SOLUCIÓN LABORATORIO 5

Grupo 4

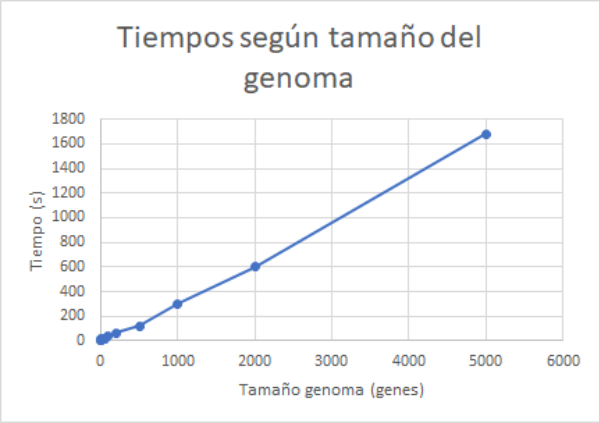
- Santiago Rodríguez Camargo
- Fredy Alexander Gonzalez Pobre
- Víctor Manuel Torres
- Daniel Leonardo Sánchez
- Daniel Felipe Vargas Gómez

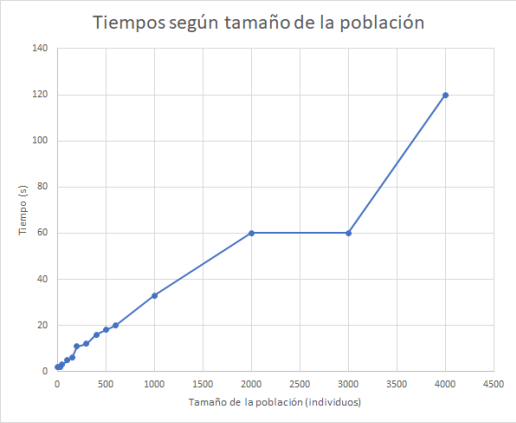
4. Hacer un análisis de complejidad del algoritmo en función de los parámetros dimensionales de configuración(**literal a, punto 3 en verde**), como también complejidad simplificada O(n).

El presente algoritmo genético, capaz de evaluar varias funciones y parámetros en general, se compone de varios algoritmos pequeños o insumos para la simulación.
En ese entendido, se tienen algunos *sub-algoritmos* para:

a. Leer y convertir de una cadena de caracteres ingresados la función a evaluar (*listToDecimal*).
b. Obtener los individuos con mutaciones (mutate) en cada generación.
c. Graficar la curva de la función digitada por el usuario (eval) con la primera población.
d. Calcular la *i-ésima* generación luego de varias centenas y graficarla.
   Entre otros algoritmos como el de mejor generación, individuo y determinación del error estadístico hasta la *i-ésima* generación.

4.1. Único ingreso de parámetros: Tamaño_genoma_Individuos y Tamaño_población.





De acuerdo con estas gráficas, evidentemente el algoritmo genético tiende a comportarse dentro de una complejidad global O(n), haciendo hincapié en que los parámetros que permanecieron constantes en sus valores fueron:

Función a evaluar: **x+math.sin(x*math.pi)**
Valor inferior del rango de búsqueda: **0**
Valor superior del rango de búsqueda: **5**

Número de genes de cada individuo: **15**, constante para el caso de la segunda gráfica y variable en el de la primera.

Tamaño de la población: **100**, constante para el caso de la primera gráfica y variable para el estudio de la segunda.

Probabilidad de mutación entre 0 y 1: **0.005**

Número de generaciones: **300**

Raíz del máximo real de la función en el intervalo: **5.55**

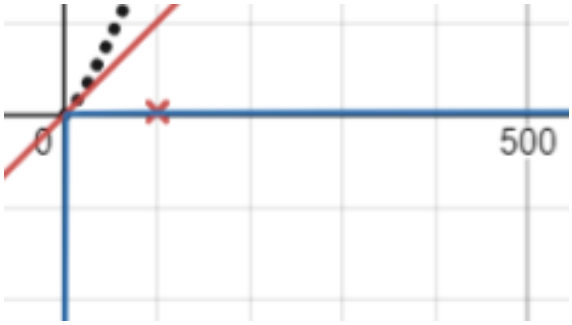
Es de anotar que el parámetro *Tamaño_genoma* (denominado *genesize* en el código del notebook) es usado por la funciones que grafican la primera población y crean la descendencia, que se valen de dos ciclos for consecutivos y un ciclo for respectivamente. Con esto, su complejidad en el por caso puede ser n*O(1) = O(n), por regla de sumas donde las ejecuciones internas iteran *n* número de veces. De similar forma sucede con la otra instrucción for de esta función. De nuevo ha de ser O(n), que sumada con la anterior se mantiene en O(n).

Por su parte, el parámetro *Tamaño_poblacion* (denominado *populationsize* en el código del notebook) lo usan las mismas funciones anteriores, exhibiendo similar comportamiento de carga algorítmica, pero además es usado por la función que crea la descendencia y la que calcula el error de la *i-ésima* generación siendo la primera de complejidad lineal O(n) debido al principal y único ciclo for que la comanda y, la segunda (la del error), de complejidad constante O(1) dado que todo el tiempo calcula un cociente.

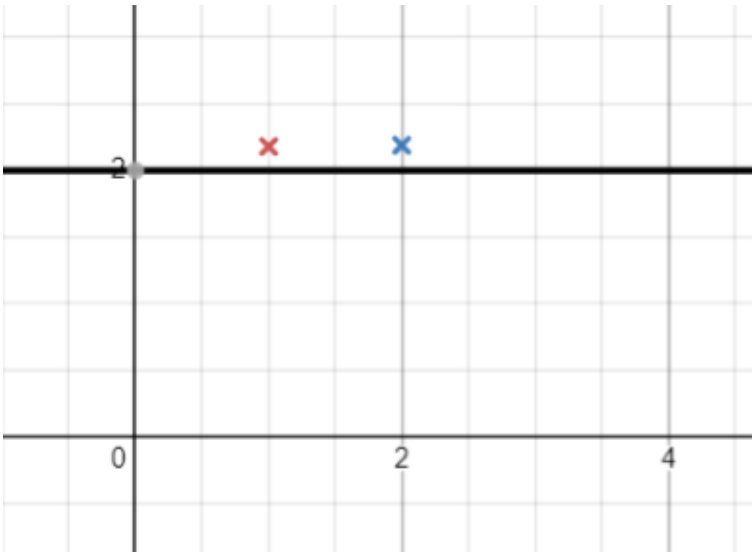
4.2. Único ingreso de parámetros: Generaciones y Rango_de_búsqueda.

El número de generaciones tiene una importancia muy grande en el tiempo de ejecución del algoritmo. Cuando se aumenta el número de generaciones, se espera que el tiempo de ejecución aumente linealmente.

Al aumentar el mismo parámetro, el espacio también es más demandado pero no en gran cantidad, porque se requiere almacenar mejores resultados por generación.



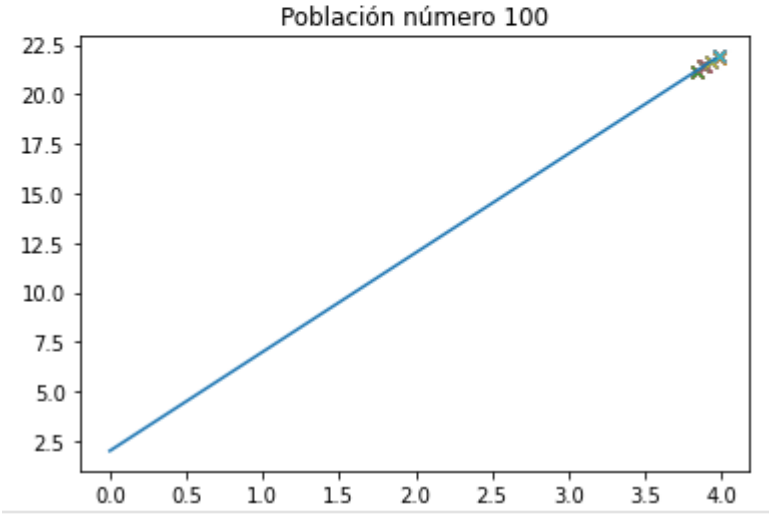
El rango de búsqueda es uno de los factores menos influyentes. Este presenta tiempos mayormente constantes independientemente de la función. En cuanto al espacio, este tiene bajo impacto, al aumentar el rango de búsqueda, es necesario un espacio, porque es necesario almacenar más opciones en la genetic_pool para construir individuos.

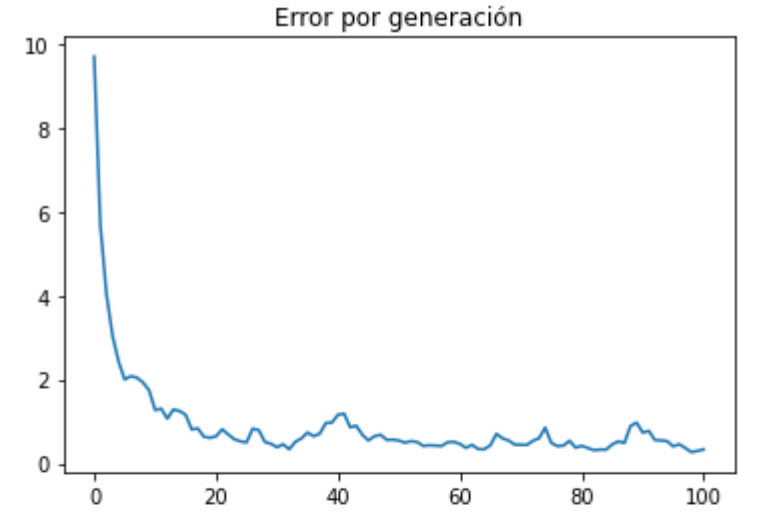


5. Cómo reacciona su algoritmo a variadas funciones de optimización (lineal, oscilantes(varias frecuencias), etc)?

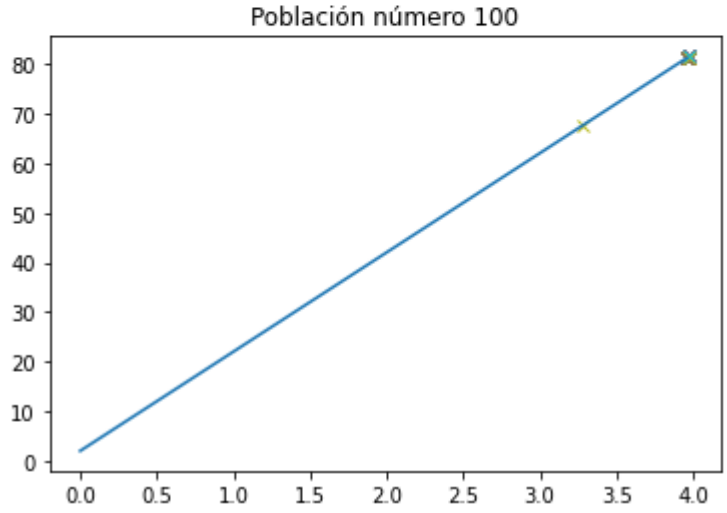
Se realizaron diferentes experimentos con funciones lineales con diferente pendiente y oscilantes de diferente frecuencia, cuyos resultados se muestran abajo. En todos los casos las variables de tamaño de población, número de generaciones, número de genes y probabilidad de mutación se mantuvieron constantes:

Función a evaluar: **5x + 2**
Valor inferior del rango de búsqueda: **0**
Valor superior del rango de búsqueda: **4**
Número de genes de cada individuo: **15**
Tamaño de la población: **100**
Probabilidad de mutación entre 0 y 1: **0.005**
Número de generaciones: **100**
Raíz del máximo real de la función en el intervalo: **22**

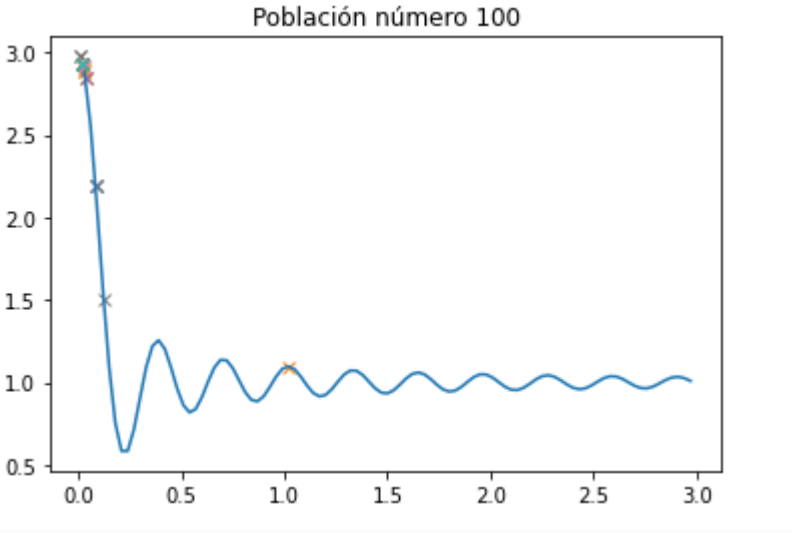


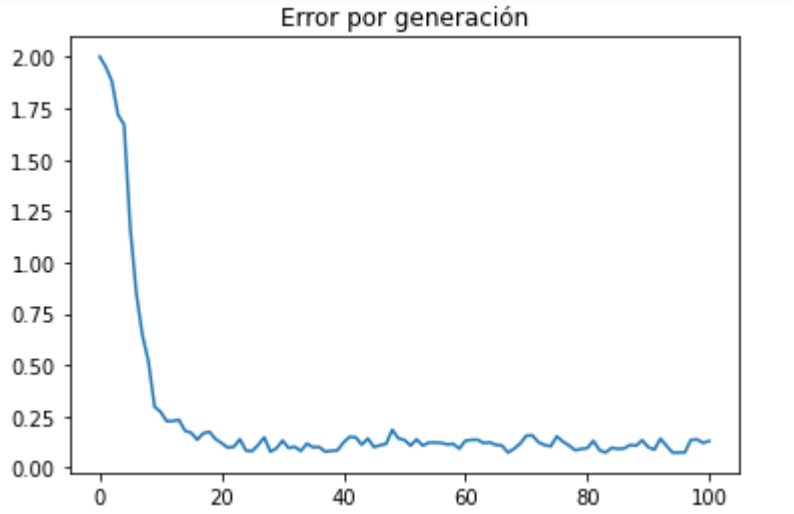



```
Ingrese la función a evaluar: 20*x+2
Ingrese el valor inferior del rango de búsqueda: 0
Ingrese el valor superior del rango de búsqueda: 4
```

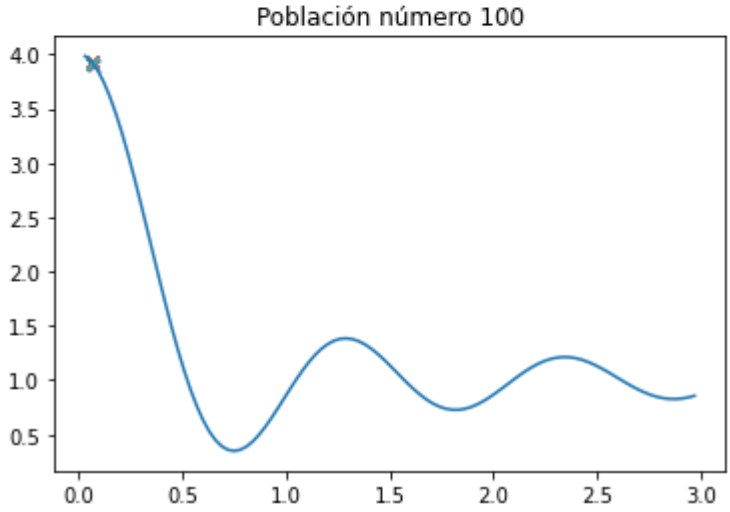



```
Ingrese la función a evaluar: math.sin(20*x)/(10*x)+1
Ingrese el valor inferior del rango de búsqueda: 0
Ingrese el valor superior del rango de búsqueda: 3
```






```
Ingrese la función a evaluar: math.sin(6*x)/(2*x)+1
Ingrese el valor inferior del rango de búsqueda: 0
Ingrese el valor superior del rango de búsqueda: 3
```



En todos los experimentos se ve, gracias a la gráfica del error, que los individuos encuentran el máximo antes de la generación 100, de hecho lo encuentran en menos de 20 generaciones. Dicha gráfica desciende rápidamente en las primeras generaciones y se estabiliza cerca de la generación 10, dibujando una curva similar a la de e^{-x}. Podemos observar que en las funciones lineales la curva desciende un poco menos abruptamente, mientras que en las oscilantes este es más rápido. Parece que la frecuencia de la función oscilante también afecta la velocidad con la que se estabilizan los individuos, pues es evidente que en la función con menos oscilaciones la curva del error es mucho más brusca en su descenso que en la de la función con más oscilaciones. Esto quizá puede deberse a que los individuos identifican más rápidamente cuál de los picos es más alto. Finalmente el error parece moverse en todos los casos en dimensiones muy similares (probablemente debido a la probabilidad de mutación), sin embargo es un poco mayor en la función lineal de menor pendiente, quizá porque los cambios de altura son menos abruptos.

6. Concluya y Describa, por sus experimentos, cómo están relacionados los parámetros de configuración de su algoritmo, con el costo en (tiempo y espacio).

- a pesar de que al modificar el número de genes en el genoma de cada uno de los individuos, se modifica el costo en tiempo, pero su cambio no es significativo, en donde se nota el efecto de modificar el número de genes dentro del genoma, es en el espacio ocupado por cada uno de los individuos, el aumento es significativo, tanto así que si el número de genes es demasiado alto, el entorno de ejecución tendrá problemas para leer cada individuo
- La población es un elemento que viene atado al número de genes de cada individuo, de esta manera, si se aumenta la población, pero hay un conteo bajo de genes en cada individuo, el efecto no es significativo ni en tiempo de ejecución ni es espacio o recursos usados para la ejecución, mientras que, al aumentar el número de genes en cada individuo, el aumento en tiempo de ejecución y espacio usado para la misma, aumenta de manera significativa
- El efecto de la probabilidad de mutación es uno dependiente de otros factores tales como, la población y el número de generaciones,Mientras que la población facilita que un valor alto en la probabilidad de mutación aumenta el tiempo de ejecución, el número de generaciones al repetir el proceso de mutación cada vez que una generación nueva es generada, si la probabilidad de mutación es baja, No importa el número de generaciones, pues en la mayoría de los casos, no se realizará ninguna mutación, mientras que, con una mutación alta, El número de generaciones aumenta el tiempo de ejecución del programa, puesto que para cada nueva generación casi en todos los casos se generará una mutación
- Dentro del argumento relacionado a la función que se debe evaluar y teniendo en cuenta los resultados anteriores, se entiende que, a pesar que se estabiliza más rápido una función oscilante en función de su frecuencia,la diferencia resulta mínima en los experimentos realizados, es decir, no genera un aumento o disminución significativos en el tiempo de ejecución total del programa, lo mismo se puede decir de la “complejidad” de la función a evaluar, solamente al incrementar el grado de la ecuación a un grado 5, se nota un aumento significativo en el tiempo de ejecución.

7. ¿Qué parámetro considera usted más relevante para acelerar la convergencia de la búsqueda?

Gracias a los análisis realizados anteriormente, podemos ver que, aunque todos los parámetros tienden a afectar de alguna manera la complejidad de la búsqueda, algunos resultan mucho más relevantes cuando se mantienen constantes el resto de parámetros, claro está; si se cambia el valor de todos los parámetros constantemente, resulta demasiado abstruso la búsqueda de una optimización o aceleración del algoritmo. Dicho esto, consideramos que, en general, el **número de generaciones** necesarias para encontrar un máximo considerable se puede mantener en un valor bastante pequeño (cerca a 20), de esta manera, gracias al análisis que ya se discutió, se ahorrará bastante tiempo debido a que cada generación aporta considerablemente tiempo a la complejidad, y se tendrá una solución admisible, sin necesidad de realizar el triple o cuádruple de generaciones.