

Introducción a los algoritmos genéticos

Siro Moreno



Evolución Natural



Algoritmos genéticos

- Objetivo: optimizar algo
- Individuo: posible solución

Ejemplo:

- Objetivo: optimizar resistencia aerodinámica



Low-speed ULM (1 m)



Airliner (8 m)



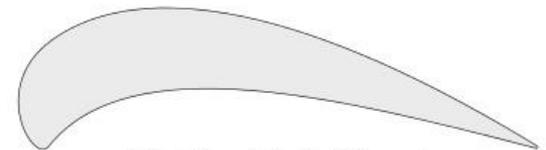
Turbofan fan blade (80 cm)



Propeller blade (15 cm)



Supersonic interceptor (2 m)

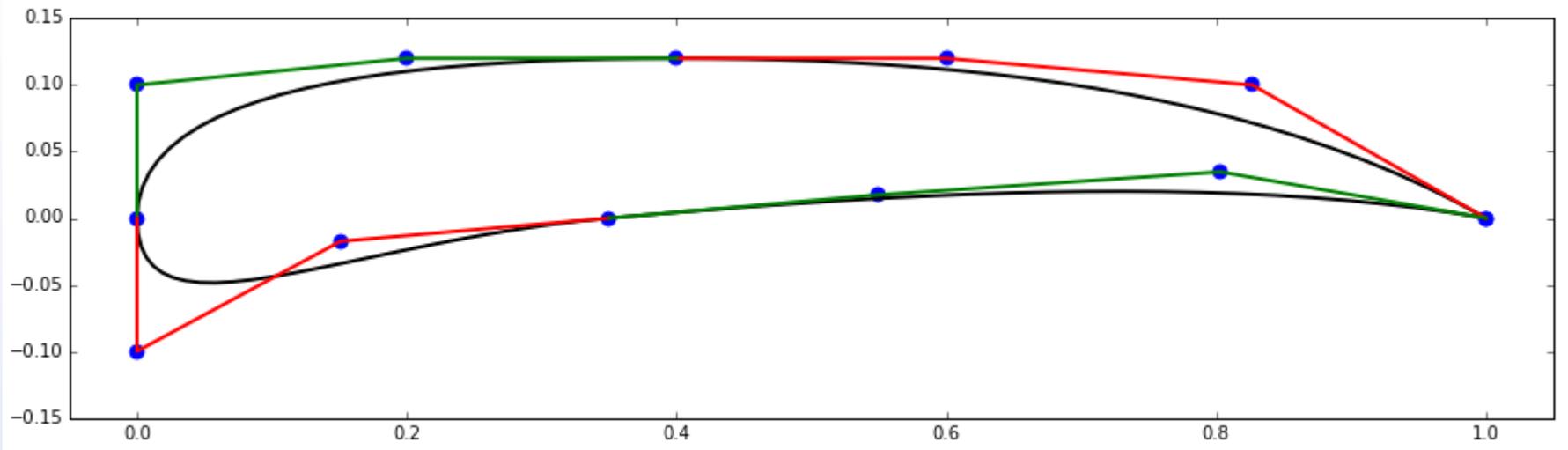


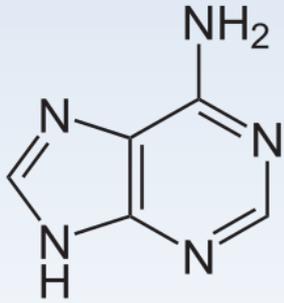
Turbine blade (8 cm)

Individuos

- Conjunto de individuos = población
- Definidos por parámetros.

Ejemplo

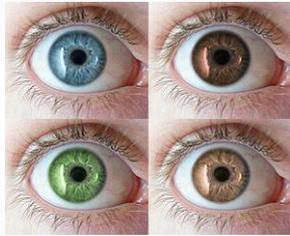




Base Nitrogenada

ATA GTC CCA TGG ATT GTA ACG GCG

Gen



Traits / Características



Performances / Desempeños



Fitness / Aptitud

1

Bit



100110100110101100

Gen

Transcripción



Problema

$x_point_1 = 0.3456$



$aero_lift = f(x, y, z)$

Traits / Características

Performances / Desempeños

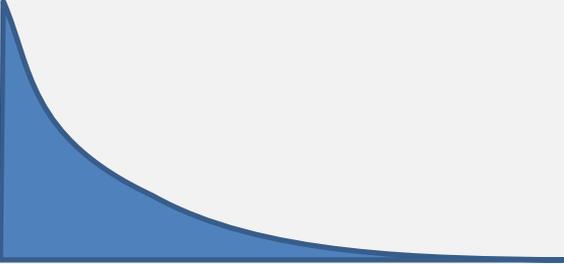
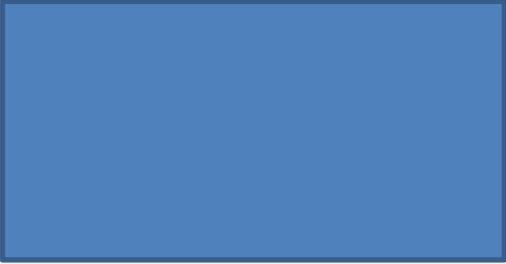
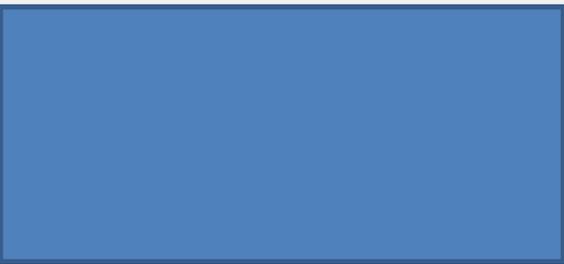
Función de Fitness



$fitness = f(results)$

Fitness / Aptitud

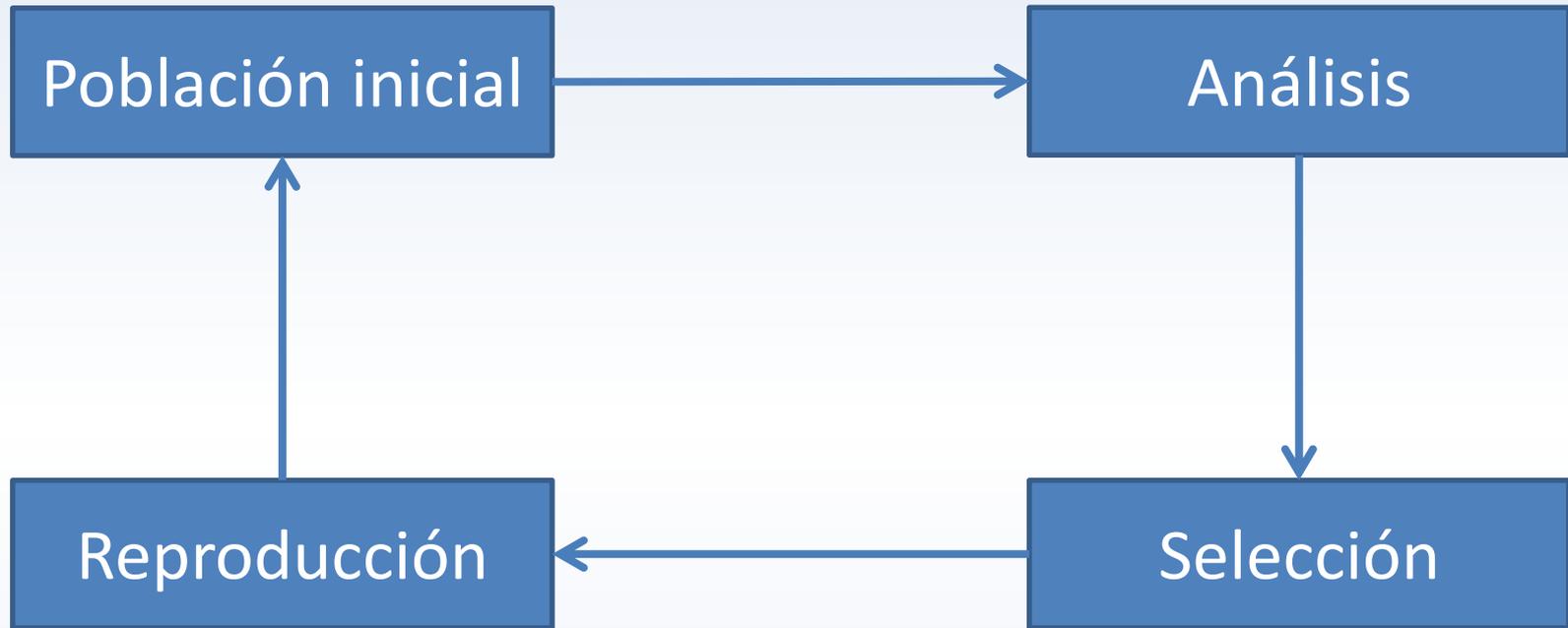
Transcripción

Tipo	Efecto de cambio de 1 bit	Probabilidad de valores
Bits con peso	 A blue shaded area under a curve that starts high on the left and decays exponentially towards the right, representing the effect of a 1-bit change in weighted bits.	 A solid blue rectangle, representing a uniform probability distribution for the values of weighted bits.
Bits sin peso	 A solid blue rectangle, representing a uniform probability distribution for the effect of a 1-bit change in unweighted bits.	 A blue shaded area under a bell-shaped curve, representing a probability distribution for the values of unweighted bits.

Función de Fitness

- La función más importante del algoritmo
- Condensa en un solo valor la calidad de una solución.
- Suele contener condicionales para desechar zonas no interesantes.

Bucle principal: Generación



[José Carlos Cortizo Pérez](#)



[order_242](#)



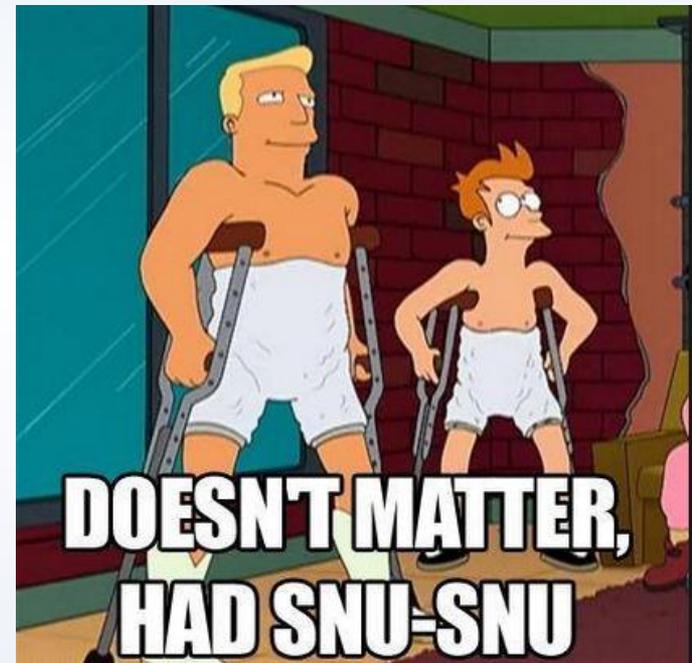
[Grendelkhan](#)

Selección

- Mortalidad diferencial, aleatoria o semi-aleatoria
- Elegir qué individuos mueren y cuáles se reproducen
- Equilibrio:
 - Suficientes plazas para la siguiente generación
 - Pérdida de información

Reproducción

- Reproducción diferencial, aleatoria o semi-aleatoria.
- Genera una población nueva para la siguiente generación.
- 2 fases:
 - Cruzamiento
 - Mutación
- Si se conservan pocos de la generación anterior: Elite Clones



Cruzamiento

- Padre 1 : 10010100101010010101
 - Padre 2 : 10101001100101001100
 - Hijo: 10110100101101001101
-
- Parámetros:
 - Número de puntos de corte
 - Posición de los puntos de corte

Mutación

- Añade variedad al acervo genético (gene pool)
- Permite explorar soluciones nuevas
- Antes de la mutación: 1001010010101
- Después de la mutación: 1011010010001

Apertura y cierre del algoritmo

- Población inicial aleatoria
- Criterio de parada:
 - Número de generaciones
 - Estabilidad



Modular el algoritmo: El dilema exploración-explotación

- Exploración:
 - Buscar soluciones nuevas
 - Escapar de máximos locales
 - Añade ruido
- Explotación
 - Afinar los máximos encontrados
 - Conservarlos
 - Atasca en máximos locales



Foto: [Rob Young](#)

Python



- Fáciles de programar:
 - Individuo: objeto
 - ADN: lista, tupla, np array, etc.
 - Genes, traits, performances: diccionarios
 - Población: lista de objetos
- DEAP (Distributed Evolutionary Algorithms in Python)
 - Google -> “genetic algorithm python”

Muchas gracias

