<u>Programación Evolutiva:</u>
Práctica 1.

Pablo Mac-Veigh
Jorge Sánchez
https://github.com/Pabsilon/PracticasPE

Hemos realizado en Java un programa que busca la solución a varias funciones de distintos tipos y parámetros con algoritmos de selección distintos.

Los valores por defecto son: Población 100, Generaciones 100, Precisión 0.0001, Ruleta, Sin elitismo, Cruce 60, Mutación 5 y Semilla 0.

<u>Modo de uso:</u>
La interfaz es bastante intuitiva:

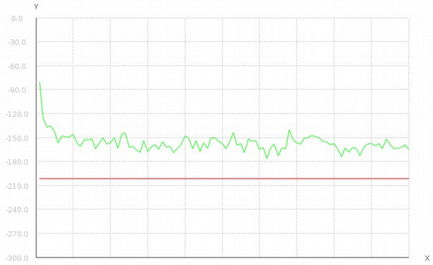
- Un primer JComboBox elige el problema a simular, y si se trata del problema 4 (tanto en la versión con genes como con números reales) se añade un JTextField para introducir el número de parámetros.
- Los tres siguientes JTextField permiten elegir la Población, el número de Generaciones y la Precisión con la que buscamos la solución.
- A continuación tenemos los Métodos de Selección: Un JComboBox permite elegir entre 4 métodos, y si elegimos uno de los torneos, se añade un JTextField para el número de participantes. También tenemos un JCheckBox que nos permite conservar la élite durante la selección de individuos.
- Los dos siguientes JTextField permiten elegir el porcentaje de Cruce y de Mutación.
- El siguiente JTextField permite introducir una semilla numérica (para poder realizar varias veces la misma simulación). Si se introduce el valor '0' se elije una aleatoria (System.Time)
- Por útlimo, se muestra por pantalla la última semilla utilizada en un JTextField, tenemos el botón para lanzar la simulación y un timer que muestra cuanto ha tardado la simulación.
- Al terminar la simulación, se muestran en la gráfica la media de cada generación (verde), el mejor de cada generación (rojo) y el mejor absoluto (azul). Bajo la gráfica se muestra el valor de la función y el/los punto/s que lo han generado.

<u>Problema 1:</u>

$$f(x) = -\left|x.sen(\sqrt{|x|})\right| : x \in [-250,250]$$

Esta función presenta un mínimo de -201,843 en 203,841.

Se trata de una función en la que generando 100 valores aleatorios casi siempre se haya una solución, ya que está en valor absoluto.



Aquí vemos con los valores por defecto que siempre encuentra una solución mejor. Al cambiar los parámetros de cruce, mutación, e incluso la selección, casi todas las gráficas generadas son iguales.

Sin embargo, al reducir la población a 10 y las generaciones a 10 podemos ver progreso: La media va bajando, y se van encontrando mejores valores conforme se avanza en la simulación.

El problema 1 no se trata de un problema muy interesante ya que es una función de un solo parámetro y se converge en una solución muy rápido.

<u>Problema 2:</u>

$$f_2(x,y) = \frac{2186 - (x^2 + y - 11)^2 - (x + y^2 - 7)^2}{2186},$$

$$x, y \in [-6,6].$$

Esta función presenta cuatro máximos idénticos de 1.0 en (3,2), (3.584,-1.848), (-3.779, -3.383), (-2.805, 3.131).

Se trata de una función bastánte mas interesante, al tener dos parámetros de entrada, la solución se complica más que en el problema 1. Pero es polinómica de grado 2 y no debería tener muchos sitios donde se pueda estancar la búsqueda.

Con los valores por defecto, se puede ver que al igual que en el problema 1, se converge a la solución de una manera muy rápida. Por otro lado, al tener dos parámetros, el mejor valor de cada generación no es siempre igual al mejor absoluto.
Se observa una ligera mejora en la media.

En este caso, al utlilizar el sistema de selección por ranking, la media mejora mucho mas rápido que con cualquier otro método de selección, y consigue encontrar la solución en 1.0 mucho mas a menudo que los otros métodos, que suelen encontrar valores del tipo 0.99~

Aun así, los cambios en los parámetros del cruce no influyen mucho en el resultado encontrado. Con 100 generaciones y 100 de población termina hayando la solución (casi) óptima.

<u>Problema 3:</u>

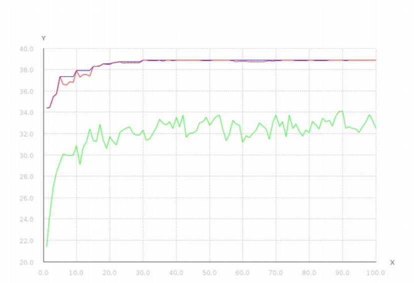
$$f(x,y) = 21.5 + x.sen(4\pi x) + y.sen(20\pi y) \ :$$

$$x \in [-3.0, 12.1] \qquad y \in [4.1, 5.8]$$

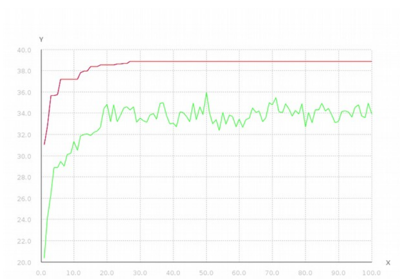
Que presenta un máximo de 38.809 en 11.625 y 5.726. (Según el enunciado)

Nuevamente, esta función es más interesante que la anterior. Se trata igualmente de una función de dos parámetros de entrada, pero al tratarse de una función trigonométrica se pueden encontrar un montón de minimos locales.

Utilizando los valores por defecto encontramos un respetable resultado de 38,4657 con la semilla 1458060129521, por lo que se queda a ~0.4 de encontrar la solución óptima. Vemos una clara mejoría del mejor individuo, pero la media de la población se queda estancada.



A tulizar el torneo probabilístico, hayamos una solución mejor que la puesta en el enunciado: 38,849. Se ve una clara mejora tanto en el mejor absoluto como en la media y el mejor de la generación.



Al utilizar elitismo con selección por ranking y por torneo normal, encontramos muy a menudo el mísmo máximo: 38,850292. Se puede observar que tanto ranking como torneo generan unas gráficas muy similares, pero torneo haya está solución (máxima) mas a menudo que ranking.

Se trata de una función muy interesante donde ruleta no encuentra la solución óptima, sino que lo hacen otros métodos de selección. El elitismo permite encontrar los valores mas altos de la función.