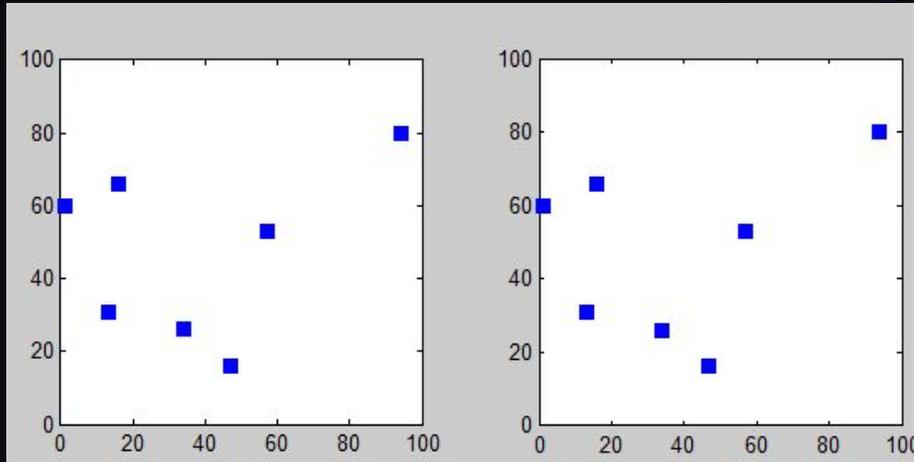
Algoritmos de Optimización Especializados

- Ejemplo: buscar el camino más corto en un grafo
 - Datos: son el grafo **G**
 - Parámetros: el camino **C** a tomar
 - Función de error: Longitud camino **L(C)**
 - Encontrar camino **C ∈ G**
 - Con el menor valor de **L(C)**
 - Solución **única**
 - *Varios* algoritmos **especializados**
 - Dijkstra
 - Bellman-Ford
 - etc..

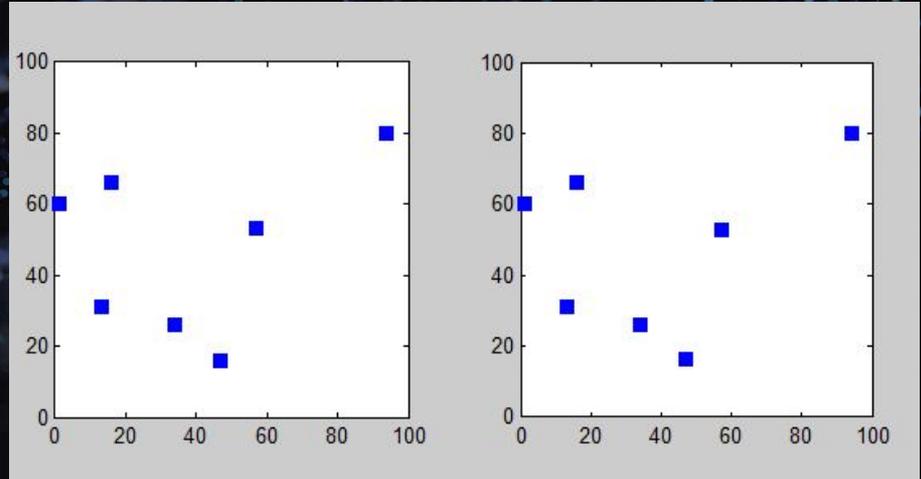


Ejemplo de Algoritmo Especializado

Encontrar la ruta más corta para que un vehículo de reparto entregue paquetes a varios destinos, minimizando el tiempo y el consumo de combustible.



Fuerza Bruta

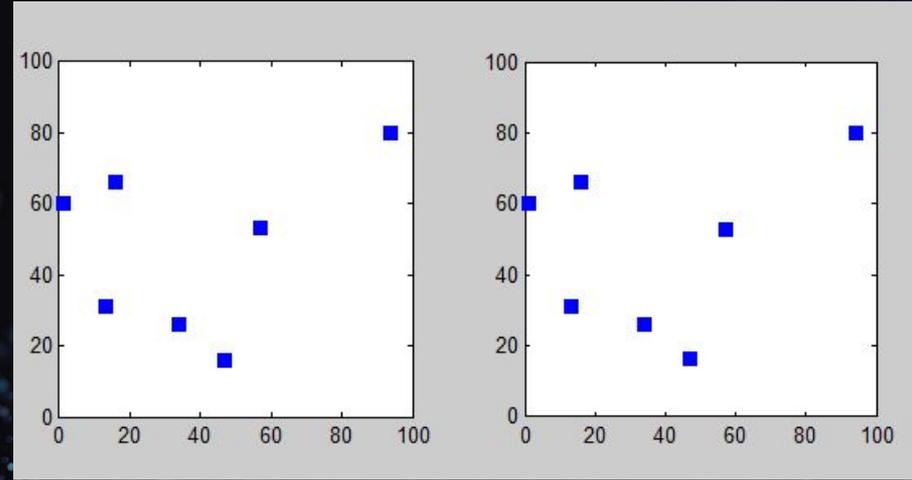


Algoritmo Branch & Bound

Soluciones exactas en Optimización

- Generalmente requieren demasiado cómputo
- No son necesarias en muchos casos

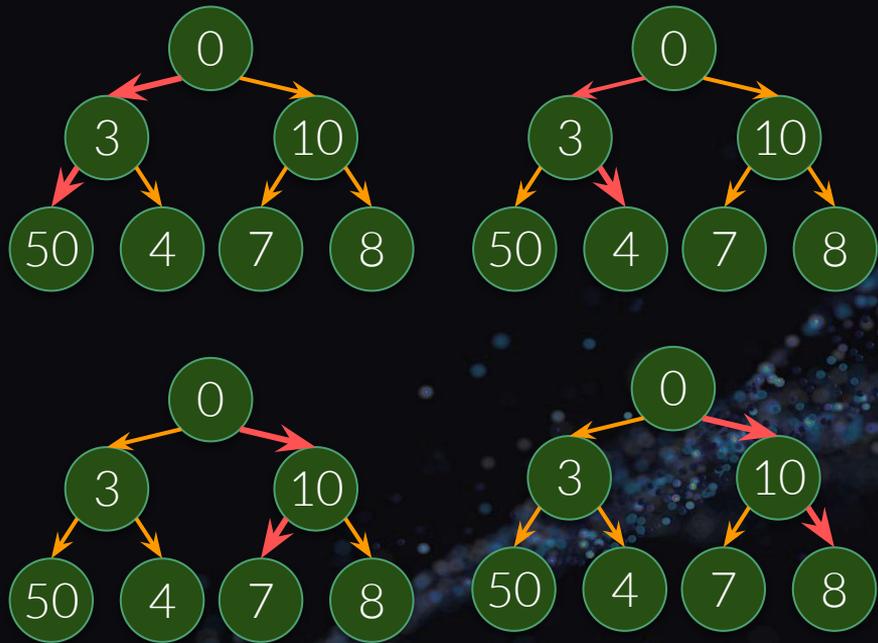
**Algoritmo
Branch &
Bound
(exacto)**



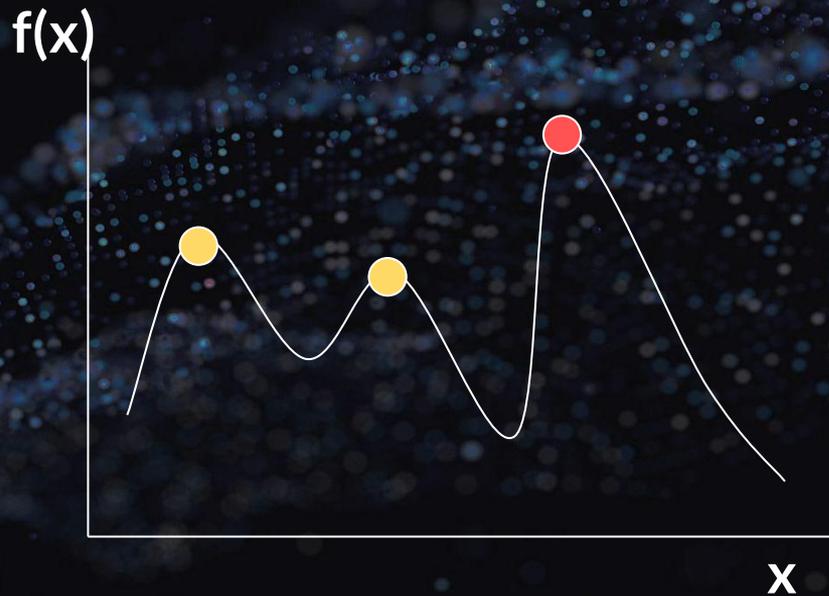
**Ant Colony
Optimization
(aproximado)**

Espacio de Soluciones

- También llamado Espacio de Búsqueda
- Todos los resultados posibles



Discreto
(más difícil)

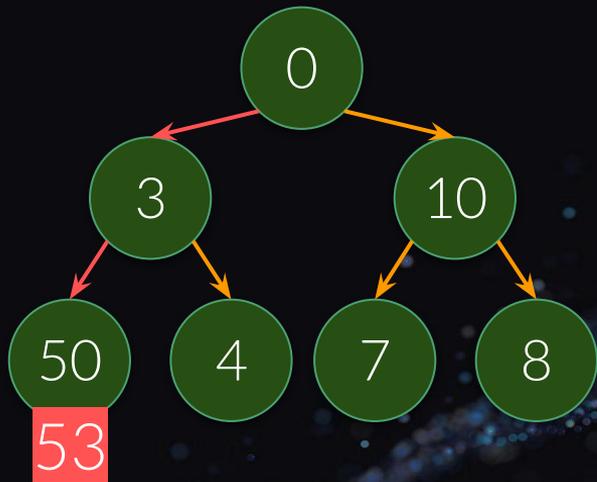


Continuo
(más fácil)

Heurísticas en Optimización

Ejemplo: Encontrar el camino más largo en el grafo

Fuerza Bruta

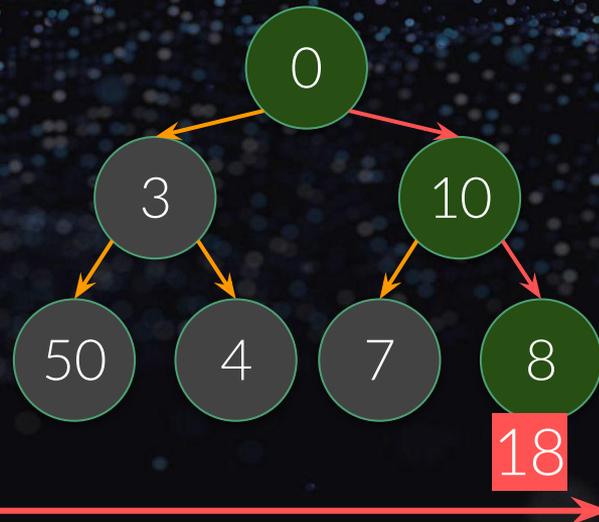


Imposible si N
es grande

Otras **Heurísticas**

??????
.....

Heurística Greedy



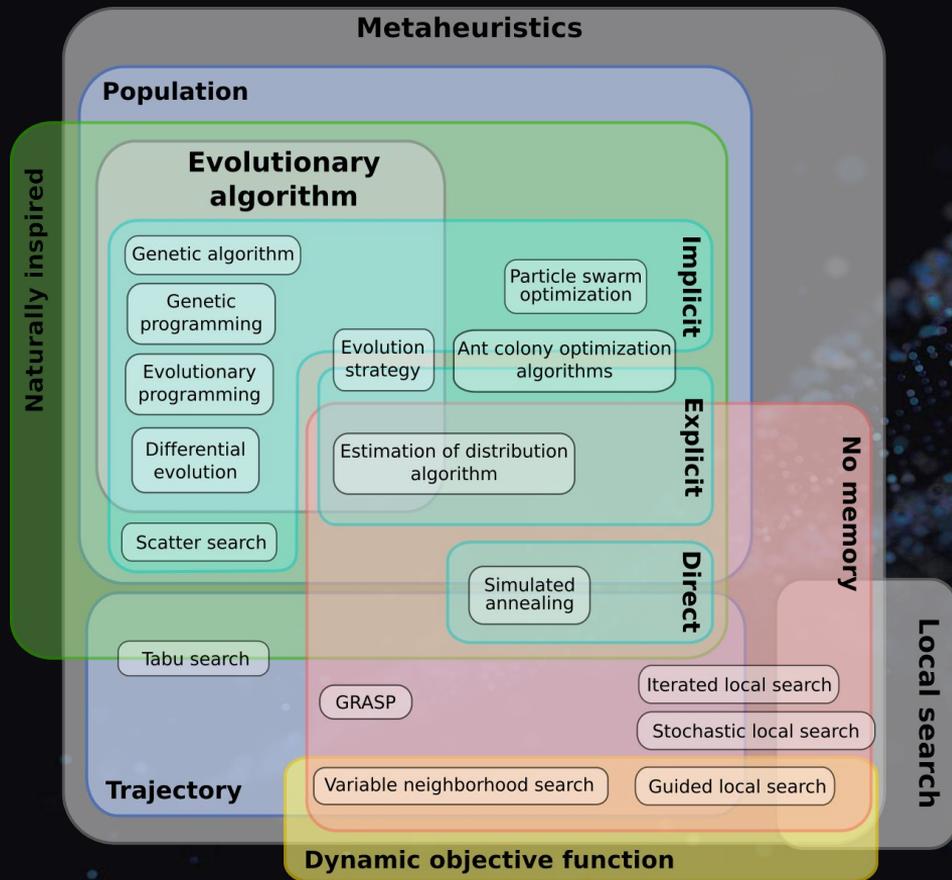
↑ error
↓ cómputo

Metaheurísticas

- **Conjunto** de reglas empíricas
- Reducen espacio de búsqueda
- No garantizan solución óptima
- Generales
- Búsqueda global

Ant Colony Optimization

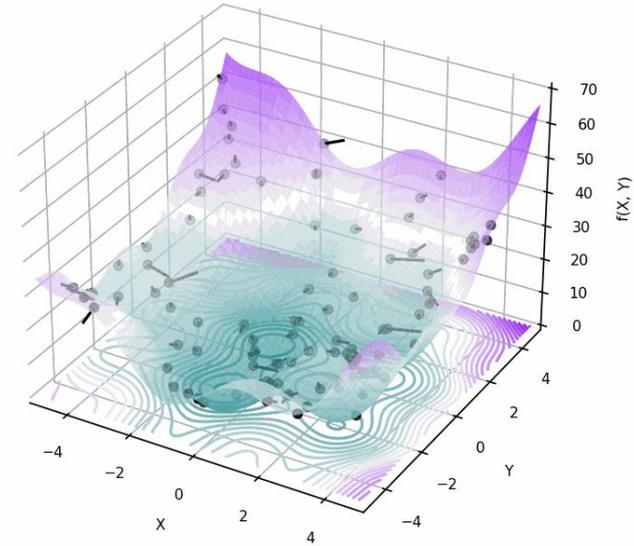
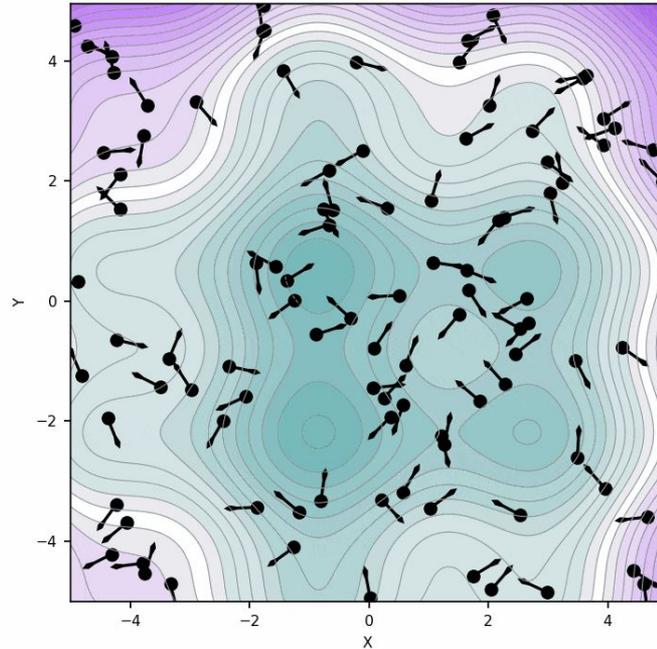
Metaheurísticas



- Distintos tipos
- Muchas basadas en la naturaleza
- Miles de algoritmos diferentes

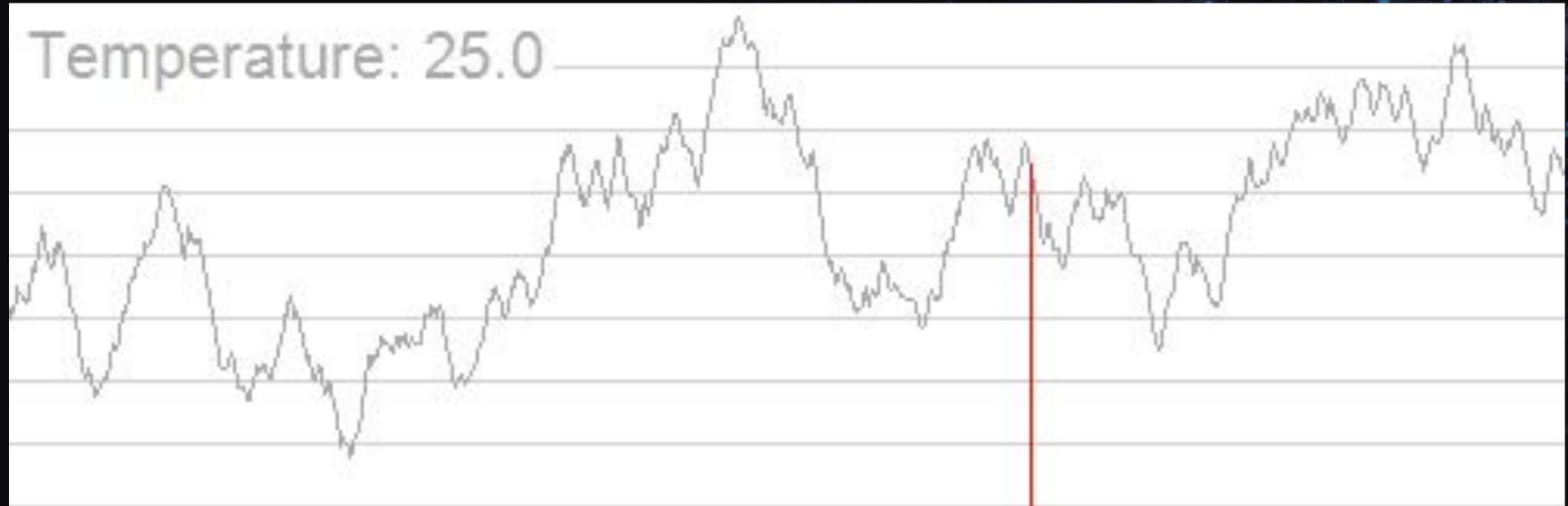
Metaheurísticas de Población

[1/100] $w:0.800 - c_1:2.000 - c_2:2.000$



Particle Swarm Optimization (PSO)

Metaheurísticas de Trayectoria



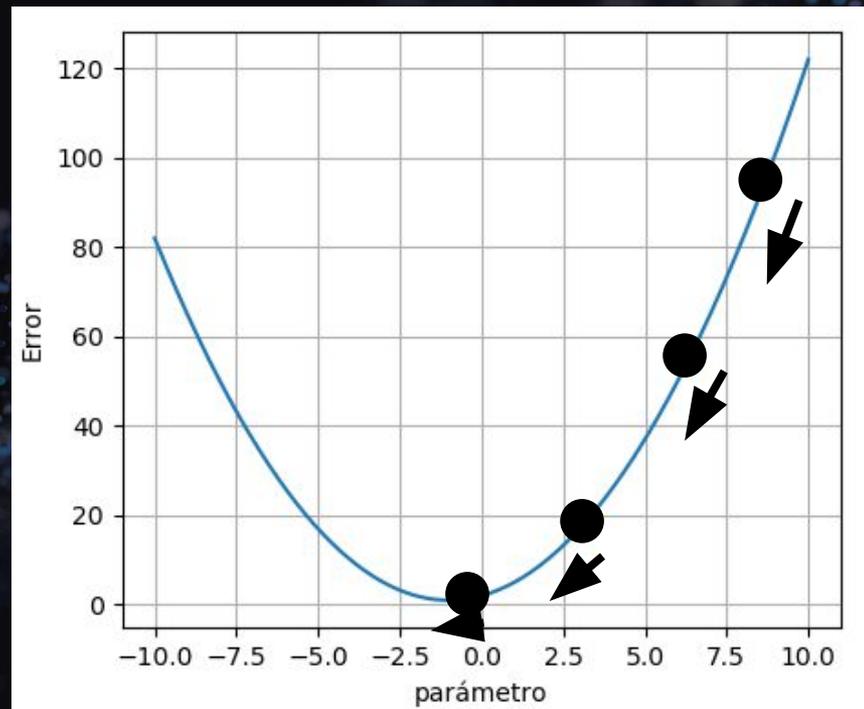
Hill Climbing + Simulated Annealing

Heurísticas vs Metaheurísticas

	Heurística	Metaheurística
Especificidad	Problemas particulares	Problemas generales
Tipo de Búsqueda	Local	Global
Velocidad	Alta	Baja

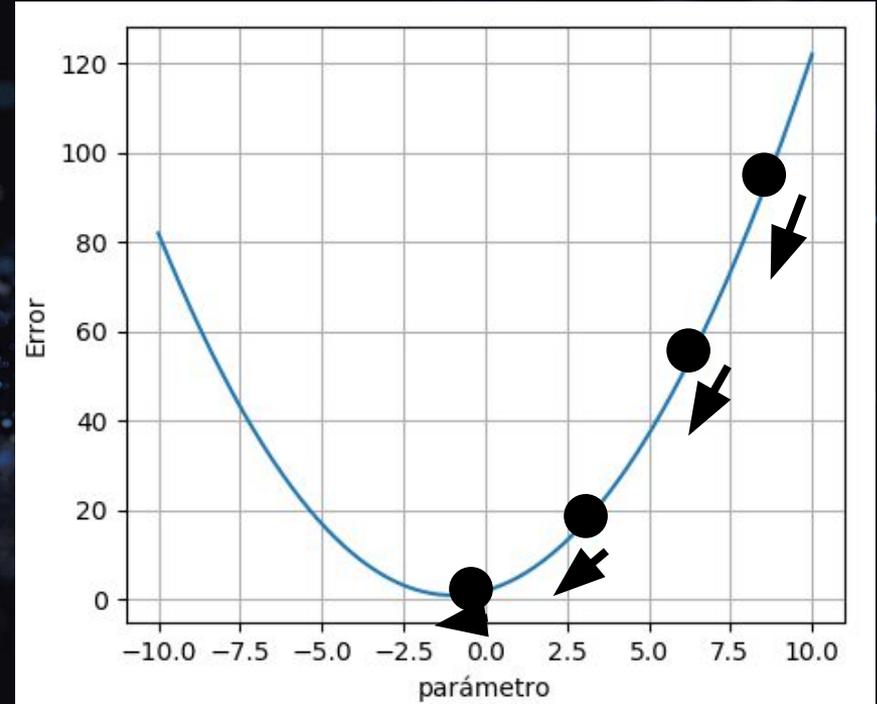
Descenso de Gradiente (DG): Algoritmo especializado

- Asunción: **f** es **derivable**
- Utilizar el **gradiente** o **derivada** para guiar la optimización
 - **Gradiente** en un punto
 - **Dirección** de crecimiento de la función
- Problemas
 - Mínimos locales
- Escalable a $n >$ millones con variante estocástica
 - heurística



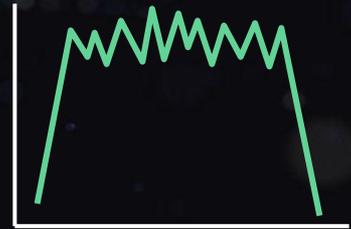
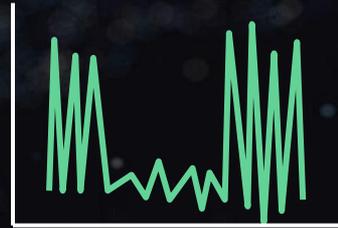
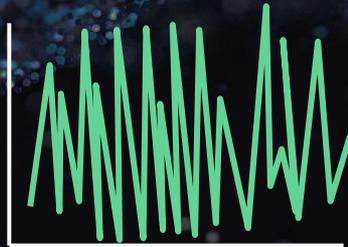
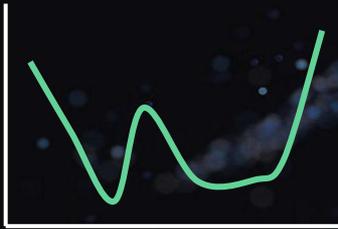
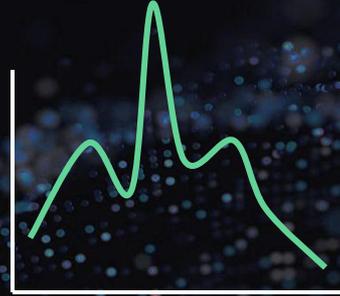
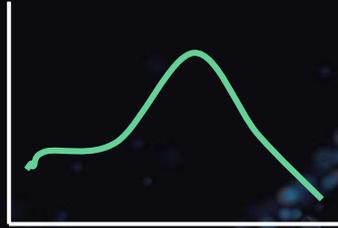
Descenso de gradiente (DG)

- Empezar en un **valor aleatorio** del parámetro
 - Iterar hasta que derivada=0
 - Calcular **gradiente** o **derivada**
 - Indica dirección de maximización
 - Moverse en dirección opuesta
- Valor final: mínimo local (probablemente no global)



Teorema “No Free Lunch” de Optimización

Ningún algoritmo de optimización es mejor que la **búsqueda aleatoria** si promediamos su error en **todos** los **problemas** posibles



Problemas P y NP

NP \approx imposible de resolver si N es grande

