Optimización multiobjetivo para la selección de portafolios

Sara Gallego Villada & Hernán Octavio Moreno Mora
Departamento de ciencias e ingeniería , Universidad EAFIT, Medellín, Colombia
(Dated: September 10, 2025)

Resumen. Este proyecto busca analizar los algoritmos NSGA-II y MOEA/D de optimización multiobjetivo para la selección de portafolios, considerando criterios como rentabilidad y riesgo, donde se busca máximizar rentabilidad y minimizar riesgo. Se evaluarán ambas estrategias de optimización y se darán ejemplos en Matlab, con el fin de desarrollar herramientas más precisas y eficientes para la selección de activos de inversión.
Palabras clave: optimización multiobjetivo [Wik21], NSGA-II, selección de portafolios, rentabilidad, riesgo, algoritmo genético.

I. INTRODUCCIÓN

La selección de portafolios es un problema fundamental en la gestión de inversiones financieras. La elección de activos y la asignación de pesos a cada uno de ellos es un desafío importante que debe enfrentar cualquier inversor o gestor de fondos. En un mercado financiero altamente competitivo y en constante cambio, la selección de portafolios es un proceso crítico que puede tener un impacto significativo en los rendimientos de inversión y en la gestión del riesgo.

El problema de selección de portafolios se puede abordar desde una perspectiva de optimización multiobjetivo. En este enfoque, el objetivo es encontrar una solución que maximice la rentabilidad y minimice el riesgo de la cartera de inversión. Sin embargo, encontrar la combinación óptima de activos para alcanzar estos objetivos no es una tarea fácil debido a la complejidad y la incertidumbre del mercado financiero.

El objetivo de este proyecto es abordar el problema de selección de portafolios utilizando dos algoritmos de optimización multiobjetivo: NSGA-II [Mäk08] y MOEA/D. El primero es un algoritmo de selección basado en el concepto de dominancia no dominada, mientras que el segundo se basa en la descomposición de los objetivos en subproblemas más simples.[Gon15]

La implementación de estos algoritmos permitirá evaluar su eficacia en la selección de portafolios en términos de la calidad de la solución, el tiempo de ejecución y la escalabilidad. Se utilizará un conjunto de datos de valores históricos de activos financieros para evaluar y comparar el rendimiento de ambos algoritmos.

II. OPTIMIZACIÓN MULTI-OBJETIVO

La optimización multi objetivo es un área de toma de decisiones de criterio múltiple que se ocupa de los problemas de optimización matemática que involucran más de una función objetivo para ser optimizado simultáneamente. Esta es un tipo de optimización vectorial que se ha aplicado en muchos campos de la ciencia, incluidos la ingeniería, la economía y la logística, donde se deben tomar decisiones óptimas en presencia de compensaciones entre dos o más objetivos en conflicto. En problemas prácticos, puede haber más de tres objetivos.

Este problema puede ser formulado como:

$$\min_{x\in X}(f_1(x), f_2(x), \dots, f_k(x))$$

Donde $k \geq 2$ es el número de objetivos y X es el conjunto factible de vectores de decisión, que normalmente es $X \subseteq \mathbb{R}^n$ pero depende del dominio de aplicación n-dimensional. El conjunto factible suele estar definido por algunas funciones de restricción. Además, la función objetivo con valores vectoriales a menudo se define como

$$f : X \to \mathbb{R}^k$$
$$x \mapsto \begin{pmatrix} f_1(x) \\ \vdots \\ f_k(x) \end{pmatrix}$$

Si se quiere maximizar alguna función objetivo, es equivalente a minimizar su negativo.

En la optimización multiobjetivo, normalmente no existe una solución factible que minimice todas las funciones objetivo simultáneamente. Por lo tanto, se presta atención a las soluciones óptimas de Pareto; es decir, soluciones que no se pueden mejorar en ninguno de los objetivos sin degradar al menos uno de los otros objetivos. En términos matemáticos, una solución fáctible $x_1 \in X$ se dice que (Pareto) domina otra solución $x_2 \in X$, si:

- $\forall i \in \{1, \dots, k\}, f_i(x_1) \leq f_i(x_2)$
- $\exists i \in \{1, \dots, k\}, f_i(x_1) < f_i(x_2)$.

Una solución $x^* \in X$ se llama óptimo de Pareto si no existe otra solución que lo domine. El conjunto de resultados óptimos de Pareto, denotados X^*, se denomina frente de Pareto, frontera de Pareto o límite de Pareto.

Muchos problemas de toma de decisiones del mundo real tienen varios objetivos, a menudo en conflicto, y pueden reducirse a una optimización de criterios múltiples.

A. Uso en la selección de portafolios

Para la selección de portafolios es muy común hacer uso de la optimización multi objetivo cuando hay dos objetivos en conflicto: la busqueda de que el valor esperado de los rendimientos de la cartera sea lo más alto posible y el deseo de tener un riesgo lo más bajo posible, medido este por la desviación estándar de los rendimientos de la cartera. Este problema a menudo se representa mediante un gráfico en el que la frontera eficiente muestra las mejores combinaciones de riesgo y rendimiento esperado que están disponibles, y en el que las curvas de indiferencia muestran las preferencias del inversor por varias combinaciones de riesgo y rendimiento esperado. El problema de optimizar una función del valor esperado (primer momento) y la desviación estándar (raíz cuadrada del segundo momento central) del rendimiento de la cartera se denomina modelo de decisión de dos momentos.

III. EL MODELO MARKOWITZ DE VARIANZA MEDIA (MV)

Como una introducción al problema, vamos a mirar en detalle uno de los modelos mas utilizados en la selección de portafolios, El modelo Markowitz de Varianza Media. Primero, algunos conceptos generales:

A. Tasas de retorno de un portafolio

Llamamos *activo* cualquier instrumento financiero que puede ser comprado y vendido: acciones, bonos, CDTs, etc. Si copramos un *activo* por x_0 pesos y luego lo vendemos por x_1 pesos, llamamos R al retorno del activo: $R = \frac{x_0}{x_1}$. La *tasa de retorno* está dada por:

$$r = \frac{x_1 - x_0}{x_0} = R - 1$$

y por lo tanto $x_1 = Rx_0$ y $x_1 = (1 + r)x_0$

B. Ventas en corto o short selling

Es una técnica que permite vender un activo que no poseemos. Para ello, su corredor de bolsa (broker) puede vender una cierta cantidad de activos que usted no posee, pero el corredor si, de tal forma que usted se convierte en deudor del corredor por dicha cantidad de activos. Su deuda no es pesos, sino en especie. Es decir usted está "corto" en el número de unidades del activo que fueron vendidas en su nombre. En su balance, esta cantidad aparece con un signo negativo. Nótese que por la venta del activo, usted recibió (se le acreditaron) x_0 pesos.
En cualquier momento, usted solicita al corredor el recomprar la misma cantidad de activos vendidos originalmente, lo cual tiene un costo x_1 pesos. Así usted pagó al corredor su deuda en especie. Si $x_1 < x_0$ usted obtuvo una ganancia en la operación; de lo contrario, obtuvo una pérdida. En otras palabras, si el precio del activo baja entre el momento de compra y el de venta, usted obtiene una utilidad, pues recompra por menos plata de la que había vendido. En caso contrario, si el precio del activo aumenta, ustede liquida una pérdida en la operación. Un "short seller" está "apostando" a que el valor de un activo bajará en el futuro.
La operación contraria también es posible: usted puede comprar un activo con dinero prestado por su corredor, y venderlo en el futuro. Si el activo subió de precio, usted gana, de lo contrario pierde. Es la operación de "venta en largo" o "long selling".

C. Portafolio

Suponga que usted desea invertir en n *activos* diferentes y dispone para ello de x_0 pesos. Asigna un valor $x_{0i} = w_i x_0$ a cada uno de los n activos, donde w_i es el peso asignado a cada uno de los activos comprados, tal que $\sum_{i=1}^{n} w_i = 1$. Estos es:

$$Inversion = \sum_{i=1}^{n} w_i x_0 = x_0 \sum_{i=1}^{n} w_i = x_0$$

Si llamamos R_i el retorno del activo i, entonces el retorno total de su portafolio es:

$$x_1 = \sum_{i=1}^{n} R_i w_i x_0 = x_0 \sum_{i=1}^{n} R_i w_i$$

y el retorno total de su portafolio será:

$$R = \sum_{i=1}^{n} R_i w_i$$

y la rata total de retorno de su portafolio estará dada por:

$$r = R-1 = (\sum_{i=1}^{n} R_i w_i)-(\sum_{i=1}^{n} w_i) = \sum_{i=1}^{n}(R_i-1)w_i = \sum_{i=1}^{n} r_i w_i$$

D. Markowitz [Was]

La idea es modelar las tasas de retorno como variables aleatorias, y el objetivo es determinar los pesos w_i de forma óptima. En el contexto de Markowitz, un conjunto óptimo de pesos, es aquel con el cual el portafolio alcanza una mínima tasa de retorno esperada (definida por usted) con mínima volatilidad, donde la varianza en la tasa de retorno de un activo es la medida de su volatilidad.

Sea r_i la tasa de retorno aleatoria asociada al activo i, para i=1,2...n, y definamos un vector aleatorio z:

$$z = \begin{pmatrix} r_1 \\ r_2 \\ . \\ . \\ . \\ r_n \end{pmatrix}$$

y sea $\mu_i = E(r_i)$, $m = (\mu_1, \mu_2, ..., \mu_n)^T$, y $cov(z) = \sum$, donde μ_i es el valor esperado (media) del retorno del activo i, m es el vector traspuesto de todos los valores de retorno esperados, y $\sum$ es la covarianza de los r_i.

Sea $w = (w_1, w_2, ..., w_n)^T$ los pesos asignados, entonces su tasa de retorno $r = \sum_{i=1}^{n} r_i w_i$ es también una variable aleatoria con media $m^T w$ y varianza $w^T \sum w\mu$. Sea μ_b la tasa de retorno aceptable. En tal caso la teoría de Markowitz para el portafolio óptimo será cualquier portafolio que resuelva el siguiente problema cuadrático:

Minimizar

$$min \frac{1}{2} w^T \sum w$$

Sujeto a

$$m^T w \geq \mu_b$$

y

$$e^T w = 1$$

donde e es un vector de unos. Las condiciones KKT para este problema, son:

$$0 = \sum w - \lambda m - \gamma e \tag{1}$$

$$\mu_b \leq m^T w, e^T w = 1, 0 \leq \lambda \tag{2}$$

$$\lambda^T (m^T w - \mu_b) = 0 \tag{3}$$

para algún $\lambda, \gamma, \in \mathbb{R}$

Como la matriz de covarianza es simétrica y definida positiva, sabemos que existe una solución al problema de minimización.

Sin embargo, múltiples estudios recientes plantean ciertas dudas en relación con el modelo de Markowitz, en el sentido de que sólo es válido si se cumplen ciertas condiciones, una de las cuales es que los retornos de los activos se distribuyen Normal, y existe evidencia de que esto no se cumple.

Es por ello que se han hecho modificaciones a lo largo del tiempo al modelo Markowitz, entre ellas la que se conoce como MVS que incluye la maximización de la asimetría (skewness) de las tasas de retorno sujeto a restricciones en la media y la varianza de la rata de retorno. De allí el nombre de Mean Variance Skewness (MVS). Debido a que las tasas de retorno no se distribuyen Normal, sino que son asimétricas (skewed), el modelo MVS es una mejora frente al modelo MV.

Otros modelos que han sido propuestos en los últimos años, incluyen NPGA2, NSGA-II, PESA, SPEA2, y e-MOEA, todos ellos en la categoría MOEA [**moea**], esto es Multi Objective Evolutionary Algorithms, también conocidos como algoritmos genéticos.

E. Algoritmos Genéticos [Wik]

Los algoritmos genéticos están inspirados en el proceso de la selección natural de los seres vivos. En su aplicación matemática, se utilizan 3 operadores genéticos: **Selección** o *Supervivencia del mejor*, **Reproducción** o *Crossover* y **Mutación**.

1. Selección

Los operadores de selección le dan preferencia a las mejores soluciones (cromosomas), permitiendo así que pasen sus *genes* a la siguiente generación del algoritmo. Las mejores soluciones se determinan por medio de una **función objetivo** antes de pasarlos al siguiente operador (reproducción). Existen diferentes métodos para determinar las mejores soluciones.

El proceso es el siguiente:

1. Normalización de los resultados obtenidos (fitness values) para la función objetivo (Suma = 1)
2. Acumulación de los valores de ajuste. Para un individuo, esta es la suma de sus propios valores de ajuste, mas los de todos los individuos anteriores.
3. Generación de un número aleatorio entre 0 y 1
4. Se selecciona el primer individuo cuyo valor de ajuste acumulado es mayor a dicho número aleatorio

Existen sin embargo otros métodos de selección, tales como los llamados de *aceptación estocástica*, de *ruleta*, *selección por rango*, selección por torneo, elitista, de Boltzman y otros.

2. Reproducción

La reproducción o *crossover* toma más de una solución padre y genera una solución hijo a partir de ellas, bajo el principio de que tomando partes de las mejores soluciones, es posible producir una solución mejor.

Algunos de los métodos propuestos para la Reproducción, son los siguientes:

1. CX: de ciclos
2. OX2: basados en un orden
3. POS: basados en la posición
4. VR: recombinación por votación
5. AP: de posiciones alternantes
6. MX: por mezcla (merge)

3. Mutación

Este operador incentiva la diversidad genética entre las soluciones con el fin de evitar que el algoritmo converja a un mínimo local. Esto se logra evitando que las posibles soluciones estén muy cercanas entre si. Existen varios métodos para las mutaciones, desde el cambio aleatorio de bits poco significativos, hasta métodos complejos que cambian genes de una solución con valores aleatorios provenientes de una distribución uniforme o de una normal.:

Métodos de mutación son:

1. De cadena de bits: cambiar un bit de forma aleatoria
2. De números reales
3. Sin tener en cuenta las restricciones
4. Teniendo en cuenta las restricciones

Los tres operadores deben actuar conjuntamente, pues si sólo actúa uno de ellos, puede llegarse a soluciones no deseadas. Por ejemplo, si sólo se usa el de Selección, se llegará a múltiples copias de una misma solución. Si no se usa el de mutación, la solución podría converger a un mínimo local. Y si sólo se usa el de mutación, se obtendrá posibles soluciones aleatorias a traves de todo el conjunto de soluciones posibles.
Entre ellos, MOEA/D [Y20] es un algoritmo de optimización multiobjetivo basado en descomposción. Básicamente, se trata de descomponer (de allí el /D) el problema en sub-problemas, y utilizar el algoritmo evolucionado (EA) para resolver los subproblemas de forma simultána. Su objetivo, es encontrar un conjunto de soluciones de Pareto óptimas uniformemente distribuidas en el frente de Pareto.

IV. PLANTEAMIENTO MATEMÁTICO DEL PROBLEMA

Sea un conjunto de n activos financieros disponibles para la selección de portafolios. Cada activo tiene un rendimiento esperado y una medida de riesgo asociada, por ejemplo la varianza. Los pesos de cada activo en un portafolio se representan como un vector $w = [w_1, w_2, ..., w_n]$ con w_i denotando el peso del activo i en el portafolio, donde $0 \leq w_i \leq 1$ y $\sum w_i = 1$.

El objetivo es encontrar el conjunto de pesos w^* que maximice el rendimiento esperado y minimice el riesgo. Formalmente, estas funciones pueden escribirse como:
La función objetivo del rendimiento esperado:

$$maxF_1(w) = \sum_{i=1}^{n} w_i R_i$$

La función objetivo del riesgo, medida por la varianza:

$$minF_2(w) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \sigma_{ij}$$

donde R_i es la variable aleatoria asociada con la tasa de rendimiento del activo i y σ es la covarianza entre la tasa de rendimiento de los activos i y j. El vector w es el vector de pesos de los activos en un portafolio, donde $0 \leq w_i \leq 1$ y $\sum_i w_i = 1$.

Con estas definiciones, el problema de selección de portafolio se plantea como:

$$max_{w \in R^n} \{F_1(w), F_2(w)\}$$

Sujeto a:

$$\sum_{i=1}^{n} w_i = 1$$

$$0 \leq w_i \leq 1, i = 1, .., n$$

La solución a este problema de optimización multiobjetivo es un conjunto de soluciones óptimas no dominadas, es decir, soluciones que no pueden ser mejoradas en uno de los objetivos sin empeorar en el otro.

V. ALGORITMOS PROPUESTOS

A. NSGA-II

1. Introducción

El algoritmo NSGA-II pertenece a la familia de los llamados *EO algorithms*, esto es *Evolutionary Optimization algorithms* o algoritmos de optimización evolutiva. Se basan en una población inicial de soluciones las cuales participan en un cada iteración y evolucionan, o generan, una nueva población.

2. Descripción en el caso mono-objetivo

El algoritmo inicia la búsqueda de solución con una población inicial de soluciones, normalmente generada aleatoriamente, con un límite superior y un límite inferior específicos para cada variable. A continuación, el algoritmo inicia un proceso iterativo mediante el cual actualiza la población actual para crear una nueva población (generación) mediante el uso de cuatro operadores principales: selección, cruce (*crossover*), mutación y preservación élite. El proceso termina cuando se cumple alguna condición de parada previamente especificada.

- **Inicialización:** creación aleatoria de un conjunto de soluciones. Si se conoce un conjunto de soluciones posibles para un problema determinado, es conveniente iniciar con dicho conjunto.

- **Selección:** Cada una de las soluciones (miembros) de la población inicial se evalúa y el *operador de selección* escoge las mejores, asignándoles una probabilidad mayor de formar parte del grupo de reproducción (*mating pool*). Aunque existen diferentes métodos estocásticos para esta selección, el más utilizado es el de "torneos" (*tournament*), mediante al cual se escogen 2 soluciones aleatoriamente de la población, se evalúan, y la mejor de las dos pasa a formar parte del grupo de reproducción.

- **Cruce o *crossover***: Este operador toma aleatoriamente dos o más soluciones, llamadas "padres", del grupo de reproducción (*mating pool*) y crea una o más soluciones intercambiando información entre las soluciones padre. El operador de cruce se aplica con una probabilidad que indica la proporción de la población que participa en el cruce (p_c). El resto $(1 - p_c)$, se copian directamente a los hijos.

- **Mutación:** Cada hijo creado por el Cruce, es *perturbado* por el operador de mutación. Cada variable es *perturbada* con una probabilidad p_m normalmente igual a $\frac{1}{n}$, donde n es el número de variables, de tal forma que, en promedio, una variable es *perturbada* en cada solución. También se acostumbra usar una distribución Normal (Gaussiana) de varianza previamente definida y media el valor de la variable en el "hijo". Su ventaja es que es independiente de las otras soluciones de la población.

- **Preservación élite (*elitism*):** este operador combina la población anterior con la recientemente creada, y escoge las mejores soluciones dentro de la población combinada.

- **Terminación:** Definida por el usuario, puede ser un cierto número de generaciones, o en el caso de tener una meta, se termina una vez obtenida. Otro criterio, es la comparación estadística entre dos generaciones sucesivas para determinar la rata de convergencia alcanzada, esto es, si son muy similares entre si. En estudios recientes se utilizan las condiciones KKT para decidir la terminación.

3. Descripción en el caso Multiobjetivo

Un problema multiobjetivo se plantea como sigue:

$$Minimizar/Maximizar \;\; f_m(x), \;\; m = 1, 2, ..., M$$

Sujeto a :

$$g_j(x) \geq 0, j = 1, 2, ..., J$$

y

$$h_k(x) = 0, k = 1, 2, ..., K$$

con

$$x_i^L \leq x_i \leq x_i^U, i = 1, 2, ..., n$$

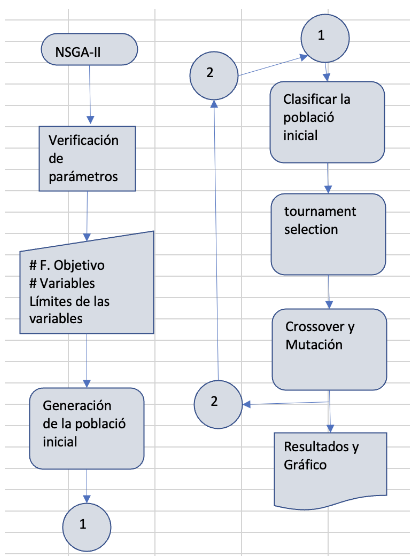
Entonces una solución $\mathbf{x} \in \mathbf{R}^n$ es un vector de n variables de decisión: $\mathbf{x} = (x_1, x_2, ..., x_n)^T$, y las soluciones que satisfacen las restricciones y los límites superior e inferior, constituyen un *espacio de soluciones factibles* $S \subset \mathbf{R}^n$.

En este caso, las funciones objetivo constituyen un espacio multi-dimensional de M dimensiones, llamado el *espacio objetivo*, $Z \in \mathbf{R}^M$. Para cada solución $\mathbf{x}$ en el espacio de soluciones factibles, existe un punto $\mathbf{z} \in \mathbf{R}^M$ en el espacio objetivo, el cual se denota como $f(\mathbf{x} = \mathbf{z} = (z_1, z_2, ..., z_M)^T$. Normalmente se llama *solución* al vector de variables y *punto* al vector objetivo correspondiente.

Las soluciones óptimas se definen bajo el concepto matemático de *orden parcial*, y la palabra usada es *dominancia*, la cual se define como sigue: x^1 domina a x^2 si, 1) La solución x^1 no es peor que la solución x^2 en todos los objetivos y 2) La solución x^1 es mejor que la solución x^2 en al menos un objetivo.

Se define entonces el *frente óptimo de Pareto*, como el conjunto de todos los puntos "no dominados", y sus pre-imagen correspondientes, se denominan las *soluciones óptimas de Pareto*.
La diferencia entonces del NSGA-II multiobjetivo frente al mono-objetivo explicado antes, consiste en que cada generación se clasifica en clases *no dominadas*, de tal forma que cada generación genera puntos en diferentes frentes no-dominados. Como en el caso anterior, en cada generación se descartan las soluciones cuya *crowding distance*, la cual se define como el espacio objetivo alrededor de una solución, no es ocupado por ninguna otra solución en la población.
El usuario tiene al final un *conjunto de soluciones posibles* y es su trabajo escoger la mejor de ellas, para lo cual debe aplicar criterios sujetivos en la mayoría de los casos.

Como ejemplo, se anexa un programa basado en la documentación de Aravind Seshadri [**Aravind**] que resuelve un problema de tres objetivos. El flujo del programa es el siguiente:



Gráfica diagrama NSGA-II

4. Descripción del programa

El Programa está escrito en Matlab Versión R2023a de 64 bits para Mac64. Realiza varias operaciones que se describen a continuación. El programa es de minimización de las funciones ingresadas. En caso de maximizar alguna de las funciones, se debe multiplicar por menos uno dicha función.

1. **Verificación de parámetros:** El programa verifica que en el llamado se haya ingresado correctamente el tamaño de la población inicial (mínimo 20) y el número de generaciones (mínimo 5). Verifica además que los números ingresados sean enteros. Pide a continuación ingresar el número de funciones objetivo, el número de variables de decisión y el rango de cada una de dichas variables. Los datos ingresados deben coincidir con los de las funciones objetivo en la función *evaluate_objective*.

2. **Generación de la población inicial:** *initialize_variables*. En la primera etapa, se crean los cromosomas con la información de los valores de las variables de decisión generadas aleatoriamente, y el valor de cada una de las funciones objetivo. Tiene entonces V + M elementos, donde V es el número de variables de decisión y M el número de funciones objetivo. Imprime los cromosomas y hace una pausa. **Nota:** Como la pblación se genera aleatoriamente, el programa dará resultados diferentes cada vez que se ejecuta. Se utiliza la función **rand(1)** de Matlab para la generación aleatoria.

3. **Clasificar la población inicial:** *non_domination_sort*. Clasificación usando el criterio de no-dominancia, siendo los factores de clasificación el *rank* y la *crowding distance*. Ambos valores se agregan al vector de cromosomas en esta etapa. Imprime los cromosomas y hace una segunda pausa.

4. **Tournament selection y reproducción:** *tournament_selection* y *replace_chromosome*. Este es el proceso evolutivo:

   - Selección de padres aptos para reproducirse
   - Aplicarles los operadores Crossover (Simulated Binary Crossover (SBX)) y Mutación polinomial. La probabilidad de Crossover es 0.9 y la de mutación es 1/n, donde n es el número de variables de decisión.
   - Seleccionar los mejores en la población combinada de padres e hijos, con el criterio del *rank* y el *crowding distance*
   - Crear una nueva población con el número original de individuos. Por lo tanto, se descarta la mitad.

5. **Crossover y mutación:** *genetic_operator*: El operador genético consiste de dos operaciones, a saber: *Simulated Binary Crossover (SBX)* y Mutación Polinomial

   - *Simulated Binary Crossover(SBX)*: Simula lo que pasa en la naturaleza, y opera como sigue:

     $$c_{1,k} = \frac{1}{2}[(1-\beta_k)p_{1,k} + (1+\beta_k)p_{2,k}]$$

     $$c_{2,k} = \frac{1}{2}[(1+\beta_k)p_{1,k} + (1-\beta_k)p_{2,k}]$$

     donde $c_{i,k}$ es el i-ésimo hijo con el k-ésimo componente, $p_{i,k}$ es el padre elegido y $\beta_k(\geq 0)$

es una muestra de números aleatorios generados con la siguiente función de densidad:

$$p(\beta) = \frac{1}{2}(\eta_c + 1)\beta^{\eta_c}, si 0 \leq \beta \leq 1$$

$$p(\beta) = \frac{1}{2}(\eta_c + 1)\frac{1}{\beta^{\eta_c} + 2}, si \beta \geq 1$$

La distribución puede obtenerse a partir de un número aleatorio u entre (0,1). η_c es el índice de la distribución para el *crossover*. Esto es:

$$\beta(u) = (2u)^{\frac{1}{\eta+1}}$$

$$\beta(u) = \frac{1}{[2(1-u)]^{\frac{1}{\eta+1}}}$$

- Mutación Polinomial:

$$c_k = p_k + (p_k^u - p_k^l)\delta_k$$

donde c_k es el hijo y p_k es el padre con p_k^u es el límite superior en el componente del padre y p_k^l es el límite inferior, y δ_k es una pequeña variación calculada con la distribución usando:

$$\delta_k = (2r_k)^{\frac{1}{\eta_m+1}} - 1$$

, si $r_k \leq 0.5$

$$\delta_k = 1 - [2(1-r_k)]^{\frac{1}{\eta_m+1}} - 1$$

, si $r_k \geq 0.5$

y r_k es un número aleatorio entre (0,1) y η_m es el índice de distribución de la mutación.

6. Impresión de resultados y generación del gráfico con el frente de Pareto de la solución.

El paquete consta de un programa principal (**NSGA2Final**) el cual debe ejecutarse con dos parámetros: el tamaño de la población (está en 200 individuos) y el número de generaciones (está en 500). La única restricción es que el tamaño mínimo de la población es de 20 individuos y como mínimo deben haber 5 generaciones. Y 7 funciones que se enumeran a continuación. Su funcionamiento fué detallado antes en este documento, y todas los códigos están debidamente comentados.

El programa y las funciones son:

1. **NSGA2Final:** Sólo llama a la función nsga_2, con dos parámetros: tamaño de la población y número de generaciones. Por defecto está en 200 y 200: **nsga_2(200,200)**
2. **nsga_2:** Verifica datos de entrada, lee la función objetivo, llama las demás funciones, muestra en pantalla el detalle de los cromosomas iniciales y finales, y genera los gráficos al terminar la optimización.
3. **objective_description_function:** pide el número de funciones objetivo, de variables de decisión y los rangos de cada una de ellas.
4. **initialize_variables:** Inicializa los cromosomas de la población inicial
5. **non_domination_sort_mod:** Hace la clasificación de no-dominancia
6. **tournament_selection:** Selecciona los padres de la próxima generación
7. **genetic_operator:** Produce los hijos a partir de los cromosomas padre seleccionados
8. **intermediate_chromosome:** Combina la población actual con la que acaba de generar
9. **replace_chromosome:** Reemplaza los cromosomas basado en el rango y la *crowding distance*

Se hicieron varias corridas del programa con diferentes conjuntos de funciones objetivo tomadas de el paper de Ran Cheng [**Cheng**] y otros con el fin de verificar su correcto funcionamiento. A continuación el resultado de dos corridas:

1. Funciones Viennet2:

$$f_1 = \frac{(x_1-2)^2}{2} + \frac{(x_2+1)^2}{13} + 3$$

$$f_2 = \frac{(x_1+x_2-3)^2}{36} + \frac{(x_1-x_2+2)^2}{8} - 17$$

$$f_3 = \frac{(x_1-4)^2}{175} + \frac{(x_2-x_1^2)^2}{17} - 13$$

$$-4 \leq x_1, x_2 \leq 4$$

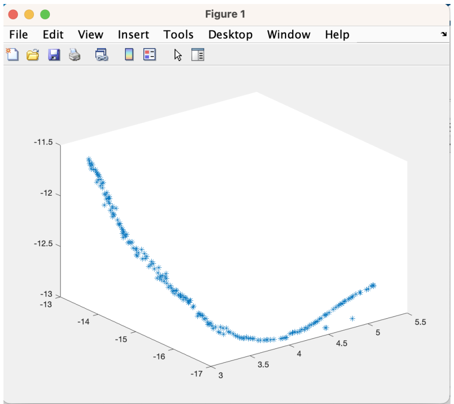
La mejor solución en este caso podría ser entonces:

$x_1 = 0.5001, x_2 = 2.5003$

con los siguientes valores para las funciones objetivo:

$f_1 = 5.0674, f_2 = -17.0, f_3 = -12.6321$

tiene rango 1 y su *crowding distance* es despreciable (inf).



Gráfica frontera de pareto ejemplo

2. Funciones DTLZ4:

$$f_1 = (1 + g(x)) * cos(x_1 * \frac{\pi}{2}) * cos(x_2 * \frac{\pi}{2}) * cos(x_3 * \frac{\pi}{2})$$

$$f_2 = (1 + g(x)) * cos(x_1 * \frac{\pi}{2}) * sen(x_2 * \frac{\pi}{2})$$

$$f_3 = (1 + g(x)) * sen(x_1 * \frac{\pi}{2})$$

con g(x):

$$g(x) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 + (x_3 - 0.5)^2$$

$$0 \leq x_1, x_2, x_3 \leq 1$$

La mejor solución en este caso sería:

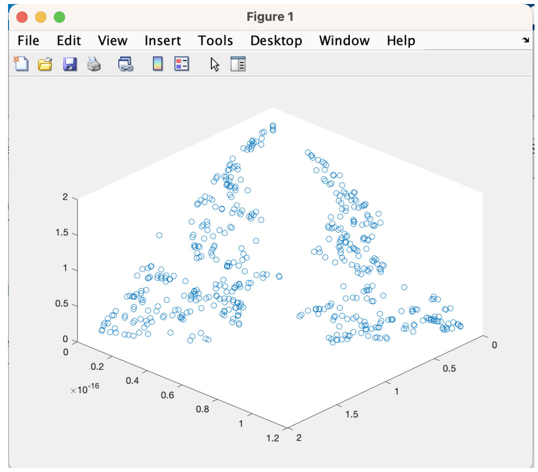
$x_1 = 0, x_2 = 0.0434, x_3 = 1.0$

con los siguientes valores para las funciones objetivo:

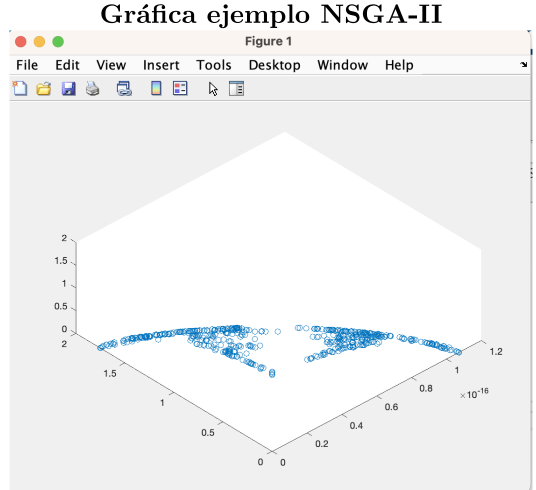
$f_1 = 0.0, f_2 = 0.1163, f_3 = 0$

tiene rango 1 y su *crowding distance* es despreciable (inf).

Dos ángulos de visualización del frente de Pareto son los siguientes:



Gráfica ejemplo NSGA-II



Gráfica ejemplo NSGA-II

El gráfico 3D nos muestra la dispersión esperada de acuerdo con la literatura de los Benchmark consultada.

B. MOEA/D(Algoritmo evolutivo multiobjetivo basado en descomposición)

Es un algoritmo de optimización multiobjetivo que se basa en la descomposición de un problema multiobjetivo en un conjunto de subproblemas escalares.

En MOEA/D, el espacio de búsqueda se divide en un conjunto de subproblemas escalares, y cada subproblema tiene como objetivo minimizar una función objetivo específica. Luego, se utiliza un algoritmo de búsqueda en cada subproblema para encontrar soluciones óptimas que minimicen la función objetivo correspondiente. El algoritmo de búsqueda utilizado en cada subproblema es generalmente un algoritmo de optimización mono-objetivo, como el algoritmo de optimización de diferencias evolutivas
MOEA/D emplea el enfoque de Tchebycheff para descomponer el problema multiobjetivo. Se utiliza un

conjunto de vectores de peso distribuidos uniformemente $\lambda_1, \lambda_2, ...\lambda_n$ para representar el peso de N subproblemas de optimización escalar descompuestos. Y el j-ésimo subproblema se denota por:

$$g^{te}(w|\lambda^j, z^*) = max_1 \leq_i \leq_m \lambda_i^j |f_i(w) - z_i^*|$$

donde λ_i es un vector de pesos no negativos y z_i es el vector de referencia. La idea es que λ_i indica la importancia relativa del i-ésimo objetivo, mientras que z_i representa la mejor solución conocida en términos del i-ésimo objetivo.

MOEA/D es útil para la selección de portafolios, ya que puede ser aplicado para buscar un conjunto de soluciones no dominadas que maximicen la rentabilidad y minimicen el riesgo de un portafolio. Además, MOEA/D es útil cuando las funciones objetivo están altamente correlacionadas y no son altamente conflictivas, lo que es común en la selección de portafolios.

1. *Algoritmo*

1. **Inicialización:** generar una población inicial de soluciones candidatas, de forma aleatoria o por métodos heurísticos

2. **Descomposición:** Descomponer el problema multi-objetivo en varios sub-problemas escalares de optimización, utilizando un conjunto de vectores de pesos. Cada vector de pesos corresponde a un sub-problema, y cada solución candidato, se asigna al sub-problema con el vector de pesos mas cercano.

3. **Optimización:** Optimizar cada sub-problema usando un método de búsqueda local, tal como el del gradiente descendente, el simplex, u otro. Los valores objetivos de las soluciones candidatas se evalúan en el sub-problema correspondiente.

4. **Actualización:** Actualizar la población de soluciones candidatas, escogiendo las mejores soluciones a cada sub-problema y aplicando operdores evolutivos, tales como *crossover* y mutaciones.

5. **Terminación:** el algoritmo termina al cumplir algún criterio de parada, tal como un numero máximo de iteraciones o un nivel de convergencia deseado.

Un diagrama de flujo del algoritmo, tomado del trabajo de Merdhad Taghian [AM15] y otros (A multiobjective evolutionary algorithm usign decomposition (MOEA/D) and its application in multipurpose multi-reservoir operations) es el siguiente:

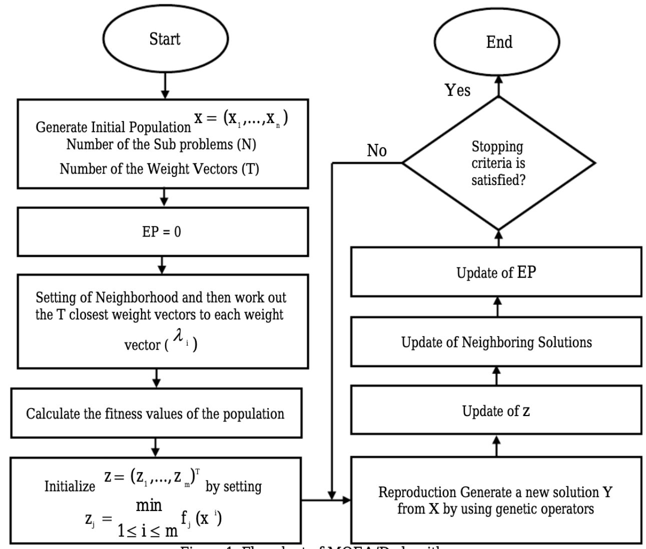
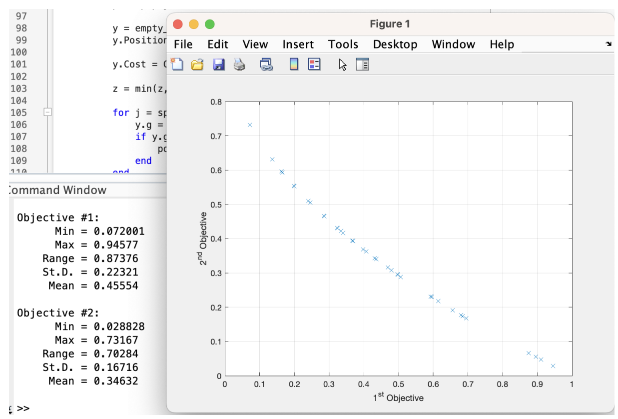


Figure 1. Flowchart of MOEA/D algorithm

Diagrama MOEA/D

A continuación, el resultado de un programa en Matlab para el método MOEA/D [Her15]



Ejemplo implementación MOEA/D

VI. METODOLOGÍA

- **Revisión bibliográfica:** realizar una revisión exhaustiva de la literatura sobre la selección de portafolios y los algoritmos NSGA-II y MOEA/D para comprender los conceptos básicos y las técnicas de optimización utilizadas.
- **Recopilación y preparación de datos:** recopilar datos históricos de precios de activos financieros relevantes y prepararlos para su uso en el modelo.
- **Definición de variables y restricciones:** definir las variables de decisión, como la asignación de pesos a cada activo en el portafolio, y las restricciones, como el presupuesto disponible y los límites de asignación de cada activo.

- **Implementación de los algoritmos:** implementar los algoritmos NSGA-II y MOEA/D en un entorno de programación como MATLAB o Python y ajustar los parámetros de los algoritmos para lograr una convergencia óptima.

- **Ejecución y comparación de los algoritmos:** ejecutar los algoritmos con los datos y las variables definidas y comparar los resultados para determinar cuál de los dos algoritmos es más efectivo para la selección de portafolios.

- **Análisis de resultados y conclusiones:** analizar los resultados obtenidos y determinar cuál de los dos algoritmos es el más adecuado para el problema de selección de portafolios y resumir las conclusiones y recomendaciones del proyecto.

- **Documentación y presentación:** documentar los resultados y las conclusiones en un informe y presentarlos a los interesados, como los inversores y los gerentes de cartera, para tomar decisiones informadas sobre la selección de portafolios.

VII. RESULTADOS

En esta sección se presentan los resultados computacionales obtenidos.
Se implementó una metodología basada en los algoritmos MOEAD (Multi-Objective Evolutionary Algorithm based on Decomposition) y NSGA-II (Nondominated Sorting Genetic Algorithm II) utilizando MATLAB para optimizar la selección de portafolios de inversión. Los resultados obtenidos ofrecen un conjunto de soluciones óptimas que representan diferentes combinaciones de activos en un portafolio.
Para la implementación del NSGA-II, se utilizaron datos históricos obtenidos de Yahoo Finance, que incluyeron precios de diferentes empresas como Apple, Tesla, Microsoft, Walmart, Amazon, entre otras. Estos datos históricos son fundamentales para realizar un análisis exhaustivo y capturar la variabilidad de los rendimientos de los activos a lo largo del tiempo. Debido a que los mercados de valores pueden tener diferentes cantidades de días de negociación, la cantidad de datos puede ser diferente para un número fijo período de tiempo. Por lo tanto, se cogió un período de tiempo del 2022 al 2023.

Primero se realizó una implementación usando la función de matlab gamultiobj.
De esta primera implementación se obtuvieron opciones de inversión, algunas de ellas son:

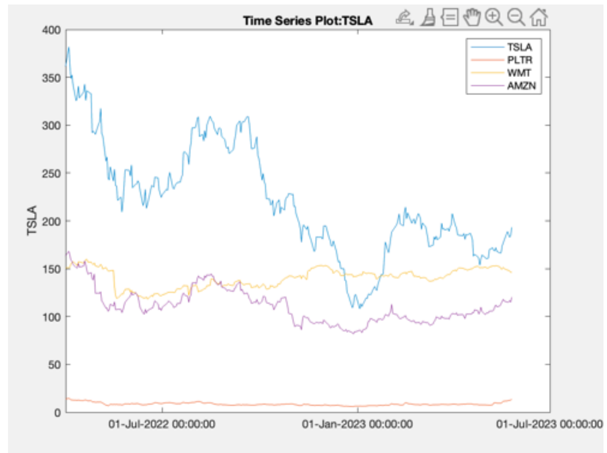
Tesla	B of A	Walmart	Amazon
2.1%	1.4%	66.9%	29.6%

Opción A portafolio

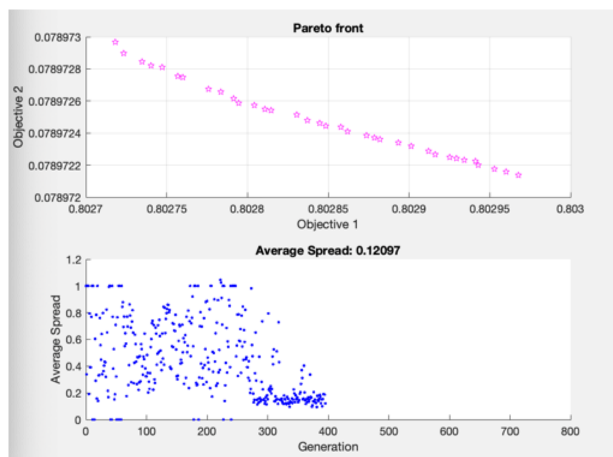
Tesla	B of A	Walmart	Amazon
68.8%	3.6%	4.8%	22.8%

Opción B portafolio

Y las gráficas resultantes fueron las siguiente:



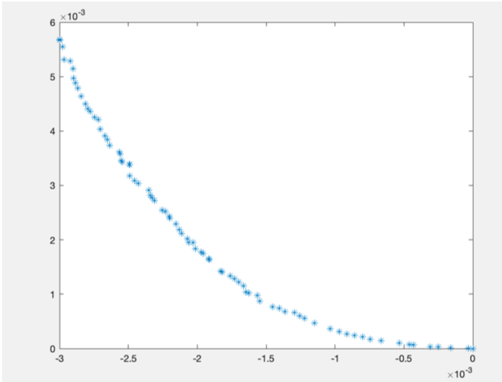
Gráfica serie de tiempo



Gráfica frontera de pareto NSGA-II

El objetivo 1 de las gráficas es máximizar el retorno y el objetivo 2 es minimizar el riesgo. Por lo tanto en el eje "x" podemos visualizar el retorno y el en eje y el riesgo. Las gráficas de la frontera de Pareto proporcionan una visualización clara de las diferentes opciones de portafolio que se pueden considerar en función de los objetivos de inversión. Estas gráficas permiten al inversor tomar decisiones informadas sobre las combinaciones de activos que mejor se ajusten a sus necesidades y preferencias de riesgo-retorno.

Posteriormente se hizo una segunda implementación de NSGA-II implementando todo el algoritmo desde cero y se obtuvieron los siguientes resultados:



Gráfica frontera de pareto NSGA-II 2

Donde el eje "x" es el retorno y el eje "y" es el riesgo. Adicionalmente se obtuvieron las siguientes opciones de inversiones:

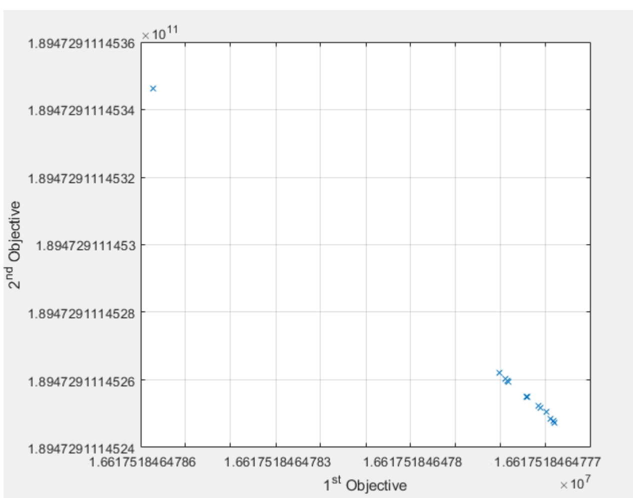
Tesla	B of A	Walmart	Amazon
30.08%	30.08%	9.76%	30.08%

Opción A portafolio

Tesla	B of A	Walmart	Amazon
22.6%	68.1%	6.8%	2.5%

Opción B portafolio

Para la implementación del algoritmo MOEA/D se utilizaron datos de Yahoo Finance.
El resuldato de la gráfica de pareto es como se ve acontinuación:



Gráfica frontera de pareto MOEA/D

El objetivo 1 de las gráficas es máximizar el retorno y el objetivo 2 es minimizar el riesgo. Por lo tanto en el eje x podemos visualizar el retorno y el en eje y el riesgo.

La gráfica de Pareto muestra claramente el compromiso entre los dos objetivos y su relación directamente proporcional. A medida que el riesgo baja se puede observar que el retorno también disminuye. Por lo tanto, disminuir el riesgo también implica disminuir las ganancias.

Por lo tanto, la toma de desiciones del inversor dependerá del riesgo que esté dispuesto a tomar a la hora de invertir.

Aunque nuestros algoritmos son matemáticamente correctos, y en efecto llegan a un frente de Pareto de soluciones óptimas, en la práctica los resultados no son aceptables como guía de inversión, a menos que el inversionista tenga muy claro su nivel de aversión al riesgo, y el horizonte de tiempo que desea analizar.

Como ejemplo, vamos a mirar el desempeño de un portafolio que contiene a Palantair, una compañía de tecnología que ha sido una estrella en los últimos meses, considerando los precios de los últimos 2 años:

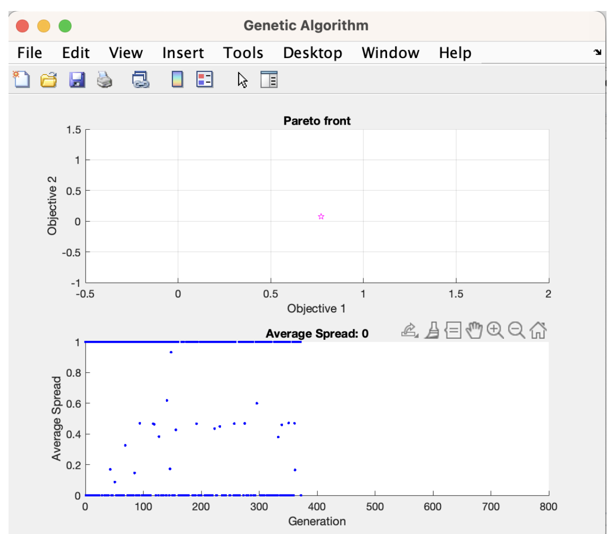
```
Los puntos del frente de Pareto son:
    1.0000    0.0003    0.0006    0.0000
    1.0000    0.0008    0.0001    0.0001
    1.0000    0.0003    0.0007    0.0000
    1.0000    0.0002    0.0008    0.0000
    1.0000    0.0005    0.0005    0.0000
    1.0000    0.0004    0.0006    0.0000
    1.0000    0.0001    0.0008    0.0000
    1.0000    0.0007    0.0003    0.0000
    1.0000    0.0003    0.0007    0.0000
    1.0000    0.0008    0.0002    0.0000
```

Como bien puede verse, la recomendación es invertir el 100% del portafolio en la acción de Tesla.

Si hacemos el análisis para lo que va del año 2023, obtenemos lo siguiente:

```
Los puntos del frente de Pareto son:
    0.2406    0.7112    0.0022    0.0470
```

En este caso, una solución única: 24% en Tesla, 71% en Palantair, 0.2% en Walmart y 5% en Amazon.



Creemos que en un futuro trabajo, deben incluirse otras variables tales como los fundamentales de las empresas consideradas, esto es, su nivel de endeudamiento, su participación de mercado, su crecimiento en períodos recientes.

VIII. CONCLUSIONES

En cuanto a los algoritmos utilizados, se observó que tanto MOEA/D como NSGA-II demostraron ser eficientes en la resolución del problema de selección de portafolios multiobjetivo. Ambos algoritmos fueron capaces de encontrar un conjunto diverso de soluciones óptimas, brindando opciones de inversión para diferentes perfiles de riesgo y rendimiento. Los resultados obtenidos brindaron una frontera de Pareto que permite a los inversionistas tomar decisiones informadas sobre la construcción de su portafolio, considerando tanto el rendimiento como el riesgo.

A pesar de los esfuerzos realizados, no se lograron obtener los resultados esperados. Durante el desarrollo del proyecto, surgieron varios desafíos y obstáculos que afectaron la calidad de los resultados. Estos desafíos podrían haber surgido debido a diversas razones, como errores en la implementación del algoritmo o suposiciones incorrectas en el diseño del modelo.
Es importante destacar que la optimización de la selección de portafolios es un desafío complejo y multifacético, y los resultados esperados pueden no ser alcanzados en todos los casos. La toma de decisiones en inversiones requiere un análisis exhaustivo y una comprensión profunda de los mercados financieros, los riesgos asociados y las preferencias individuales del inversionista.

En cuanto a trabajos futuros, se podrían explorar varias direcciones y mejoras en la implementación de los algoritmos. Una posible mejora es considerar restricciones adicionales, como limitaciones en la asignación de capital o restricciones sectoriales. Además, se podría investigar la incorporación de otros factores, como la liquidez o la correlación entre activos, en la optimización del portafolio.

REFERENCES

[AM15] A.Adib Ahmadianfar and M.Taghian. “A Multiobjective Evolutionary Algorithm using Decomposition (MOEA/D) and its application in Multipurpose Multi-Reservoir operations”. In: *Ahmadianfar* 5.2 (2015), pp. 167–187. ISSN: 0305-0483.

[Gon15] González-Ramírez R. G y González-López J. J. “Aplicación de un algoritmo evolutivo multiobjetivo para el diseño óptimo de una red de transporte”. In: *Tecnología en Marcha* 28.5 (2015), pp. 135–146. DOI: 10.18845tm.v28i5.2194.

[Her15] Mostapha Kalami Heris. “MOEA/D in Matlab”. In: *Yarpiz* (2015).

[Mäk08] & Sulonen Mäkelä M. M. “Multi-objective optimization of an industrial process with an NSGA-II algorithm”. In: *Proceedings of the International MultiConference of Engineers and Computer Scientists* 2 (2008), pp. 1155–1159.

[Was] University of Washington. “Markowitz Mean-Variance portfolio theory”. In: ().

[Wik] Wikipedia. “Genetic Operators”. In: ().

[Wik21] Wikipedia. “Multi-objective optimization”. In: (2021).

[Y20] Deng Y. Han Y. “Solving Multi-Objective Portfolio Optimization Problem Based on MOEA/D”. In: *International Journal of Online and Biomedical Engineering* 16.5 (2020), pp. 43–55.