Cómputo Evolutivo

Cuckoo Search aplicado a un portafolio financiero

Gomez Elizalde Alexys - 316086189

Fecha de entrega: 28 / Noviembre / 2024

1. Introducción

Durante el curso estuvimos estudiando diferentes áreas y aspectos del computo evolutivo, entre estos temas se mencionó sobre los algoritmos bioinspirados. Estos algoritmos se inspiran en fenómenos naturales, como el comportamiento de animales, procesos biológicos y características fisiológicas, para diseñar técnicas que resuelvan problemas de manera eficiente y creativa. Para este proyecto, decidí investigar sobre los algoritmos bioinspirados y sus aplicaciones en diversos problemas de la vida real. Durante mis estudios previos, había oído hablar de algoritmos bioinspirados populares, como las redes neuronales, el algoritmo de colonia de hormigas y hasta métodos curiosos como la horda de zombis. Buscaba un algoritmo menos conocido que fuera interesante de estudiar, implementar y, de ser posible, entretenido. Asi fue como tras varias horas de buscar llegué al algoritmo llamado **Cuckoo Search**.

1.1. Cuckoo Search

Desarrollado en 2009 por **Xin-She Yang y Suash Deb**. Su se basa en el parasitismo de cría de algunas especies de cucos. Estas aves ponen sus huevos en nidos ajenos para que las aves dueñas cuiden y críen esos polluelos al eclosionar el huevo. Al eclosionar, los polluelos del cuco pueden incluso expulsar los huevos o polluelos originales del nido, asegurando más recursos para sí mismos. El algoritmo emula este comportamiento mediante un conjunto de reglas idealizadas:

1. Cada cuco pone um huevo a la vez y lo deposita en un nido elegido aleatoriamente.
2. Los mejores nidos, definidos por la calidad de sus huevos (soluciones), se llevan a la siguiente generación.
3. El número de nidos disponibles es fijo, y el huevo depositado por un cuco debe ser descubierto por el ave anfitriona con una probabilidad de p_a, en el intervalo $[0, 1]$. En este caso, el ave anfitriona puede deshacerse del huevo o simplemente abandonar el nido y construir uno completamente nuevo.

Una característica distintiva del Cuckoo Search es el uso de los llamados **vuelos de Lévy**, que permiten explorar el espacio de búsqueda de manera eficiente y evitan que el algoritmo quede atrapado en óptimos locales.

1.2. Vuelo de Lévy

El vuelo de Lévy es una técnica de búsqueda basada en pasos aleatorios cuya longitud sigue una distribución de Lévy, caracterizada por generar saltos largos ocasionales entre pasos más cortos. Este

mecanismo equilibra la **búsqueda local** y la **exploración global**, controlado por el parámetro de conmutación p_a.
En el algoritmo, el vuelo de Lévy se define como:

$$x_i^{t+1} = x_i^t + \alpha L(s, \lambda),$$

donde α es un factor de escala del tamaño de los pasos, y $L(s, \lambda)$ representa la distribución de Lévy. Este proceso permite explorar regiones del espacio de búsqueda alejadas de las soluciones actuales, aumentando la probabilidad de encontrar soluciones globalmente óptimas.

1.3. Portafolio Financiero

Para este proyecto, decidí aplicar el CS en un problema de optimización de portafolio financiero. El objetivo principal es asignar capital a diferentes activos financieros de manera que se maximice la rentabilidad esperada y se minimice el riesgo, representado por la varianza de los retornos.
El modelo de optimización está basado en la siguiente función de utilidad:

$$U = E(R) - \lambda \cdot \text{Var}(R),$$

donde:

- $E(R)$ es la rentabilidad esperada del portafolio.
- $\text{Var}(R)$ es la varianza (riesgo) de los retornos.
- λ es el parámetro de aversión al riesgo.

El CS se utilizó para encontrar la combinación óptima de pesos asignados a los activos, cumpliendo con las restricciones de que la suma de los pesos debe ser igual a 1 y que cada peso esté en el intervalo $[0, 1]$.

2. Objetivo

El propósito de este proyecto es implementar el Cuckoo Search para optimizar un portafolio financiero, logrando generar una asignación de capital eficiente, que cumpla con las restricciones del modelo. Además de experimentar con el número de nidos para ver que tanto puede afectar en la búsqueda del óptimo.

2.1. Implementación

El código esta dividido en el algoritmo de Cuckoo Search 'CS.py' y su implementación a el portafolio 'portafolio.py' para facilitar la lectura del código. En 'CS.py' encontramos la implementación del algortimo junto con el vuelo de Lévy.

El código del proyecto se dividió en tres archivos principales para facilitar su lectura y mantenimiento:

- **CS.py**: Contiene la implementación del algoritmo CS, incluyendo los vuelos de Lévy. Está implementación del algoritmo Cuckoo Search fue diseñada y adaptada específicamente para el problema de optimización de portafolios financieros, basada en los principios fundamentales descritos en el artículo de *'Cuckoo Search and Firefly Algorithm: Overview and Analysis'* agregando los puntos para la localización del mejor nido absoluto y el se uso el metodo de distribuciones de Dirichlet para que la suma de los nidos nos diera 1.
- **portafolio.py**: Aplica el algoritmo al problema de optimización de portafolio financiero, integrando la función de utilidad y las restricciones.
- **graficación.py**: Aquí se encuentra el código correspondiente a las funciones necesarias para la creación de las gráfica. Se calcula la evolución del promedio de utilidades por generación y la evolución de la máxima utilidad por generación.

3. Experimentación

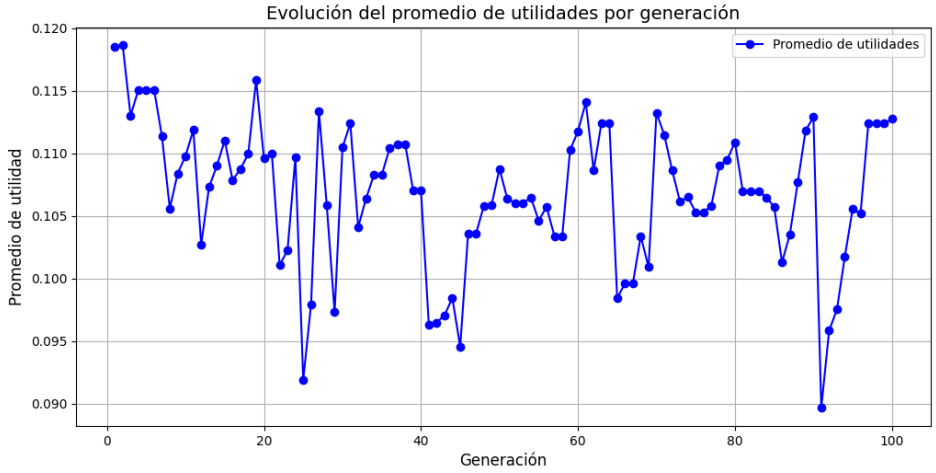
Decidí hacer la experimentación con los siguientes parámetros para el algoritmo:
número de nidos = 2, 3, 5 y 10
número de generaciones = 100
Tasa de descubrimiento (pa) = 0.30
Parámetro de aversión al riesgo (lambda_risk) = 0.5
El único parámetro que tiene una razón de su elección es la tasa de descubrimiento, según el artículo original de Yang y Deb (2009) sobre Cuckoo Search, p_a generalmente se encuentra en el rango del 25 % al 30 % para problemas generales de optimización. Este valor específico se seleccionó para priorizar un equilibrio entre estabilidad y variabilidad en las soluciones. Los demás parámetros los elegí para facilitar el análisis de los resultados y tener una ejecuciones rápidas.

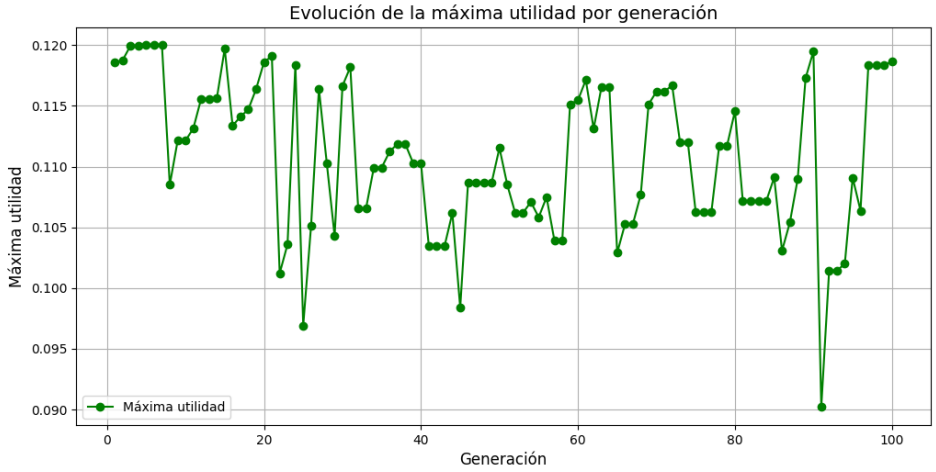
Y para el portafolio los siguientes:
Número de activos en el portafolio = 3
Rentabilidad esperada de cada activo = [0.10, 0.15, 0.12]
Matriz de covarianza = $\begin{bmatrix} 0.05 & 0.02 & 0.01 \\ 0.02 & 0.06 & 0.03 \\ 0.01 & 0.03 & 0.04 \end{bmatrix}$ Estos parámetros se decidieron de manera aleatoria imaginando que son los valores de un portafolio real pequeño.

3.1. Resultados

Se hicieron 5 ejecuciones con los parámetros de nidos de 2, 3, 5 y 10 y una extra con la tasa de descubrimiento de 0.50 y 5 nidos. Los resultados fueron los siguientes:

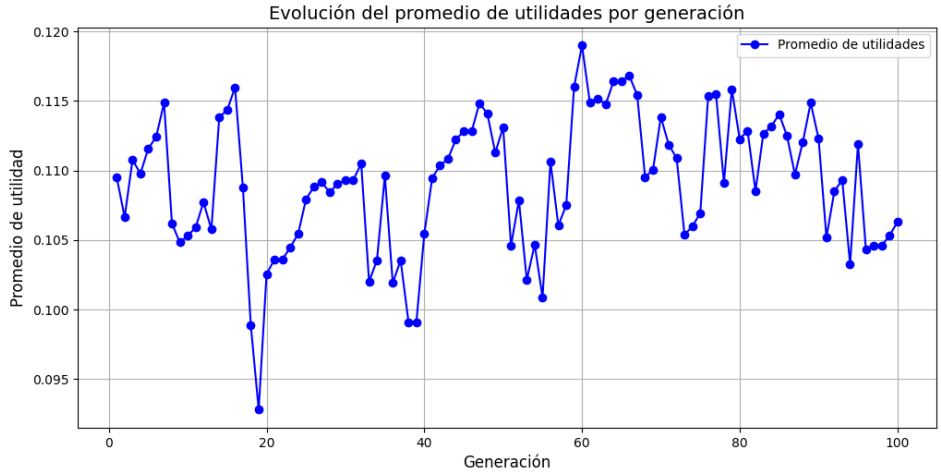
2 nidos

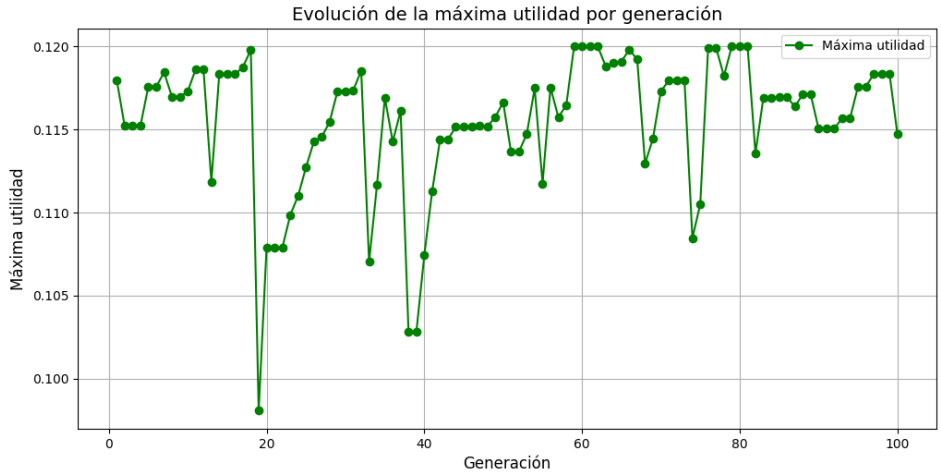





```
------Parámetros utilizados------
Número de nidos: 2
Generaciones: 100
Tasa de descubrimiento: 0.3
Mejor asignación de pesos (global): [0.20973238 0.49076174 0.29950588]
Utilidad del portafolio óptimo (global): 0.12
Mejor asignación de pesos (absoluto): [0.20973238 0.49076174 0.29950588]
Utilidad del portafolio óptimo (absoluto): 0.12
------------
```

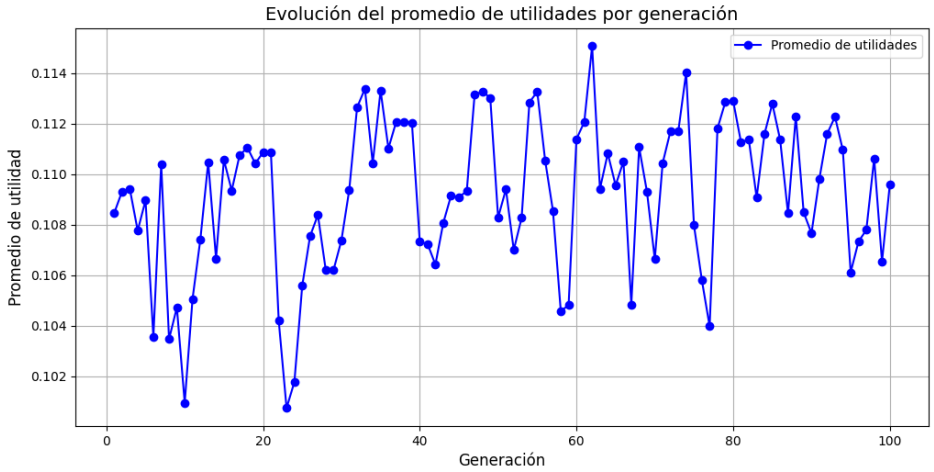
3 nidos

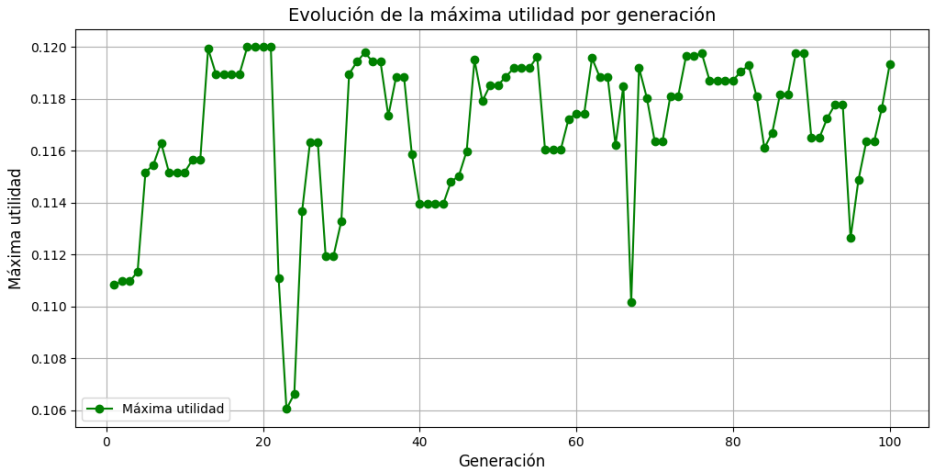





```
------Parámetros utilizados------
Número de nidos: 3
Generaciones: 100
Tasa de descubrimiento: 0.3
Mejor asignación de pesos (global): [0.31460003 0.27881211 0.40658786]
Utilidad del portafolio óptimo (global): 0.11999931799427901
Mejor asignación de pesos (absoluto): [0.31460003 0.27881211 0.4065878
Utilidad del portafolio óptimo (absoluto): 0.11999931799427901
------------
```

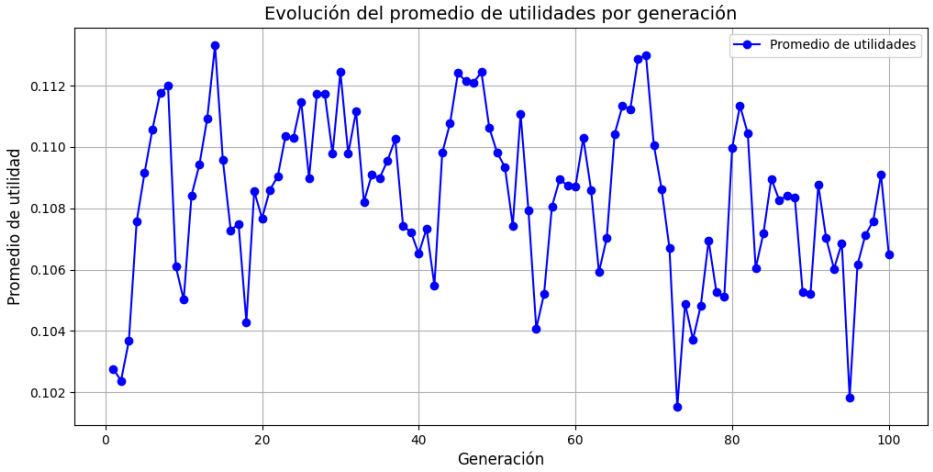
5 nidos

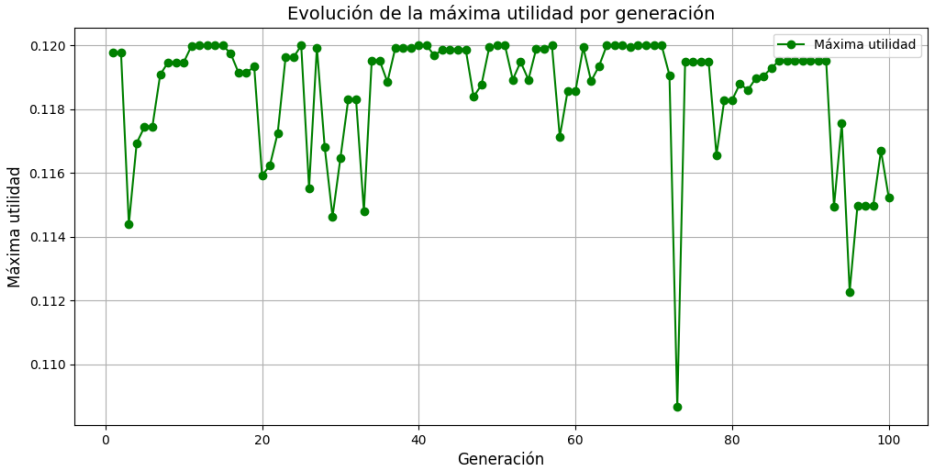





```
alexys@alexys-Inspiron-15-3525:~/Uni/Evolutivo/PCE$ python3 graficación.py
------Parámetros utilizados------
Número de nidos: 5
Generaciones: 100
Tasa de descubrimiento: 0.3
Mejor asignación de pesos (global): [0.68101564 0.16736906 0.1516153 ]
Utilidad del portafolio óptimo (global): 0.11999994691933381
Mejor asignación de pesos (absoluto): [0.68101564 0.16736906 0.1516153 ]
Utilidad del portafolio óptimo (absoluto): 0.11999994691933381
------------
```

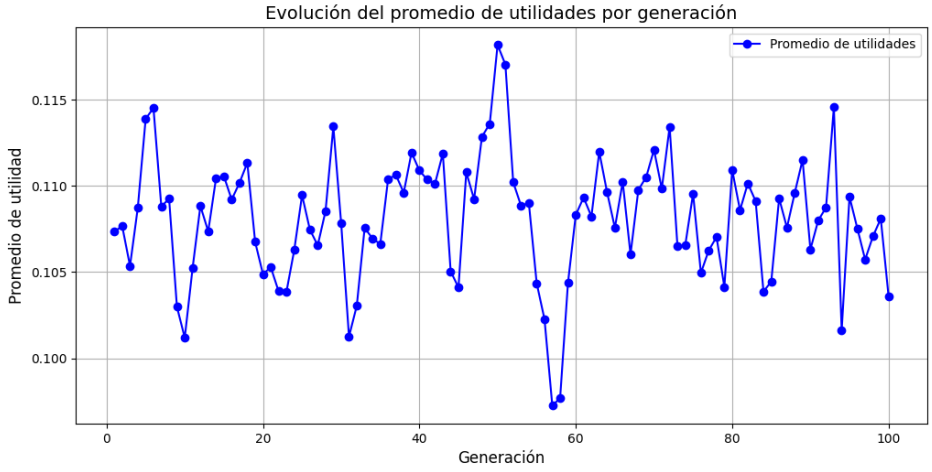
10 nidos

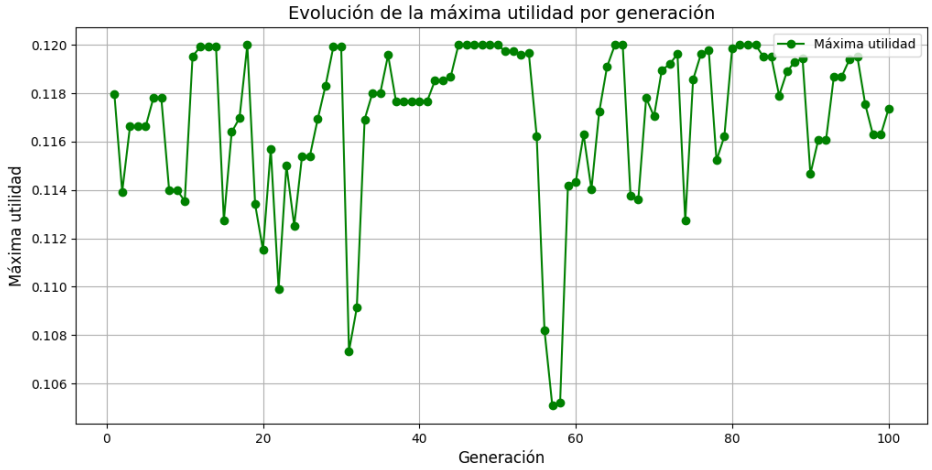





```
------Parámetros utilizados------
Número de nidos: 10
Generaciones: 100
Tasa de descubrimiento: 0.3
Mejor asignación de pesos (global): [0.34402629 0.20656235 0.44941136]
Utilidad del portafolio óptimo (global): 0.12
Mejor asignación de pesos (absoluto): [0.34402629 0.20656235 0.44941136]
Utilidad del portafolio óptimo (absoluto): 0.12
------------
```

Tasa de descubrimiento de 50

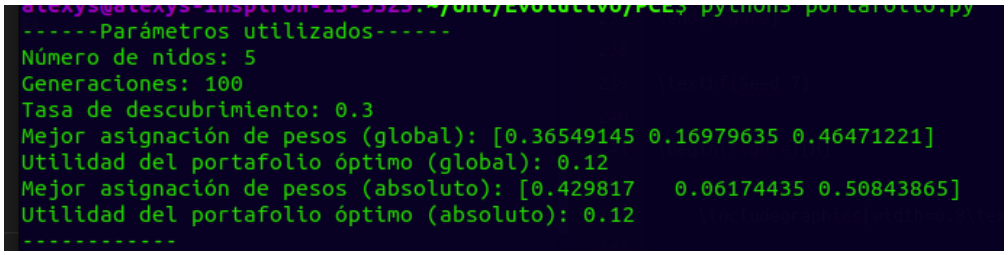





```
------Parámetros utilizados------
Número de nidos: 5
Generaciones: 100
Tasa de descubrimiento: 0.5
Mejor asignación de pesos (global): [0.17259182 0.09973474 0.72767344]
Utilidad del portafolio óptimo (global): 0.12
Mejor asignación de pesos (absoluto): [0.17259182 0.09973474 0.72767344]
Utilidad del portafolio óptimo (absoluto): 0.12
------------
```

3.2. Análisis

Para hacer el análisis necesitamos entender que nos esta devolviendo este programa, para el ejemplo pondré la terminal de una ejecución cualquiera.



Tenemos la **mejor asignación de pesos**, que en esta caso es *[0.36549145 0.16979635 0.46471221]* lo que significa que el %36.549145 de nuestra inversión total se iría al primer activo, el %16.979635 se iría al segundo activo y el %46.471221 se iría al tercer activo. Después tenemos la **Utilidad del portafolio óptimo**, en este caso es *0.12* que es el valor que representa la aptitud del portafolio. En esta caso el global y el absoluto tienen la misma utilidad pero la asignación de pesos es diferente.
Analizando los casos anteriores podemos darnos cuenta que el algoritmo alcanzo un óptimo para este caso del problema de 0.12.
A pesar de que las utilidades finales en varios casos son iguales las asignaciones de los pesos para los activos son distintos entre los experimentos lo que implica que hay diferentes combinaciones de pesos que pueden alcanzar las misma utilidad.
Las gráficas tienen muchos picos por el vuelo de Lévy.
En los casos en los que no llego al óptimo estuvo cerca de hacerlo.
Fue un caso muy sencillo para el algoritmo.

4. Conclusiones

El proyecto permitió explorar el funcionamiento del algoritmo bioinspirado **Cuckoo Search** en el ámbito de la optimización financiera, específicamente en la asignación óptima de portafolios. A continuación, se destacan los puntos principales derivados del proyecto: El algoritmo demostró ser una herramienta eficaz para abordar problemas de optimización. Su capacidad de explorar el espacio de soluciones mediante los vuelos de Lévy permitió un equilibrio entre la búsqueda global y local, maximizando la utilidad del portafolio bajo las restricciones. La tasa de descubrimiento p_a influyó en el balance entre la exploración de nuevas regiones y la explotación de áreas prometedoras. Un valor de 0.30 mostró un buen rendimiento general, mientras que un incremento a 0.50 resultó en una mayor exploración, aunque con un pequeño aumento en la dispersión de los resultados. El uso del CS en el modelo financiero permitió manejar eficientemente restricciones prácticas, como la suma de pesos igual a 1 y la asignación dentro de límites específicos. Esto muestra su potencial

para aplicaciones reales en finanzas y otras áreas.

5. Fuentes utilizadas para la investigación

Yang, X.–S. (2014). Cuckoo Search and Firefly Algorithm: Overview and Analysis. Springer International Publishing. Articulo

Baeldung. (n.d.). *Cuckoo Search*. Baeldung. March 22, 2023, de https://www.baeldung.com/cs/cuckoo-search