Interpretación del código de R:

Nos dieron este código:

```
library(GA)

Ackley <- function(x1, x2)
{
 -20*exp(-0.2*(x1^2 +x2^2)^0.5) -
exp(0.5*(cos(2*3.14159*x1)+cos(2*3.14159*x2)))+exp(1)+20
}
x1 <- x2 <- seq(-10, 10, by = 0.1)
f <- outer(x1, x2, Ackley)
persp3D(x1, x2, f, theta = 50, phi = 20, col.palette = bl2gr.colors)
filled.contour(x1, x2, f, color.palette = bl2gr.colors)


GA <- ga(type = "real-valued",
 fitness = function(x) -Ackley(x[1], x[2]),
 lower = c(-10, -10), upper = c(10, 10),
 popSize = 500, maxiter = 1000, run = 100,
optim = TRUE, optimArgs = list(method = "Nelder-Mead", poptim = 0.05,
pressel = 0.5, control = list(fnscale = -1, maxit = 100))
)

summary(GA)
```

Y se nos quedó de tarea entender las líneas de código que referían al algoritmo genético que se hacía utilizando la librería GA de R.

```
GA <- ga(type = "real-valued",
 fitness = function(x) -Ackley(x[1], x[2]),
 lower = c(-10, -10), upper = c(10, 10),
 popSize = 500, maxiter = 1000, run = 100,
optim = TRUE, optimArgs = list(method = "Nelder-Mead", poptim = 0.05,
pressel = 0.5, control = list(fnscale = -1, maxit = 100))
)
```

Así como experimentar (Para aprender cómo funcionaban) con las funciones y con los parámetros que podía recibir cada una de las líneas de código.
Entonces, hice una lista de los parámetros que lleva y los posibles valores que podían tener:

1. **type**: Este parámetro define el tipo de datos que se utilizarán en el algoritmo genético. En este caso, **"real-valued"** indica que se trabajará con valores reales (números decimales). Pero puede trabajar con otro tipo de valores, como:

    - **"real-valued"**: Indica que las variables son valores continuos y reales (números decimales).
    - **"binary"**: Indica que las variables son representadas como cadenas binarias (ceros y unos). Es útil para problemas donde las soluciones pueden ser codificadas en binario.
    - **"permutation"**: Indica que las variables son permutaciones de un conjunto finito de números. Esto es útil para problemas como el de la ruta del viajero (TSP, por sus siglas en inglés).

2. **fitness**: Esta es la función de aptitud (fitness function) que el algoritmo genético intenta optimizar. En este caso, es la función Ackley negada (para convertir un problema de minimización en maximización, ya que los algoritmos genéticos por defecto maximizan). Nuestra función Ackley es

```
Ackley <- function(x1, x2)
{
 -20*exp(-0.2*(x1^2 +x2^2)^0.5) -
exp(0.5*(cos(2*3.14159*x1)+cos(2*3.14159*x2)))+exp(1)+20
}
```

    Que recibe 2 parámetros **x1** y **x2**.
    x es un vector que contiene una posible solución del problema. Dado que el problema tiene dos variables (dimensiones), **x** tiene dos elementos: **x[1]** y **x[2]**.

    Por lo tanto, cuando se pasa **x** a la función de aptitud, se está descomponiendo en sus componentes:

    - **x[1]** representa el primer valor de la solución.
    - **x[2]** representa el segundo valor de la solución.

3. **lower** y **upper**: Estos parámetros definen los límites inferiores y superiores de los valores que pueden tomar las variables en la población inicial. Aquí, ambos parámetros están definidos como **-10** y **10** respectivamente para las dos dimensiones del problema. Y se escriben 2 veces cada una porque estamos trabajando con la función Ackley, que es una función de dos variables. Por lo tanto, cada individuo en la población del algoritmo genético tiene dos componentes (o genes): **x1** y **x2**. Cada uno de estos componentes necesita su propio límite inferior y superior.

4. **popSize**: Este es el tamaño de la población, es decir, cuántos individuos (soluciones) hay en cada generación. Un tamaño de población más grande puede explorar mejor el espacio de soluciones, pero también puede requerir más tiempo de cómputo. Aquí está establecido en **500**.

5. **maxiter**: El número máximo de iteraciones (generaciones) que el algoritmo genético ejecutará. Aquí está establecido en **1000**.
6. **run**: El número de iteraciones consecutivas sin mejora significativa en la solución óptima antes de que el algoritmo se detenga. Aquí está establecido en **100**.
7. **optim**: Este parámetro indica si se debe realizar una optimización local utilizando un método de optimización local después del algoritmo genético. Aquí está establecido en **TRUE**.
8. **optimArgs**: Este parámetro proporciona los argumentos para el método de optimización local. Es una lista que puede incluir varios parámetros. En este caso:

```
optimArgs = list(
    method = "Nelder-Mead",  # Método de optimización local a utilizar.
    poptim = 0.05,            # Proporción de la población a la que se aplicará la optimización local.
    pressel = 0.5,            # Proporción de los mejores individuos seleccionados para la optimización local.
    control = list(
        fnscale = -1,        # Escalado de la función de aptitud (para minimizar).
        maxit = 100          # Número máximo de iteraciones para la optimización local.
    )
)
```

El método Nelder-Mead es un algoritmo de optimización numérica que no requiere el cálculo de derivadas, lo que lo hace adecuado para funciones no lineales donde las derivadas no están disponibles o son difíciles de calcular. Es conocido también como el método del simplex

La función **optim** en R ofrece varios métodos que pueden ser especificados en el parámetro **optimArgs** de la función **ga**.

1. **Nelder-Mead ("Nelder-Mead")**: Este método no requiere derivadas y es útil para funciones no diferenciables.
2. **BFGS ("BFGS")**: Método de cuasi-Newton que utiliza aproximaciones a la matriz Hessiana.
3. **L-BFGS-B ("L-BFGS-B")**: Variante de BFGS que permite límites en las variables.
4. **CG ("CG")**: Método de gradiente conjugado para optimización sin restricciones.
5. **SANN ("SANN")**: Recocido simulado, adecuado para encontrar el mínimo global en funciones con múltiples mínimos locales.

`"poptim = 0.05",` se refiere a la proporción de la población que será sometida a la optimización local.
Este parámetro es importante porque permite combinar la búsqueda global realizada por el algoritmo genético con una búsqueda local más precisa mediante el método de optimización local seleccionado **poptim (Population Optimization Fraction)**: Es un número entre 0 y 1 que indica la fracción de la población sobre la que se aplicará la optimización local.

0.05 significa que el 5% de la población será sometida a la optimización local.

El parámetro **pressel** indica la proporción de los mejores individuos seleccionados para la optimización local antes de aplicar **poptim**.

- **pressel = 0.5**: Significa que el 50% de los mejores individuos de la población serán considerados para la optimización local.
- **poptim = 0.05**: De estos mejores individuos, el 5% será sometido a la optimización local.