Práctica: 1 GrupoID: L3G07
Alberto Durbey Carrasco
Gerardo Óscar Jiménez Tornos

Memoria de Programación Evolutiva

Índice

Visión general

El algoritmo obtiene soluciones buenas en todas las funciones utilizando los siguientes parámetros:

- Número de generaciones: 100
- Tamaño de la población: 100
- Probabilidad de cruce: 0.8
- Probabilidad de mutación: 0.01

Sin embargo, podemos ajustar estos parámetros en cada función para conseguir un algoritmo más eficiente que utilice menos memoria y menos tiempo.

A continuación se analiza la repercusión de cada uno de los parámetros en el resultado del algoritmo genético. También se comenta la influencia del método de selección.

Variación de la población

La variación de la población nos muestra que con valores pequeños de la misma, esta tarda más en estabilizarse, ya que tanto la selección como el cruce son menores y necesita más generaciones para llegar a un valor máximo.

Variación del número de generaciones

Cuando variamos el número de generaciones podemos observar que las distintas funciones expuestas tienden a tener un número de generaciones más o menos fijo, a partir del cual las sucesivas generaciones no crean un cambio sustancial en el valor óptimo de las mismas.

Variación de la probabilidad de cruce

Variando la probabilidad de cruce podemos llegar a la conclusión de que el cruce ayuda en funciones de optimización donde la magnitud de búsqueda es mayor. Se puede observar que en la función 5 en la que podemos aumentar la variable, obteniendo un rango de búsqueda mayor, una probabilidad de cruce alta nos puede ayudar a llegar antes al máximo.

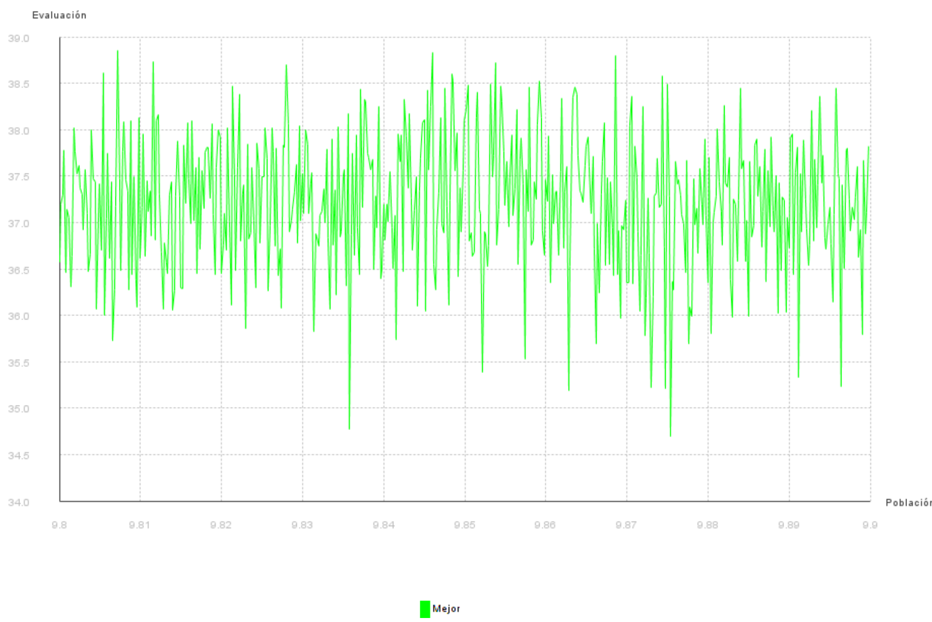
Variación de la probabilidad de mutación

Las mutaciones se hacen presentes en poblaciones pequeñas de individuos, estas pueden proporcionar individuos con los que podemos llegar antes al valor óptimo. Pero en poblaciones grandes y con un número elevado de generaciones las mutaciones aunque sean elevadas no representan un cambio sustancial en las distintas generaciones.

Método de selección

Ambos métodos de selección obtienen resultados buenos para parámetros de población y número de generaciones suficientemente grandes. Sin embargo, la selección mediante torneo consigue valores más aproximados a la solución óptima.

La gráfica anterior corresponde al mismo algoritmo genético ejecutado varias veces sin variar los parámetros y con el método de selección de ruleta. Sin embargo, si utilizamos selección por torneo los mejores cromosomas de cada generación presentan valores más homogéneos (gráfica siguiente).

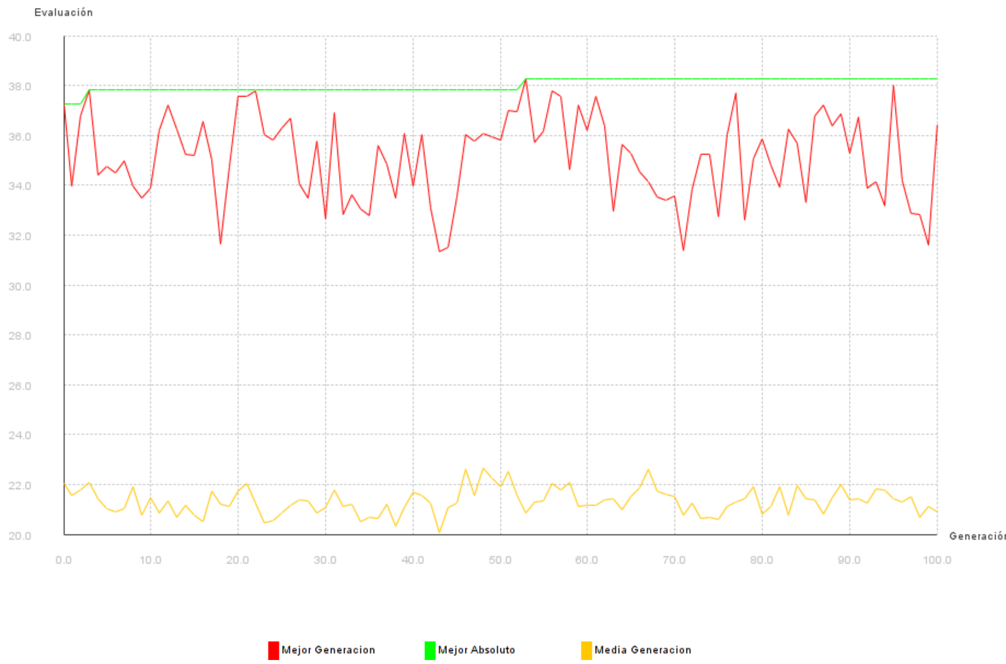


Elitismo

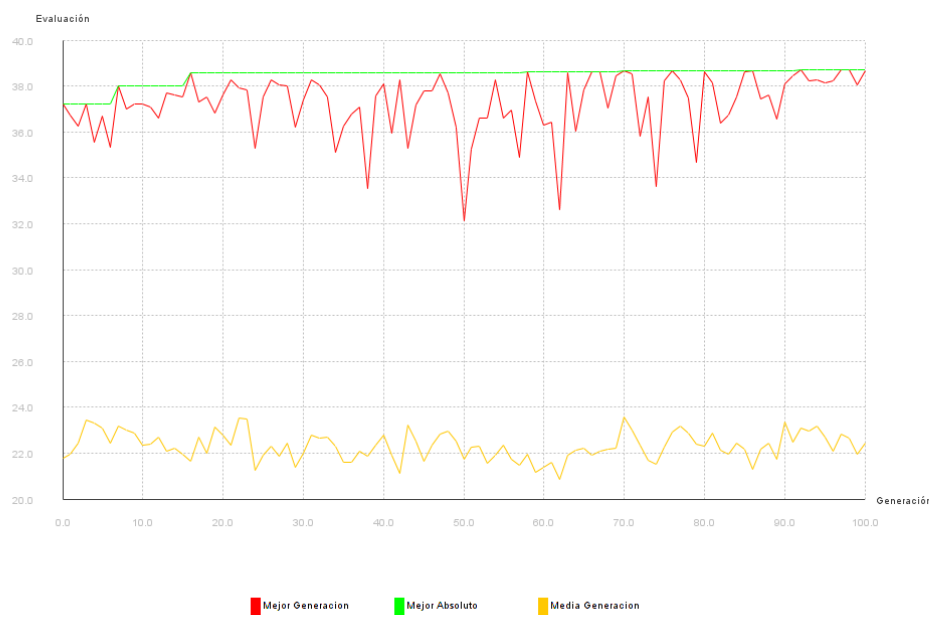
El elitismo fuerza la convergencia hacia la solución. Esto se observa en la curva de la media que tiende hacia la del mejor. Con élites del 3% de la población se consguen los resultados óptimos para la mayoría de las funciones.

Si una función tiene muchos óptimos locales, es posible con élites grandes la población converja al óptimo local y no llege a alcanzar la solución óptima.

Ejemplo de la función 2 sin elitismo que presenta un máximo de 38.2857001972215 (1.3% de error relativo) con X_0 = 12.093548583984374 y X_1 = 5.6252685546875:



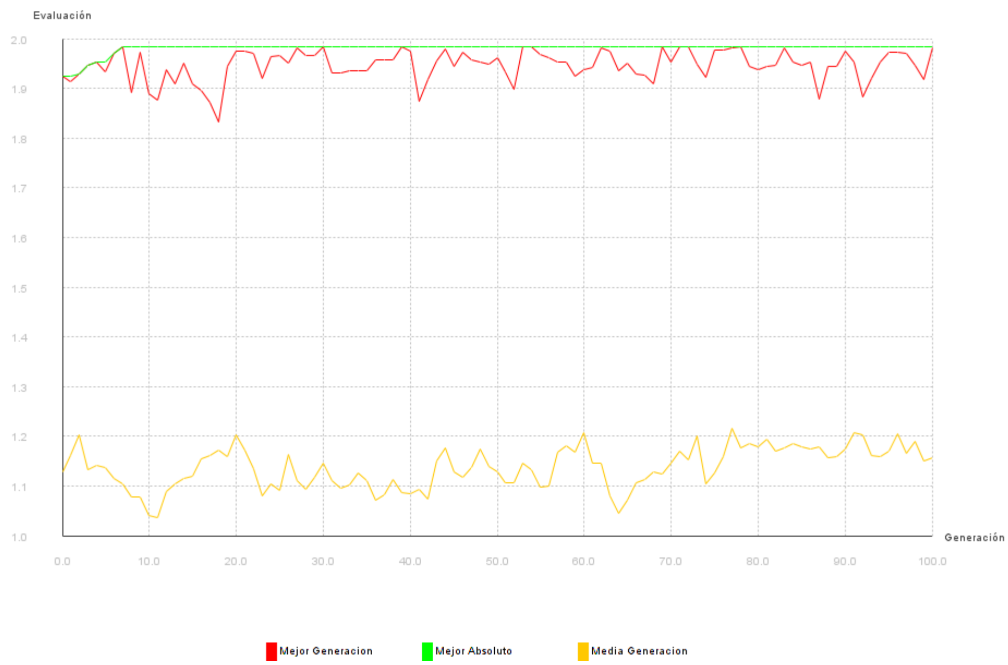
Ejempo de la función 2 con elitismo que presenta un máximo de 38.6878131178406 (0.3% de error relativo) con X_0 = 12.099078369140624 y X_1 = 5.724981689453125:



Análisis de las funciones a optimizar

Función 1

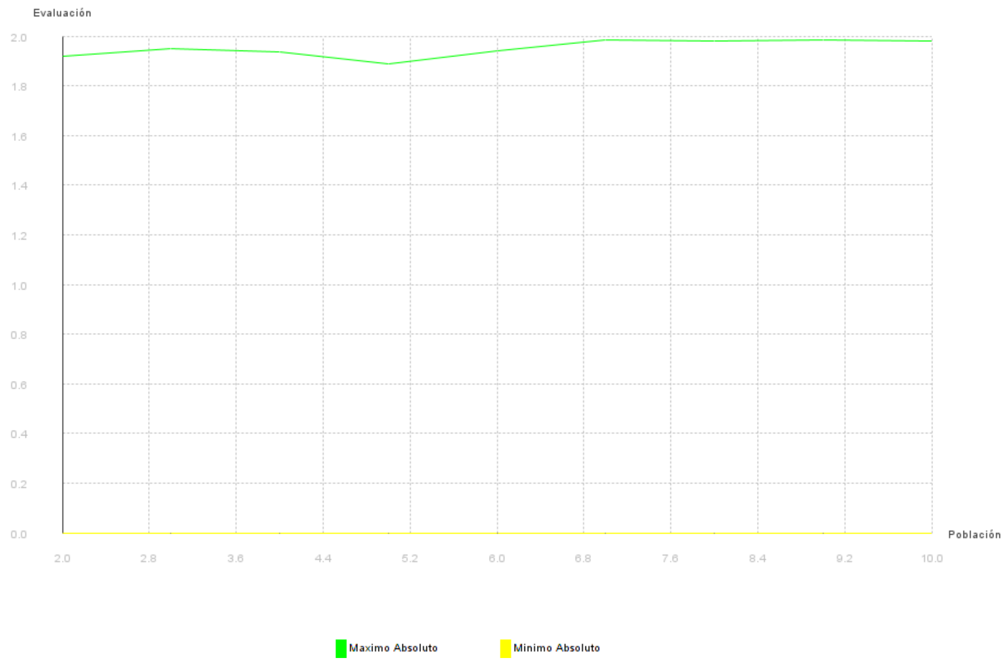
$$f(x) = x + |sen(32\pi x)| \;\; \forall\, x \in [0,1]$$



Número de generaciones = 100; Prob. cruce =0.8; Prob. mutación = 0.01; Población = 100; Selección por ruleta.

Maximo 1.984375 [Gen 0 = 0.984375]
Mínimo 0.043643608598219766 [Gen 0 = 0.0313720703125]

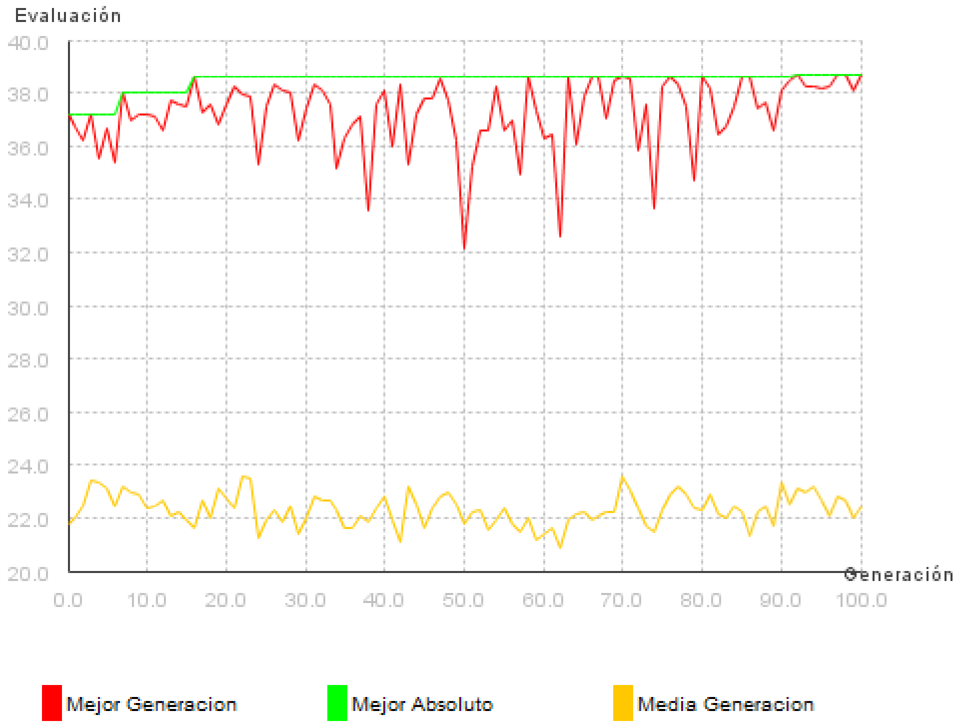
Observamos como alcanzamos el valor óptimo en el valor esperado.



Número de generaciones = 50; Prob. cruce =0.8; Prob. mutación = 0.01; Población = [2..10]
Se observa que cuanto mayor es la población manteniendo el número de generaciones constante se llega a un valor más óptimo. Y a partir de un numero de población el valor óptimo permanece más o menos constante.

Función 2

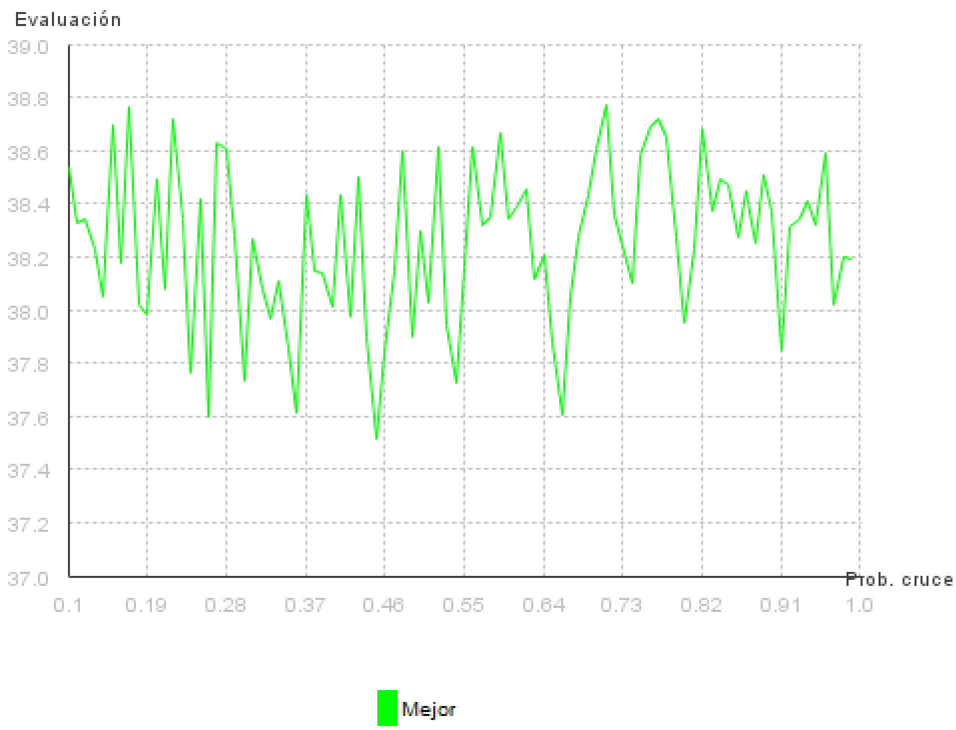
$$f(x,y) = 21.5 + x[?][?]sen(4\pi x) + y[?][?]sen(20\pi y)$$



Número de generaciones = 100; Prob. cruce =0.8; Prob. mutación = 0.01; PoblaciÓn = 100; Selección por ruleta.

Maximo 38.6878131178406 [Gen 0 = 12.099078369140624] [Gen 1 = 5.724981689453125]

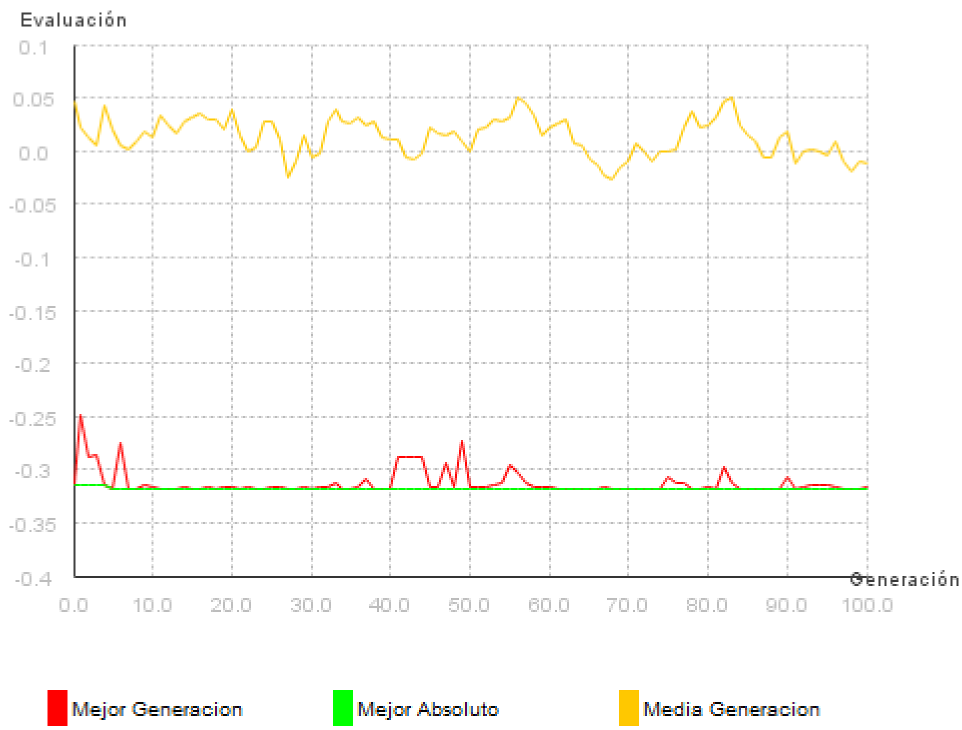
Vemos como la función se acerca al máximo valor óptimo.



Se puede observar que en una función con dos variables, lo que implica un mayor espacio de búsqueda, una tasa mayor de 0,64, hace que el máximo se estabilice próximo a los valores de solución óptima.

Función 3

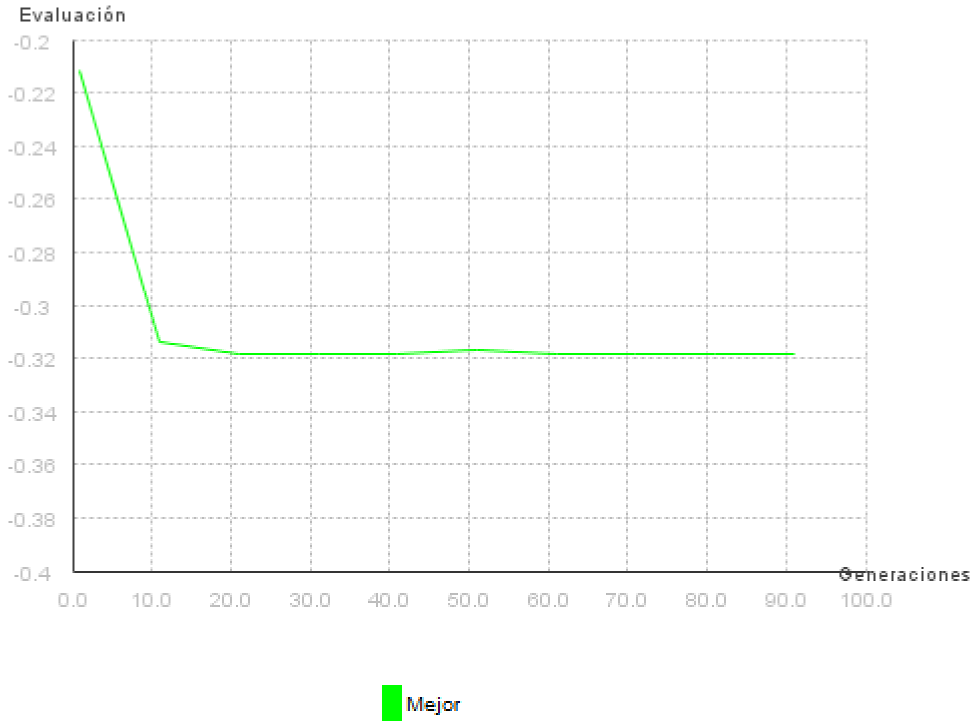
$$f = \frac{sen(x)}{1+\sqrt{x}+\frac{cos(x)}{1+x}} \ \forall \ x \in [0,25]$$



Número de generaciones = 100; Prob. cruce =0.8; Prob. mutación = 0.01; Población = 100; Selección por ruleta.

Maximo -0.3180704796302587 [Gen 0 = 4.58221435546875]

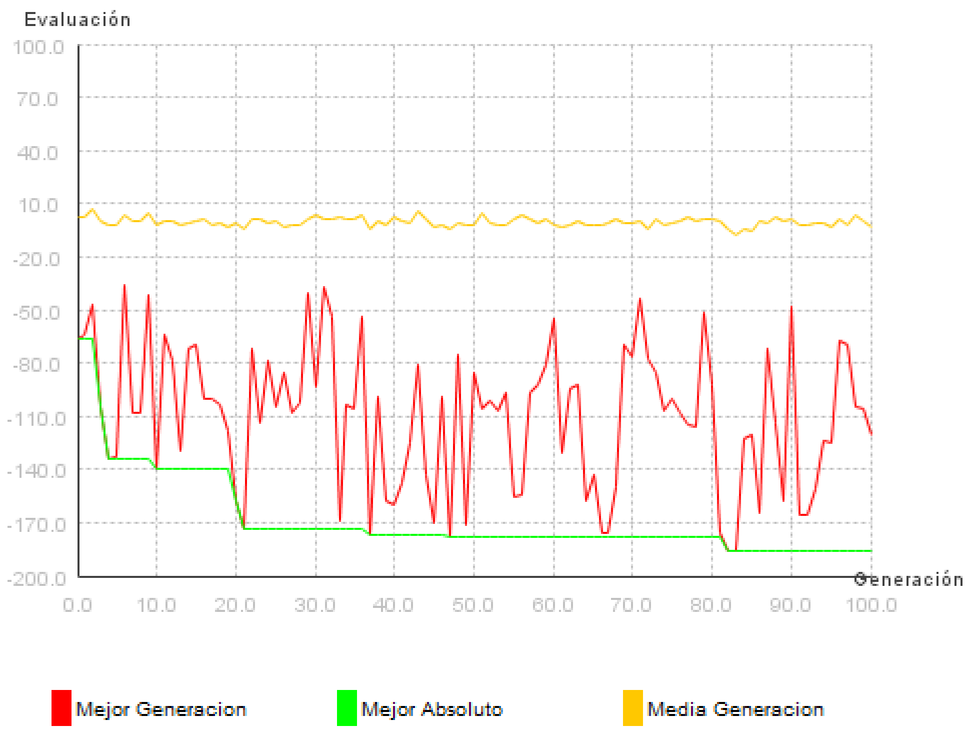
Observamos como converge al valor óptimo.



Numero de generaciones = [1..100]; Prob. cruce =0.8; Prob. mutación = 0.01; Población =25
Observamos que para una población no muy grande el número de generaciones es importante a la hora de aproximarnos a un valor óptimo de la función.

Función 4

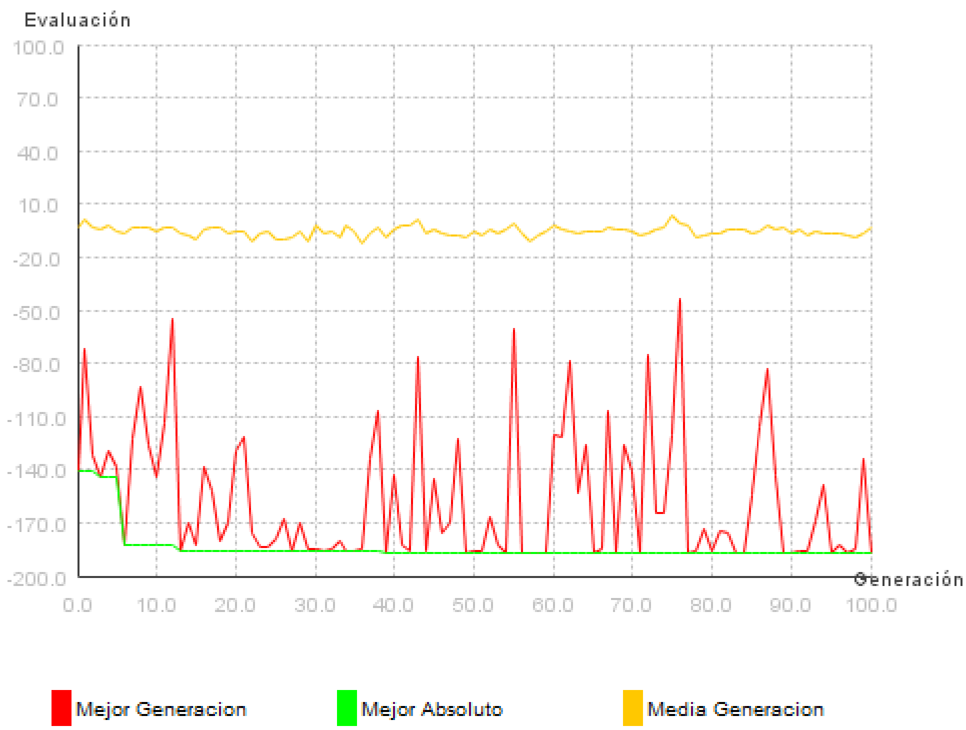
$$f(x_i) = \sum_{i=1}^{5} i[?][?]cos((i+1)x_1+i) \sum_{i=1}^{5} i[?][?]cos((i+1)[?][?]x_2+i) \forall x_i \in [-10,10] \wedge i \in \{1,2\}$$



Numero de generaciones = 100; Prob. cruce =0.8; Prob. mutación = 0.01; Población =100;

Maximo -185.95923699769605 [Gen 0 = -1.407470703125] [Gen 1 = 5.478515625]

Al tratarse de un espacio de búsqueda muy extenso, la convergencia no es total, vamos a ver ahora un ejemplo con elitismo, donde veremos que se aproxima más al valor óptimo.



Maximo -186.704198333354464 [Gen 0 = -1.424560546875] [Gen 1 = -7.080078125]

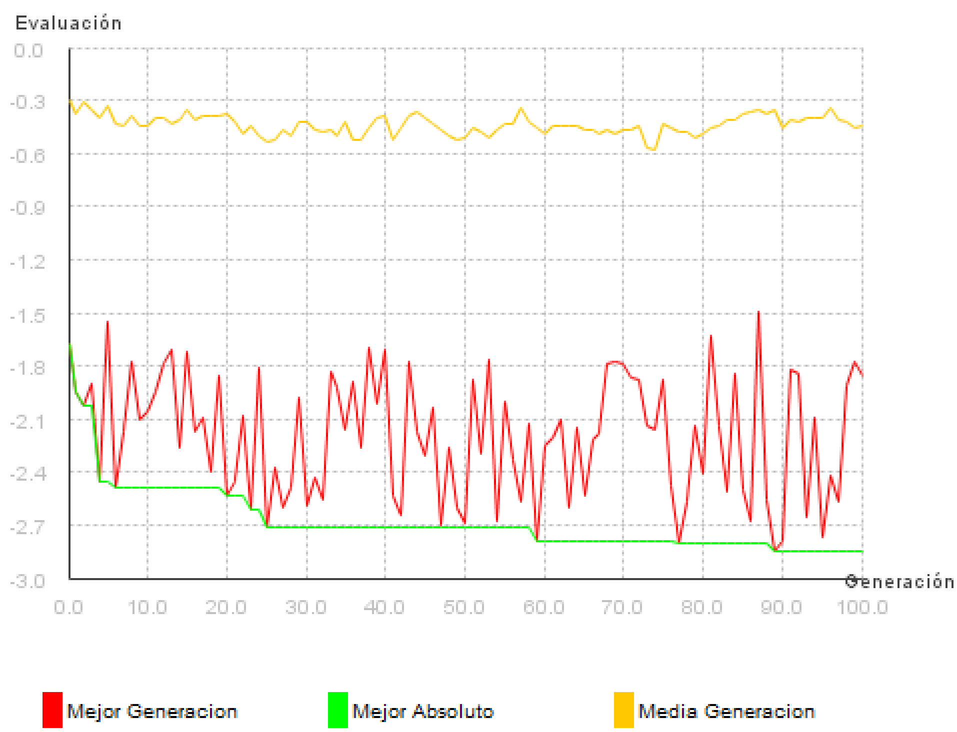
Con un elitismo del 2% hemos conseguido aproximar más la fpoblación al punto de convergencia óptimo.

Función 5

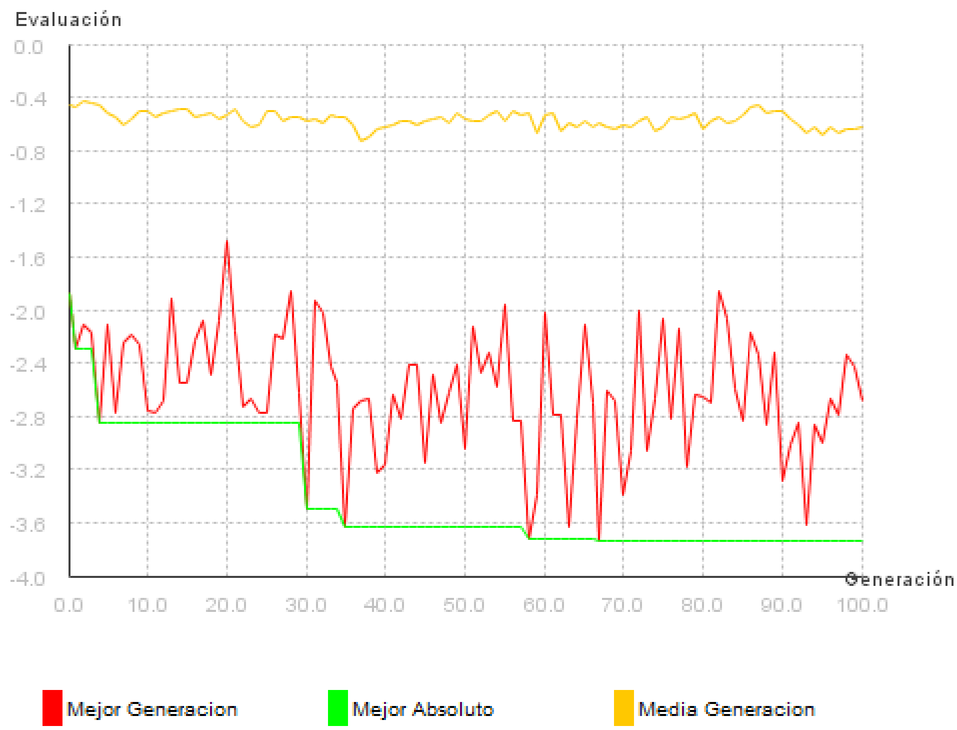
$$f(x_i) = -\sum_{i=1}^{n} sen(x_i)[?][?]sen^{20}(\frac{(i+1)x_i^2}{\pi})\ \forall x_i \in [0,\pi] \wedge i \in [1..n]$$

Esta función cuenta con un número variable de argumentos de entrada. Conforme se aumenta el número de argumentos, también aumenta el espacio de búsqueda. En esta función vamos a utilizar un elitismo del 3% para intentar aproximarnos lo más posible a los valores óptimos. Con un número de variables igual a 3 obtenemos la siguiente gráfica:

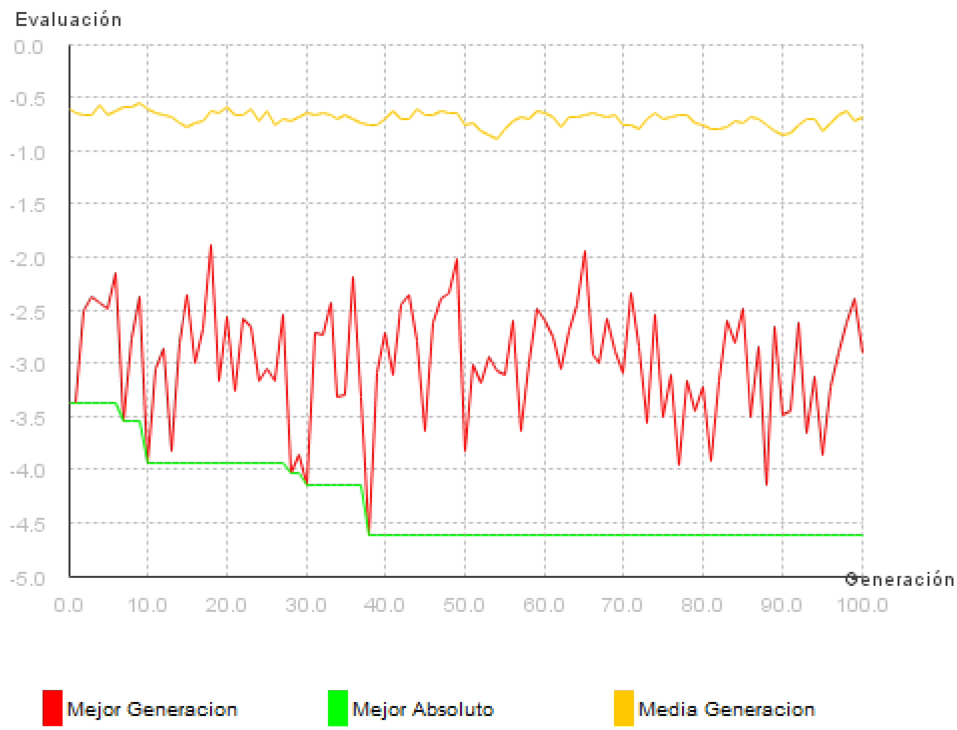
Con tres argumentos de entrada se obtiene un resultado con un error relativo de un 1,8%:
Maximo -2.843240352891735 [Gen 0 = 1.5734807931736965]
[Gen 1 = 1.2814492006799676] [Gen 2 = 1.1263253935050321]



Con cuatro argumentos de entrada se obtiene un resultado con un error relativo del 3.9%:
Maximo -3.732434306421362 [Gen 0 = 1.5811506971131246]
[Gen 1 = 1.273971044339025] [Gen 2 = 1.1512525813081738]
[Gen 3 = 1.722085182000118]



Con 5 argumentos obtenemos un error relativo de 5.6% y el siguiente resultado:
Maximo -4.61248899676l6465 [Gen 0 = 1.5615924420675826] [Gen 1 = 1.3117453212407089] [Gen 2 = 1.9378012302965362] [Gen 3 = 0.9993884833074952] [Gen 4 = 1.5615924420675826]

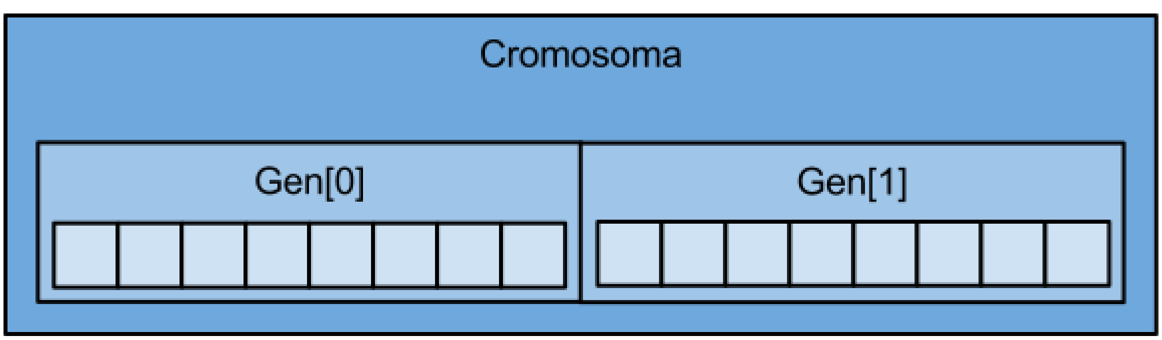


Sobre la implementación

En este epígrafe se comentan algunos puntos generales sobre la implementación.

La arquitectura interna de los cromosomas consiste en una secuencia de genes. Cada uno de estos genes, a su vez, consta de una cadena de bits.

El siguiente diagrama ilustra un cromosoma con dos genes que a su vez tienen cada uno una secuencia binaria de 8 bits.

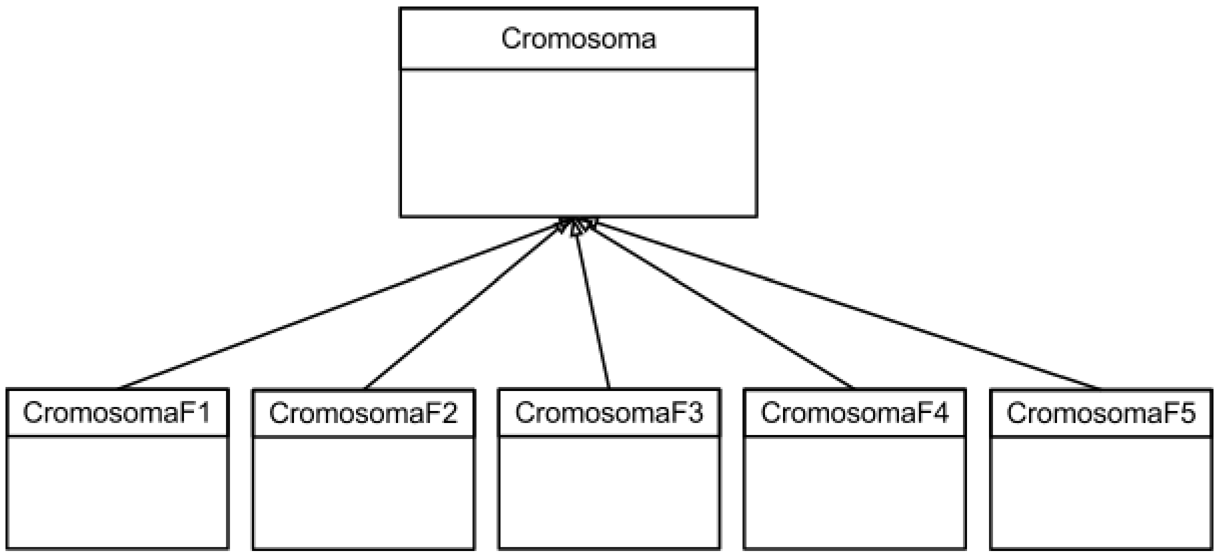


Por otro lado, cada función tiene un cromosoma propio heredado de “Cromosoma” en el que se indican las características propias:

- El número de genes
- La función de fitness
- La función de fenotipo de cada gen del cromosoma

Todos los cromosomas tienen las siguiente operaciones comunes:

- Aleatorizar (genera todos los valores de bit de los genes de forma aleatoria)
- Fenotipo (una representación textual del valor de todos los genes del cromosoma)
- Mutación (aplica mutación sobre un cromosoma completo a partir de una probabilidad de mutación)
- Cruce (Cruza dos genes)



Los algoritmos de selección extienden de un tipo genérico “Selección” lo que permite intercambiar distintos métodos sin problema.