# Ejercicios

*Johana Tellez - Michael Caicedo*

1. Maximizar la función $f(x) = x\,sen(10\,\pi\,x) + 1$, con $x \in [0,1]$
   - L=8: número de genes del cromosoma (bits de longitud del cromosoma).
   - k: un gen cualquiera del cromosoma (puede tomar valor 0 o 1).
   - K=8: número de cromosomas en la población.
   - M=4: número de generaciones ejecutadas en la simulación.
   - x: valor real representado por un cromosoma, decodificado desde la cadena binaria mediante:

$$x = \frac{DEC}{2^L - 1}, DEC\epsilon[0,255]$$

   - f(x): función de aptitud aplicada en el punto x, en este caso se busca **maximizarla**

Primera generación:

| Indice | Bits | Dec | x | f(x) | Aptitud | Prob. Selección |
|---|---|---|---|---|---|---|
| 1 | 01101100 | 108 | 0,423529 | 1,28533 | 1,28533 | 0,149377 |
| 2 | 01010101 | 85 | 0,333333 | 0,711325 | 0,711325 | 0,082668 |
| 3 | 01000010 | 66 | 0,258824 | 1,248943 | 1,248943 | 0,145148 |
| 4 | 00011000 | 24 | 0,094118 | 1,017294 | 1,017294 | 0,118227 |
| 5 | 10100011 | 163 | 0,639216 | 1,602879 | 1,602879 | 0,186282 |
| 6 | 10001011 | 139 | 0,545098 | 0,461353 | 0,461353 | 0,053617 |
| 7 | 00111001 | 57 | 0,223529 | 1,150591 | 1,150591 | 0,133718 |
| 8 | 00111000 | 56 | 0,219608 | 1,126884 | 1,126884 | 0,130963 |

Para la selección se usó el **método de ruleta**, cada cromosoma obtiene una probabilidad de ser elegido padre proporcional a su aptitud, que en este caso es directamente el resultado de la función a maximizar. Se aplicó un **cruce de un punto**, se selecciona al azar un punto dentro de los bits del cromosoma y se intercambia la parte restante entre dos padres para formar dos hijos. Se aplicó una **mutación simple de bit**, con una baja probabilidad al azar, se cambia un bit del cromosoma (de 0 a 1 o de 1 a 0).

**Resultado obtenido después de la cuarta generación:**

$X \approx 0.839 \Rightarrow f(x) \approx 1.803$

Al ejecutar el algoritmo genético con los parámetros dados (L=8, K=8, M=4), se obtuvo como mejor solución:

$$x \approx 0.839, \qquad f(x) \approx 1.80$$

El valor máximo que puede alcanzarse en este problema es aproximadamente:

$$x \approx 0.851, \qquad f(x) \approx 1.85$$

La diferencia se debe a que la población inicial fue pequeña y el número de generaciones bajo, lo que limitó la exploración y diversidad del algoritmo.

Para mejorar los resultados, se recomienda:

- aumentar el número de generaciones,
- incrementar la población inicial,
- y aplicar el elitismo (conservar siempre el mejor cromosoma encontrado).

2. Tome el algoritmo de la dieta y ahora incluya costos. Ahora encuentre una dieta que trate de satisfacer la dieta, pero con un costo mínimo. Este es un ejemplo de AG multiobjetivo con dos funciones objetivo.

   Lo que se modificó fue la definición del problema para pasar de un algoritmo genético mono-objetivo (minimizar solo la diferencia de calorías) a un enfoque multi-objetivo que también considera el costo total de la dieta.

   El código implementa un algoritmo genético multiobjetivo utilizando la librería DEAP, configurado para trabajar con individuos que representan cantidades de distintos alimentos. Cada individuo se inicializa dentro de los rangos permitidos (mínimo y máximo por producto), se evalúa según dos criterios (diferencia calórica respecto al objetivo y costo total), y evoluciona mediante selección, cruza y mutación bajo el esquema NSGA-II. Al final, se construye el frente de Pareto con las soluciones no dominadas y se extraen algunas de las mejores para analizarlas. Los resultados se presentan en dos tablas: la tabla de datos, que contiene la información nutricional y de costo de todos los productos disponibles (sin modificarla con cantidades), y la tabla de resultados, que muestra únicamente los productos seleccionados para una semana con cantidad mayor a cero en la solución final. Esta segunda tabla incluye las columnas SelectedQty, TotalCalories y TotalCost, permitiendo ver qué alimentos fueron elegidos, en qué cantidad, cuántas calorías aportan en total y cuánto cuestan

```
--- Tabla de datos (sin columnas de selección) ---
                         Nombre  Min  Max  Calorias  Gram_Prot  Gram_Grasa  Gram_Carb  Cost
                     Banano 1u    0    4        89          1           0         23  0.30
                   Mandarina 1u    0    4        40          1           0         10  0.25
                      Piña 100g    0    7        50          1           0         13  0.60
                      Uvas 100g    0    7        76          1           0         17  0.80
                Chocolate 1 bar    0    4       230          3          13         25  1.50
              Queso Paipa 100g     0    8       350         28          26          2  2.00
                 Quesillo 100g     0    8       374         18          33          1  1.80
                    Pesto 100g     0    8       303          3          30          4  1.20
                   Hummus 100g     0    8       306          7          25         11  1.00
        Pasta de berenjena 100g    0    4       228          1          20          8  1.00
            Batido de proteinas    0    5       160         30           3          5  1.80
      Hamburguesa vegetariana 1    0    5       220         21          12          3  2.50
      Hamburguesa vegetariana 2    0   12       165         16           9          2  2.00
                Huevo cocido 1     0    8       155         13          11          1  0.40
                 Huevo frito 1     0   16       196         14          15          1  0.45
                Medio baguette     0    3       274         10           0         52  0.90
            Pan tajado 1 tajada    0    3        97          3           1         17  0.25
               Pizza de queso1u    0    3       903         36          47         81  6.00
            Pizza vegetariana 1u   0    3       766         26          35         85  6.50
             Leche de soya 200ml   0    1       115          8           4         11  0.80
Leche de soya achocolatada 250ml   0    3       160          7           6         20  1.00
```

```
--- Resultados: Productos seleccionados (SelectedQty > 0) ---
                        Nombre  SelectedQty  TotalCalories TotalCost
                   Banano 1u             3            267     $0.90
                 Mandarina 1u            4            160     $1.00
                    Piña 100g            7            350     $4.20
                    Uvas 100g            3            228     $2.40
               Chocolate 1 bar           4            920     $6.00
             Queso Paipa 100g            2            700     $4.00
                Quesillo 100g            4           1496     $7.20
                   Pesto 100g            8           2424     $9.60
                  Hummus 100g            6           1836     $6.00
       Pasta de berenjena 100g           4            912     $4.00
     Hamburguesa vegetariana 1           1            220     $2.50
              Huevo cocido 1             1            155     $0.40
               Huevo frito 1            15           2940     $6.75
              Medio baguette             3            822     $2.70
          Pan tajado 1 tajada            3            291     $0.75
Leche de soya achocolatada 250ml         3            480     $3.00
```

Además se muestra cómo evoluciona el algoritmo generación tras generación. Al inicio, en la **Generación 1**, la mejor solución encontrada todavía está lejos del objetivo calórico, con una diferencia de 37 calorías, y además tiene un costo relativamente alto de 85.25. Sin embargo, a medida que avanzan las generaciones, el algoritmo comienza a refinar los individuos, y ya en la **Generación 10** logra una solución mucho más cercana a la meta calórica (solo 4 calorías de diferencia), aunque con un costo ligeramente mayor. Más adelante, en la **Generación 30**, aparece una solución casi perfecta en calorías, con solo 1 de diferencia, y a la vez con un costo reducido a 61.4, mostrando una mejora simultánea en ambos objetivos. Desde la Generación 40 hasta la 60, esa solución se mantiene estable, indicando que el algoritmo ya encontró un punto de equilibrio sólido entre calorías y costo.

```
calorias de proteina:  4200 calorias de carb:  7000 calorias de grasa:  2800
gramos de proteina:  1050.0 gramos de carb:  1750.0 gramos de grasa:  311.111111111111
Gen 1: mejor individuo (cal_diff, cost) = (37.0, 85.25)
Gen 10: mejor individuo (cal_diff, cost) = (4.0, 86.15)
Gen 20: mejor individuo (cal_diff, cost) = (4.0, 86.15)
Gen 30: mejor individuo (cal_diff, cost) = (1.0, 61.4)
Gen 40: mejor individuo (cal_diff, cost) = (1.0, 61.4)
Gen 50: mejor individuo (cal_diff, cost) = (1.0, 61.4)
Gen 60: mejor individuo (cal_diff, cost) = (1.0, 61.4)
```

En la segunda salida se observa el **frente de Pareto** que agrupa las soluciones no dominadas obtenidas al final del proceso. El algoritmo encontró 100 combinaciones distintas que representan diferentes compromisos entre la precisión calórica y el costo. Dentro de estas soluciones, algunas logran un ajuste prácticamente perfecto en calorías, con solo 1 de diferencia, aunque con un costo de alrededor de 61.4. Otras soluciones, en cambio, permiten abaratar el costo de la dieta hasta unos 50.85, pero a cambio de aceptar una diferencia mayor frente a la meta calórica (por ejemplo, 25 calorías)

```
Número de soluciones en el frente de Pareto: 100
Muestreo de soluciones (cal_diff, cost):
1 (1.0, 61.4)
2 (1.0, 61.4)
3 (11.0, 51.75)
4 (11.0, 51.75)
5 (25.0, 50.85)
```

Este código tiene en cuenta los costos porque en la función de evaluación de cada individuo no solo se calcula la diferencia entre las calorías totales de la dieta propuesta y la meta, sino que también se calcula el costo total de los alimentos seleccionados. Ambos valores se devuelven como un par de objetivos: el primero mide qué tan cerca está la solución de las calorías requeridas, el segundo representa el costo acumulado de la dieta.

3. Verdadera democracia. Suponga que usted es el jefe de gobierno y está interesado en que pasen los proyectos de su programa político. Sin embargo, en el congreso conformado por 5 partidos, no es fácil su tránsito, por lo que debe repartir el poder, conformado por ministerios y otras agencias del gobierno, con base en la representación de cada partido. Cada entidad estatal tiene un peso de poder, que es el que se debe distribuir. Suponga que hay 50 curules, distribuya aleatoriamente, con una distribución no informe entre los 5 partidos esas curules. Defina una lista de 50

entidades y asígneles aleatoriamente un peso político de 1 a 100 puntos. Cree una matriz de poder para repartir ese poder, usando AGs.

Para resolver el problema de **“Verdadera democracia”**, se planteó un algoritmo que primero asigna de manera aleatoria y no uniforme las 50 curules del congreso entre los 5 partidos políticos. Luego, se definieron 50 entidades estatales, cada una con un peso de poder asignado aleatoriamente entre 1 y 100, representando la importancia política de cada cargo. Posteriormente, se aplicó un algoritmo genético (AG) para repartir los ministerios y agencias (poder político) entre los partidos, buscando que la distribución resultante guarde proporcionalidad con la representación que ya poseen en curules, lo cual asegura un balance entre legitimidad electoral y gobernabilidad.

En los **resultados**, la primera tabla muestra cómo se distribuyeron las curules entre los 5 partidos, reflejando la fuerza legislativa de cada uno.

```
Distribución inicial de curules (poder en el Congreso):
  Partido  Curules
Partido 1       23
Partido 2        1
Partido 3        5
Partido 4        2
Partido 5       19
```

La segunda tabla presenta las 50 entidades estatales con sus respectivos pesos de poder, generados aleatoriamente.

```
 Lista completa de entidades (50) y sus pesos (aleatorios 1-100)
    Entidad  Peso
  Entidad 1    95
  Entidad 2    97
  Entidad 3    87
  Entidad 4    14
  Entidad 5    10
  Entidad 6     8
  Entidad 7    64
  Entidad 8    62
  Entidad 9    23
 Entidad 10    58
 Entidad 11     2
 Entidad 12     1
 Entidad 13    61
 Entidad 14    82
 Entidad 15     9
 Entidad 16    89
 Entidad 17    14
 Entidad 18    48
 Entidad 19    73
 Entidad 20    31
```

Finalmente, la tabla consolidada expone cómo quedaron asignados tanto los curules como las entidades a cada partido, junto con la proporción respectiva para cada partido, lo que permite visualizar si la distribución fue equitativa y consistente con el objetivo.

```
=== Comparación final de poder y curules ===
   Partido  Curules  Porc_Curules  Poder_Entidades  Porc_Poder
Partido 1       23          46.0             1049   45.867949
Partido 2        1           2.0               47    2.055094
Partido 3        5          10.0              227    9.925667
Partido 4        2           4.0               96    4.197639
Partido 5       19          38.0              868   37.953651
```

Se puede apreciar una cercanía entre las proporciones del número de curules y el poder asignado en las entidades.
Por ejemplo, para el **partido 1**:

$$\frac{23}{50} = 0.46\ (46\%)$$

mientras que su poder en entidades es:

$$\frac{1049}{2287} \approx 0.458\ (45.8\%)$$

Es decir, ambas proporciones son muy cercanas, alrededor del **46%**.