FACULTAD DE INGENIERÍA

FORMATO
PRÁCTICAS DE LABORATORIO

UNIVERSIDAD AUTÓNOMA DE CAMPECHE

NOMBRE DE LA ASIGNATURA: *INTELIGENCIA ARTIFICIAL*
NOMBRE DEL PROFESOR: *RAMIREZ ORTEGON JUSTINO*
NOMBRE DEL ALUMNO: *ALBA ARCOS CHRISTOPHER GREGORIO*
TEMA: *COMPUTACIÓN EVOLUTIVA*
PRÁCTICA NÚM. [**3**]

OBJETIVO:
Modelar la función Rastrigin en MATLAB para después aplicar optimización por medio del toolkit de optimización con el algoritmo genético además de la consola.

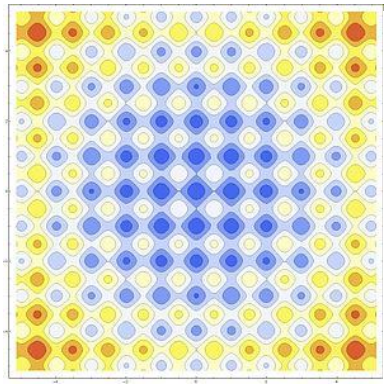
RESUMEN:
Implementamos la función Rastrigin la cual es una función dedicada a la prueba de algoritmos de optimización, por lo que utilizaremos el toolkit de optimización además de la consola de MATLAB con el algoritmo genético, usando variaciones en los parámetros para observar los efectos de estos en el rendimiento del algoritmo. Hacemos hincapié en mostrar que la optimización de la función Rastrigin no es trivial creando casos por medio d ellos parámetros donde no se llega al optimo global.

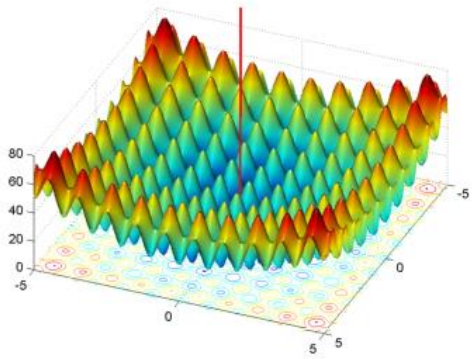
MARCO TEÓRICO:
En optimización matemática , la función Rastrigin es una función no convexa que se utiliza como problema de prueba de rendimiento para algoritmos de optimización . Es un ejemplo típico de función multimodal no lineal. Fue propuesto por primera vez en 1974 por Rastrigin como una función bidimensional y ha sido generalizado por Rudolph. La versión generalizada fue popularizada por Hoffmeister & Bäck y Mühlenbein et al. Encontrar el mínimo de esta función es un problema bastante difícil debido a su gran espacio de búsqueda y su gran número de mínimos locales. En un dominio n-dimensional se define por: $$f(\mathbf{x}) = An + \sum_{i=1}^{n} \left[x_i^2 - A\cos(2\pi x_i)\right]$$ Tiene un mínimo global en dónde: $A = 10 x_i \in [-5.12, 5.12] \mathbf{x} = \mathbf{0} f(\mathbf{x}) = 0$ Este es un ejemplo tipo de función no lineal multimodal, caracterizada por un número muy grande de mínimos locales, aunque regularmente distribuidos, en su dominio habitual de definición. La función se define en un dominio n-dimensional, xi variando entre −5.12 y +5.12. En el hipercubo [−5.12,5.12] n esta función alcanza su máximo cuando cada uno de los sumandos (idénticos respecto de cada variable) de la función alcanza su máximo. Así, pues, considera la función.

$$g(x) = x^2 - 10\cos(2\pi x).$$

Se puede ver que esta función es par, y es "globalmente" creciente, debido al término $x2$, pero con cierto carácter oscilatorio debido al término del coseno. Este coseno alcanza su máximo en $x = k \pm 0.5$, siendo k un entero. Es fácil ver que su máximo en [0, 5.12] se alcanza en $x = 4.5$. Así, el máximo en [−5.12, 5.12], alcanzado, por ejemplo, en x = (4.5, … , 4.5) es:

$$M = 10n + n(4.5^2 + 10) = 40.25n.$$





Un algoritmo genético (AG) es un método para solucionar problemas de optimización con o sin restricciones basándose en un proceso de selección natural que imita la evolución biológica. Este algoritmo modifica repetidamente una población de soluciones individuales. En cada paso, el algoritmo genético selecciona individuos de la población actual aleatoriamente y los utiliza como padres para producir los hijos de la siguiente generación. Tras varias generaciones sucesivas, la población "evoluciona" hacia una solución óptima. El algoritmo genético se puede aplicar para solucionar problemas que no se adaptan bien a los algoritmos de optimización estándar, incluidos aquellos problemas en los que la función objetivo es discontinua, no diferenciable, estocástica o altamente no lineal. El algoritmo genético difiere de un algoritmo de optimización clásico basado en derivadas de dos formas principales, tal y como se resume en la tabla siguiente.

Algoritmo clásico	Algoritmo genético
Genera un único punto en cada iteración. La secuencia de puntos se aproxima a una solución óptima.	Genera una población de puntos en cada iteración. El mejor punto de la población se aproxima a una solución óptima.
Selecciona el siguiente punto de la secuencia mediante un cálculo determinista.	Selecciona la siguiente población mediante un cálculo que emplea generadores de números aleatorios.

FACULTAD DE INGENIERÍA

FORMATO
PRÁCTICAS DE LABORATORIO

UNIVERSIDAD AUTÓNOMA DE CAMPECHE

EQUIPO Y SOFTWARE UTILIZADO:
• Computadora personal (HP Omen con Core i7 y 8 Gb de RAM) • El software de MATLAB

FUNCIONALIDAD DEL PROGRAMA:

```
function [Y] = FRastrigin(X)
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here
    Y = 20+(X(1)^2)+(x(2)^2)-10*(cos(2*3.1416*X(1))+cos(2*3.1416*X(2)));

end
```

```
>> L=[-5.12;-5.12]

L =

   -5.1200
   -5.1200

>> U=[5.12;5.12]

U =

    5.1200
    5.1200
```

Empezamos definiendo la función Rastrigin, en este caso para términos de simplicidad utilizaremos la versión con solo dos dimensiones, sin embargo, no por esto se debe subestimarla, pues como veremos no cualquier algoritmo puede resolverlo como si fuese un caso trivial de optimización.

Posterior a eso definiremos los limites superiores e inferiores, ya que buscar en toda la función o en extensiones mas grandes a estos limites es inviable. Por lo que estos irán de 5.12 a -5.12 en ambas dimensiones.

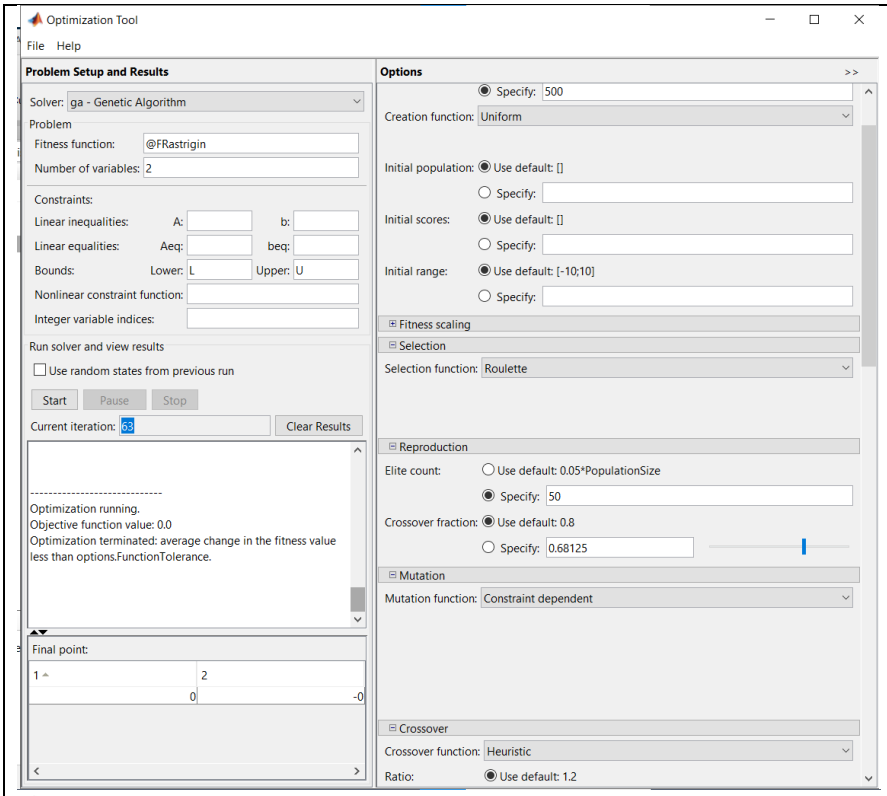
```
>> ga(@FRastrigin,2,[],[],[],[],L,U)
Optimization terminated: average change in th

ans =

   1.0e-04 *

    0.1820    0.0209

>> [X,Y] = ga(@FRastrigin,2,[],[],[],[],L,U)
Optimization terminated: average change in th

X =

   1.0e-04 *

    0.1228    0.1317


Y =

   6.4294e-08
```

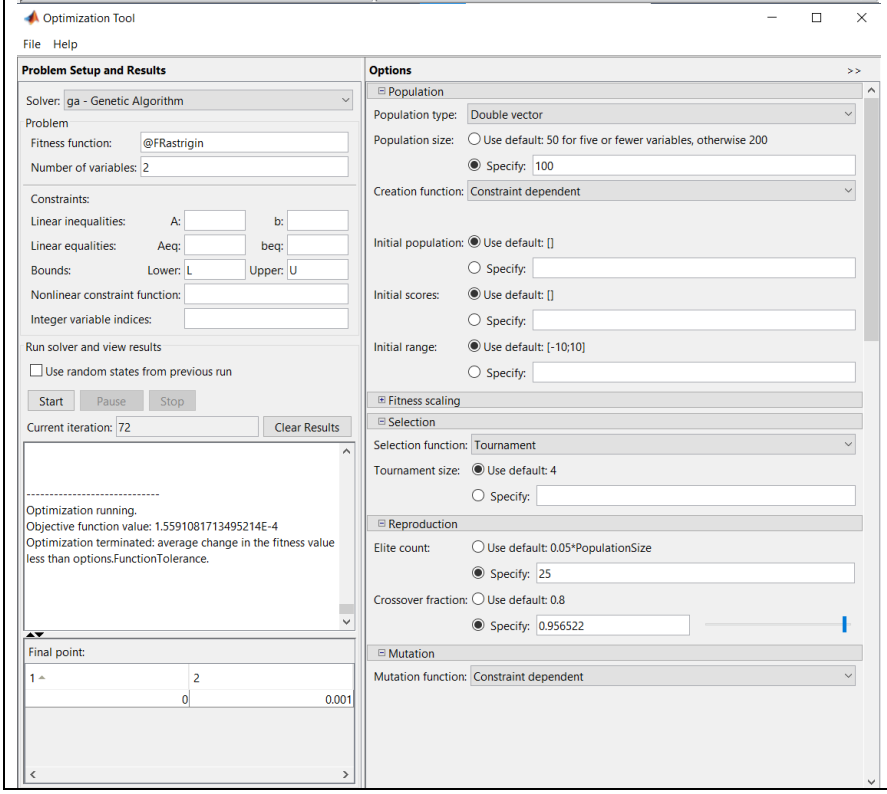
Una vez definida la función y los limites procederemos a utilizar la función en la consola, para esto llamaremos a la función ga con los parámetros correspondientes, esta es la función de optimización por medio de un algoritmo genético.

Si deseamos obtener tanto el valor Y como el X deberemos darle el vector donde guardara los resultados a los que llego el algoritmo. En las primeras pruebas que realice podemos observar que los resultados se acercan muchísimo a ser 0, mientras que en otros casos se estanca en un mínimo local, y es ahí precisamente donde vemos el espíritu de la función Rastrigin.

```
>> ga(@FRastrigin,2,[],[],[],[],L,U)
Optimization terminated: average char

ans =

    0.0000   -0.9950
```

```
>> [X,Y] = ga(@FRastrigin,2,[],[],[],[],L,U)
Optimization terminated: average change in th

X =

   -0.9950    0.0000


Y =

    0.9950
```





Una vez realizada la optimización a través de la consola, procederemos a utilizar el toolkit de optimización que integra MATLAB esto con el fin de jugar con los parámetros de una manera más sencilla y rápida.

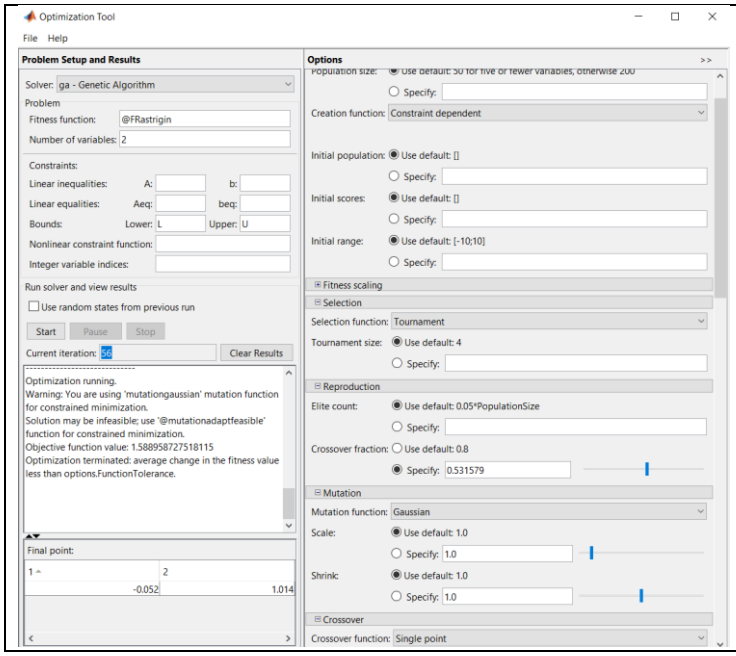
Vemos que con 500 de población inicial, una selección por ruleta y una reproducción del 10% de la población, una mutación dependiente a la restricción y una cruza heurística obtenemos un resultado muy cercano a 0, esto lo observamos en el -0 lo que nos indica un valor levemente menor a 0, además de que el valor de la función objetivo si llego a un 0.0 por lo que podríamos creer que la optimización es relativamente sencilla.

Sin embargo en las siguientes pruebas, donde reducimos la población inicial a solo 100, lo cual ya es un golpe bastante fuerte para este algoritmo, con una selección de torneo con lo que la posible exploración del espacio de búsqueda o visto de otra forma la variedad genética se verá severamente afectado además de la reproducción es de un 25% vemos que la función objetivo nos da un valor relativamente cercano a cero, del orden de 10^-4 sin embargo se suele esperar del orden 10^-8 por lo que es un valor bastante malo, cosa que podemos observar al momento de que nos da 0.001 en vez de 0.

FACULTAD DE INGENIERÍA

FORMATO
PRÁCTICAS DE LABORATORIO

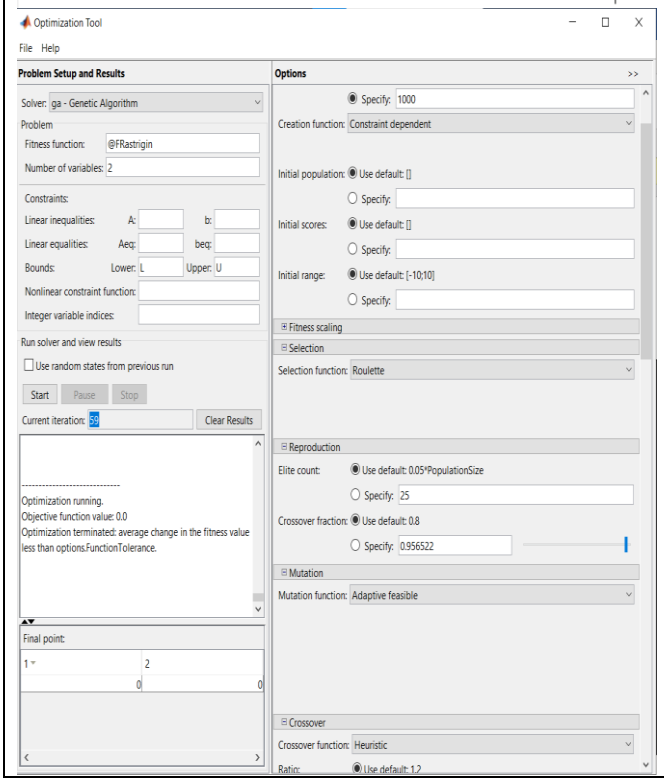
UNIVERSIDAD AUTÓNOMA DE CAMPECHE



En esta corrida tendremos una población de 50, con torneo y una reproducción por default, además de una reproducción Gaussiana la cual precisamente con estos parámetros, los cuales ya son pobres por si mismo, la variedad genética se verá rápidamente estancada, ya que el espacio de búsqueda no puede ser explorado ni por una gran cantidad de pobladores empezando en varios puntos, ni por nuevos pobladores que surgen de la cruza de otros, o por la mutación que nos lleva a nuevos puntos, por lo que en esencia tenemos un algoritmo con parámetros bastante pobres, el 1.014 que obtenemos lo dice por si mismo, con lo que claramente notamos que se ha estancado en un mínimo local.

INSTRUCCIONES DE USO:

```
>> ga(@FRastrigin,2,[],[],[],[],L,U)
```



Tanto el uso de la función, así como el del toolkit son para nada complicados, la función ga cuenta con documentación sobre los parámetros que son necesarios ingresar, tanto los minios que se necesitan, así como los opcionales. Para utilizar el toolkit accederemos a este por medio de la pestaña “APPS” de MATLAB y el botón “optimization” una vez en la interface el modificar los parámetros del algoritmo es relativamente sencillo; antes de esto se debe de seleccionar “ga-Genetic Algoritm” en la sección de “solver”, con un numero de variables de 2 y los limites superiores en inferiores con los vectores que definimos previamente. Una vez realizados la modificación de los parámetros que

CODIGO RELEVANTE:

Función Rastrigin en dos dimensiones

```
function [Y] = FRastrigin(X)
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here
    Y = 20+(X(1)^2)+(x(2)^2)-10*(cos(2*3.1416*X(1))+cos(2*3.1416*X(2)))

end
```

BIBLIOGRAFÍA:

- Apuntes y clase del profesor
- https://en.wikipedia.org/wiki/Rastrigin_function
- https://www.mathworks.com/help/gads/example-rastrigins-function.html
- https://la.mathworks.com/discovery/genetic-algorithm.html
- Bajpai, P., & Kumar, M. (2010). Genetic algorithm–an approach to solve global optimization problems. Indian Journal of computer science and engineering, 1(3), 199-206.
- Pohlheim, H. (2007). Examples of objective functions. Retrieved, 4(10), 2012.