Algoritmo de optimización mediante forrajeo de bacterias híbrido para el problema de selección de portafolios con restricción de cardinalidad

Christian Leonardo Camacho-Villalón[1], Abel García-Nájera[2], Miguel Ángel Gutiérrez-Andrade[1]

[1] UAM Iztapalapa, Departamento de Ingeniería Eléctrica, Ciudad de México, México

[2] UAM Cuajimalpa, Departamento de Matemáticas Aplicadas y Sistemas, Ciudad de México, México

clcamachov@gmail.com, gamma@xanum.uam.mx, agarcian@correo.cua.uam.mx

Resumen. Este trabajo aborda el problema de la selección de portafolios de inversión óptimos (PSP). Mucha investigación se ha hecho en torno a esta tema, la mayor parte de los trabajos han buscado extender el modelo de Markowitz considerando restricciones realistas (piso-techo, clases y cardinalidad), y/o introduciendo otras medidas de riesgo (semivarianza, desviación absoluta, valor en riesgo, etc.). En este documento presentamos los resultados preliminares de un algoritmo de optimización multiobjetivo híbrido basado en optimización por forrajeo de bacterias (BFO), al cual integramos el enfoque de aprendizaje incremental basado en probabilidad (PBIL). El enfoque de PBIL hace uso de información estadística para guiar el proceso de mejora incremental de las bacterias. Para mejorar el desempeño de BFO, implementamos una función lineal decreciente para el tamaño de los pasos quimiotácticos, reinicialización de las bacterias y asignación de pesos aleatorios durante la fase de reproducción. Nuestra formulación incluye las restricciones de cardinalidad y piso-techo, dos restricciones realistas que son necesarias en la mayoría de los mercados bursátiles del mundo. Basados en el modelo de media-varianza propuesto por Markowitz, utilizamos la bien conocida formulación de Frontera Eficiente (EF) que integra en un solo objetivo el riesgo y el retorno a través de un parámetro de aversión al riesgo. Con la formulación anterior y utilizando un conjuntos de datos estándar para el PSP, llevamos a cabo la evaluación del desempeño del algoritmo. Los resultados obtenidos mostraron que nuestro algoritmo es capaz encontrar soluciones de buena calidad distribuidas uniformemente sobre la frontera eficiente.

Palabras clave: Optimización de portafolios, selección de portafolios, optimización por forrajeo de bacterias, BFO, inteligencia de enajambre, aprendizaje incremental.

Christian Leonardo Camacho-Villalón, Abel García-Nájera, Miguel Ángel Gutiérrez-Andrade

Hybrid Bacterial Foraging Optimization Algorithm for the Cardinality Constrained Portfolio Selection Problem

Abstract. In this paper we tackle the optimal portfolio selection problem (PSP). Many research has been made around this subject mainly in two ways, whether extending the Markowitz model by taking into account real-world constraints (floor-ceiling, class and cardinality) or introducing different risk measures like semivariance, value at risk, absolute desviation, etc. Here, we present the preliminary results of a new multiobjective heuristic based in the bacterial foraging optimization (BFO) which integrates the population based incremental learning (PBIL) approach. PBIL uses statistical information to guide the optimization process in the bacteria population. Furthermore, to improve the BFO heuristic we introduced a lineal decreasing function for the chemotaxis steps size, bacterias reinitialization and random weighing in the reproduction step. Our formulation include the cardinality and floor-ceiling constraints, both are real-world constraints needed in most of the stock markets. Based in the mean-variance model (first proposed by Markowitz) we used the well-known efficient frontier formulation which introduces a risk aversion parameter to weigh between risk and mean return, leading into a single-objective formulation problem. Applying our algorithm to solved the above mentioned model, we performed tests with a standard dataset taken from the OR-Lib. The experimental results shown our algorithm is able to find good-quality solutions uniformly distributed over the real efficient frontier.

Keywords: Portfolio optimization, portfolio selection, bacterial foraging optimization, BFO, swarm intelligence, population based incremental learning

1. Introducción

En 1952 Harry Markowitz hizo la mayor contribución sobre el problema de la selección de portafolios (PSP) con la publicación del modelo de media-varianza [1], también conocido como el modelo de Markowitz. Este modelo involucra dos objetivos en conflicto, por un lado se busca maximizar la ganancia (media) y por otro minimizar el riesgo (varianza), resultando en un problema de programación cuadrática (QP) de gran escala [2]. El modelo de media-varianza de Markowitz es ampliamente utilizado, sin embargo, hace una serie de simplificaciones y suposiciones irreales [3], entre las que están: 1) un mercado perfecto en donde no hay impuestos, 2) no considera costos de transacción, 3) la venta en corto no está permitida y 4) los activos se pueden dividir de manera infinita para su comercialización. Al extender el modelo original para incluir restricciones prácticas que son relevantes (esto es, hacerlo más realista), se vuelve más complicado de resolver. Si se incluye en la formulación del problema alguna restricción

que implique números enteros (como la restricción de cardinalidad o la de lotes mínimos), el problema se transforma de uno de programación cuadrática (QP) a uno de programación entera mixta cuadrática (QMIP), que está probado es de tipo NP-difícil [4]. De igual manera, si hay por lo menos una restricción de tipo cuadrático, es necesario recurrir a técnicas de optimización alternativas.

El PSP se puede formular como un problema de optimización multiobjetivo, en este tipo de problemas ya no se busca obtener una única solución, sino un conjunto de soluciones que representen el mejor compromiso entre todos los objetivos del problema. Las técnicas de optimización multiobjetivo tienen la habilidad de manejar de manera simultánea un conjunto de soluciones llamada población. Al conjunto de soluciones eficientes de la población se les llama óptimos de Pareto. Una característica esencial que se busca en los problemas multiobjetivo es lograr una distribución uniforme de las soluciones eficientes sobre el frente de Pareto.

Existen diversas técnicas matemáticas y métodos analíticos para resolver el problema de la selección de portafolios [5], sin embargo, la eficacia de estos métodos es limitada al no considerar restricciones realistas a la formulación del problema. El análisis utilizando en estas técnicas generalmente tiene que "adaptar" el problema para que pueda ser resuelto. Al considerar un número grande de activos en el problema, las técnicas analíticas se pueden ver rebasadas, además de volverse muy complicadas de emplear con un número grande de restricciones en el modelo o ser de tipo cuadrático. Por otro lado, las técnicas metaheurísticas pueden hacer frente a estos inconvenientes y encontrar la frontera eficiente con restricciones [6]. Dentro de las técnicas metaheurísticas están el algoritmo de recocido simulado (SA) y búsqueda tabú (TS). También se han empleado técnicas híbridas basadas en búsqueda local (LS) y el procedimiento de programación cuadrática (QP), los cuales han mostrado resultados comparables o superiores a los soluciones matemáticas y los métodos analíticos.

Muchos trabajos han utilizado algoritmos basado en poblaciones estocásticas, dentro de éstos, los algoritmos genéticos (GA) han mostrado mejores resultados que SA y TS [7]. Una técnica híbrida que utiliza un LS para encontrar el número óptimo de activos y después QP para determinar el peso de cada uno en el portafolio mostró buenos resultados [9]. La optimización multiobjetivo por colonia de hormigas (ACO) [8] se ha presentado como una metaheurística especialmente efectiva, los resultados obtenidos con esta técnica son comparables a los que se obtienen con la optimización de Pareto por recocido simulado y el algoritmo NSGA. El uso de un modelo híbrido de una red neuronal artificial con el algoritmo de optimización por enjambre de partículas (PSO) mostró la flexibilidad de las técnicas híbridas, así como su superioridad en predecir el desempeño del portafolio [6].

La optimización por forrajeo de bacterias (BFO) fue propuesta originalmente por Passino [10] en 2002 y es parte de las técnicas de inteligencia de enjambre (SI). Las bacterias en el algoritmo de BFO implementan un tipo de caminata aleatoria influenciada para encontrar las mejores soluciones. El algoritmo sigue la estrategia de forrajeo (alimentación) de bacterias reales en tres aspectos:

dirigirse hacía las regiones donde están las mejores soluciones y permanecer ahí más tiempo, evadir las regiones con las peores soluciones y salir de las regiones donde no se puedan mejorar las soluciones. Utilizando este comportamiento, el algoritmo propuesto tiene la habilidad de exploración y explotación del espacio de búsqueda para encontrar la frontera eficiente. Por otro lado, PBIL utiliza la idea evolutiva de una población de individuos basada en información estadística recolectada durante el proceso evolutivo.

El algoritmo propuesto integra a BFO la técnica de PBIL, así como algunas mejoras al algoritmo de BFO que ayudan a una adecuada exploración y explotación del espacio de búsqueda.

El resto del documento está estructurado de la siguiente manera. En la Sección 2 se describe el modelo de media-varianza y la formulación de Frontera Eficiente. En la Sección 3 se describen los algoritmos de BFO y PBIL. En la Sección 4 se introduce el algoritmo propuesto y las mejoras. En la Sección 5 se discuten los resultados obtenidos por el algoritmo. Finalmente, en la Sección 6 aparecen las conclusiones y el trabajo futuro.

2. El problema de selección de portafolios

2.1. Formulación de Frontera Eficiente

El modelo clásico de media-varianza de Markowitz [1] busca de manera simultánea la minimización del riesgo y la maximización del retorno esperado considerando como restricción que la suma de todos los activos debe ser igual a uno. Una de las formulaciones más utilizadas que emplean la formulación clásica de media-varianza es la siguiente:

$$\begin{aligned} \text{minimizar} \quad & \lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} x_i r_i\right] , \\ \text{sujeto a} \quad & \sum_{i=1}^{N} x_i = 1 , \\ & 0 \leq x_i \leq 1, \quad i = 1, \ldots, N . \end{aligned} \tag{1}$$

El modelo integra en un solo objetivo el riesgo y el retorno. Es posible encontrar diferentes valores para la función objetivo variando el retorno esperado deseado $R^{\star} = \sum_{i=1}^{N} x_i r_i$. La forma más común de hacerlo es introduciendo un factor de aversión al riesgo $\lambda \in [0, 1]$. Con este nuevo parámetro λ el modelo puede ser descrito a través de una sola función objetivo.

Cuando λ es cero, el modelo maximiza el retorno esperado del portafolio sin considerar la varianza (riesgo). En cambio, cuando λ es igual a uno, el modelo minimiza el riesgo del portafolio sin tomar en cuenta el retorno esperado. La sensibilidad del inversionista al riesgo se incrementa al incrementarse λ. Para diferentes valores de λ se obtienen diferentes valores de la función objetivo. Si se traza la intersección entre el valor del retorno esperado y la varianza para los

diferentes valores de λ se obtiene una curva continua llamada frontera eficiente, en donde cada punto de la frontera eficiente indica un valor óptimo.

Las dos restricciones realistas que más frecuentemente se han utilizado para el problema de optimización de portafolios de inversión son las siguientes:

i) *Piso-techo*: Imponen los límites inferiores y/o superiores (ϵ , δ) para el peso de los activos en lugar de utilizar cero como mínimo y uno como máximo. Por lo tanto, un activo no puede representar menos o más de cierta proporción del total del capital a invertir.

ii) *Cardinalidad*: Obligan a que los activos seleccionados en el portafolio respeten ciertas restricciones. Existen dos versiones de esta restricción. La primera versión (exacta) impone que el número de bonos seleccionados sea igual a un valor K . La segunda versión (suave) imponen los límites inferior y superior (Z_L, Z_U) para este valor.

La formulación del problema de la selección de portafolios con restricción de cardinalidad (CCPS) y piso-techo es la siguiente:

$$\begin{aligned} \text{minimizar} \quad & \lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} x_i r_i\right] , \\ \text{sujeto a} \quad & \sum_{i=1}^{N} x_i = 1 , \\ & \sum_{i=1}^{N} z_i = K , \\ & \epsilon z_i \leq x_i \leq \delta z_i, \quad i = 1, \ldots, N , \\ & z_i \in [0,1], \quad i = 1, \ldots, N . \end{aligned} \tag{2}$$

donde la variable z_i es de tipo binario y permite saber si el activo i está presente en la solución. El problema que resuelve nuestro algoritmo de optimización es el que se encuentra formulado en (2).

3. Algoritmos híbrido BFO-PBIL

3.1. BFO

El algoritmo original de optimización por forraje de bacterias (BFO) fue propuesto por Kevin M. Passino [10] en 2002 y es uno de los métodos más recientes dentro del área de inteligencia de enjambre (SI) para la optimización de problemas continuos. El algoritmo imita el comportamiento de forrajeo que llevan a cabo las bacterias de Escherichia coli (E. coli) presentes en el intestino humano. Las bacterias artificiales realizan tres actividades de forrajeo básicas: quimiotaxis, reproducción y eliminación-dispersión. En un movimiento quimiotáctico, el enjambre de bacterias trata de moverse y permanecer en los

entornos ricos en nutrientes, abandonar las regiones pobres en nutrientes rápidamente y permanecer alejadas de los lugares peligrosos.

Una bacteria lleva a cabo un movimiento quimiotáctico en dos pasos: *nado* y *desplome*. Las bacterias pueden hacer varios nados en una misma dirección si la concentración de nutrientes se incrementa a su alrededor. Una vez que la bacteria detecta que los nutrientes a su alrededor disminuyen, ejecuta la acción de desplome para cambiar rápidamente la dirección de la búsqueda. Los pasos de nado y desplome se ejecutan de manera alternada, a través del nado las bacterias permanecen por mayor tiempo en las regiones ricas en nutrientes y mediante el desplome son capaces de salir rápidamente de las regiones poco atractivas. La quimiotaxis puede ser vista como una estrategia bacteriana de optimización local, cuyo comportamiento móvil se describe mediante la siguiente fórmula:

$$\Theta^i(j+1,k,l) = \Theta^i(j,k,l) + C(i) \times \Phi(j), \tag{3}$$

donde $\Theta^i(j,k,l)$ denota la posición de la bacteria i en el paso quimiotáctico j, el paso reproductivo k y el paso de eliminación-dispersión l. $C(i)$ es el tamaño del paso quimitáctico de la bacteria i, el vector $\Phi(j)$ se utiliza para definir la dirección del movimiento aleatorio de un movimiento de desplome en el paso quimiotáctico j.

Para producir nuevas soluciones, las bacterias realizan una serie de movimientos quimiotácticos en los cuales se incrementa y decrementan el peso de los activos presentes en el portafolio (a través de nado y desplome). Cada nueva solución es evaluada por el algoritmo, si la solución nueva es mejor que la solución actual, esta última es reemplazada en el enjambre de bacterias. La ecuación quimiotáctica está definida como sigue:

$$nxx_b = xx_b^t + C(j) \times \Delta D_b(j), \forall j, \tag{4}$$

donde nxx_b son los nuevos valores obtenidos para la bacteria b, t es el número de la iteración del paso quimiotáctico, $C(j)$ es una constante que representa el tamaño del movimiento quimiotáctico (controla la distancia del movimiento), y $\Delta D_b(j)$ es un número aleatorio en el intervalo $[-1,1]$ que denota que la magnitud del cambio en la dirección en un paso de desplome. Tanto el nado como el desplome utilizan constantes para indicar el tamaño del paso, $C_d(j)$ indica el valor de $C(j)$ para el desplome y $C_n(j)$ el valor para el nado. Un paso quimiotáctico incluye un desplome y número de nados, el algoritmo incluye un mecanismo de autorevisión que implementa cada bacteria para controlar la ocurrencia de estos pasos.

Después de ejecutar una serie de movimientos quimiotácticos las bacterias intentarán reproducirse para mejorar las probabilidades de supervivencia. Cada una de las bacterias más fuertes se reproduce dividiéndose asexualmente en dos bacterias, las bacteria recién creada se ubicarán cerca del padre. Al mismo tiempo, las bacterias más débiles mueren dejando el número de bacterias en la población constante (este proceso es similar a la selección en los GA).

Finalmente, debido a cambios repentinos o graduales en el entorno local, el evento de eliminación puede suceder de tal manera que un subconjunto del

enjambre de bacterias sea eliminado o forzado a moverse a otro lugar. Si una bacteria es eliminada, una nueva será generada y colocada de manera aleatoria en el espacio de búsqueda (esta operación es similar a la mutación en los GA). El proceso de dispersión se encarga de cambiar de lugar las bacterias existentes a una mejor región. Aunque la probabilidad de que ocurran los eventos de eliminación-dispersión es baja, después de un periodo largo de tiempo, este proceso incrementa la diversidad de las soluciones y mejora la búsqueda local (evitando quedar atrapado en mínimos locales).

3.2. PBIL

PBIL está basado en la idea evolutiva de una población de individuos basada en información estadística recolectada durante el proceso evolutivo. Asumiendo que no hay dependencia entre las variables (esto es, la selección de los activos son eventos mutuamente excluyentes), PBIL utiliza un vector de probabilidad para representar la distribución de todos los individuos. El vector de probabilidad adquiere aprendizaje hacia el vector que representa la mejor solución y se utiliza para generar la siguiente generación de individuos.

3.3. Mejoras al algoritmo BFO

Existen estudios recientes que buscan mejorar algunas características del algoritmo de BFO, con respecto a nuestra técnica de optimización vale la pena mencionar los siguientes trabajos. En [11] los autores agregaron un mecanismo de comunicación que emplea la fórmula de actualización de movimiento de PSO (*Gbest*), de esta manera las bacterias son guiadas hacia la mejor solución en cada iteración. Esta mejora está basada en el hecho de que otras técnicas de optimización (como la evolución diferencial (DE) y PSO) que hacen uso de la comunicación para aprender de las demás partículas a su alrededor, han mostrado buenos resultados y un mejora significativa en el desempeño del algoritmo.

Otra propuesta importante aparece en [12], donde los autores utilizaron una función decreciente linealmente para definir el tamaño de los pasos quimiotácticos hasta un valor fijo. De esta manera, las bacterias hacen cambios grandes al inicio del proceso de optimización y progresivamente se vuelven más pequeños. Esta mejora también tiene su justificación en el algoritmo de PSO y los coeficientes de aceleración utilizados para actualizar la posición de una partícula. Al igual que la técnica propuesta en [12], los valores disminuyen gradualmente de forma lineal. La fórmula propuesta por los autores para calcular el tamaño de los pasos quimitácticos es básicamente la misma que la de PSO.

Las dos técnicas anteriores ofrecen al algoritmo BFO original una mejoría notable en los resultados reportados por los autores. Respecto a la primera ([11]), en nuestro algoritmo BFO-PBIL mejorado la técnica de PBIl permite hacer uso de la información de las mejores soluciones de una manera muy eficiente sin necesidad de introducir cálculos adicionales para cada una de las bacterias. Es decir, después de un proceso quimiotáctico se identifica a la mejor y peor solución en la población, con esta información se actualiza el vector de probabilidad y se

completa la población de bacterias para el siguiente ciclo de optimización por quimiotaxis.

Respecto a la segunda técnica ([12]), en el algoritmo original de BFO el tamaño de los movimiento quimiotácticos de nado y desplome es constante durante toda la ejecución del algoritmo, sin embargo, se ha visto que si el tamaño es demasiado grande las bacterias pueden fallar en encontrar al óptimo global realizando numerosos nados. Por otro lado, si el tamaño del movimiento es muy pequeño es posible que a las bacterias les tome mucho tiempo encontrar el óptimo global. En nuestro algoritmo BFO-PBIL mejorado utilizamos este enfoque e implementamos una cantidad decreciente para el tamaño de la constante de desplome ($C_d(j)$) según [12].

3.4. Algoritmo BFO-PBIL mejorado

El algoritmo de optimización híbrido propuesto BFO-PBIL mejorado está inspirado principalmente en tres trabajos importantes y recientes [13], [16] [14].

Definiciones para el algoritmo BFO-PBIL mejorado:

λ = Factor de aversión al riesgo,
$C_{d_{max}}$ = Valor máximo para el tamaño del desplome,
$C_{d_{min}}$ = Valor mínimo para el tamaño del desplome,
N_B = Número de bacterias en la población,
N = Número de activos disponible,
v = Vector de probabilidad (PBIL),
ED_{max} = Número de movimientos de eliminación dispersión,
R_{max} = Número de movimientos reproductivos,
Q_{max} = Número de movimientos quimiotácticos,
G_{best} = La bacteria con mejor valor de aptitud,
G_{wort} = La bacteria con el peor valor de aptitud,
$Prob_{ED}$ = Probabilidad de eliminación-dispersión de un activo,
$Capital$ = Capital disponible para invertir,
ϵ = Límite inferior (restricción de piso-techo),
δ = Límite superior (restricción piso-techo),
K = Número de activos en el portafolio (restricción cardinalidad),
LR = Porcentaje de aprendizaje positivo,
NEG_LR = Porcentaje de aprendiza negativo.

Utilizamos el enfoque propuesto inicialmente por [7] dividiendo λ en 50 partes iguales. El valor del factor de aversión al riesgo λ en la iteración del algoritmo j se calcula con:

$$\lambda_j = (j-1)/49 \quad j = 1, ..., 50. \tag{5}$$

Tamaño de desplome decreciente: La función decreciente linealmente para el tamaño de la constante de desplome se determina con base en un valor inicial máximo ($C_{d_{max}}$) y un valor final mínimo ($C_{d_{min}}$), si Q_{max} es el número máximo

Algoritmo 1 BFO-PBIL mejorado

```
Inicializa vector de probabilidad incremental en 0.5
Inicializa población aleatoria inicial
para N_ed ← 1 to ED_max hacer
  para N_rep ← 1 to R_max hacer
    para N_quim ← 1 to Q_max hacer
      para b ← N_B/2) to N_B hacer
        Realiza movimientos de desplome y nado {4}
      fin para
    fin para
    Elimina la mitad de la población según f(b);
    Actualiza el vector v con (G_best) y (G_wort) {Ecs:78}
    Genera nueva bacteria b {Según el algoritmo:2}
  fin para
  para b ← 1 to N_B hacer
    si rand[0,1] ≤ Prob_ED entonces
      Elimina el Activo de la bacteria
      Selecciona un activo no incluido previamente
    fin si
  fin para
fin para
```

de pasos quimiotácticos y Q_{act} el número de la iteración actual, para el paso quimiotáctico j el tamaño de la constante de desplome $C_d(j)$ está dado por:

$$C_d(j) = C_{d_{min}} + \frac{Q_{max} - Q_{act}}{Q_{max}} \times \left(C_{d_{max}} - C_{d_{min}}\right). \tag{6}$$

Vector PBIL: El algoritmo que proponemos utiliza el enfoque de [16] para actualizar el vector de probabilidad (v). La actualización se realiza de acuerdo a un porcentaje de aprendizaje que puede ser positivo (LR) o negativo (NEG_LR). El porcentaje utilizado no solo controla la velocidad a la que el vector cambia para parecerse a la mejor solución, sino también la cantidad del espacio de búsqueda que será explorado. El uso de aprendizaje positivo y negativo tiene como objetivo aumentar la probabilidad de incluir los activo que contribuye a generar una buena solución y alejarse de los que no lo hacen.

$$v_i = v_i \times (1 - LR) + s_i^{G_{best}} \times LR, \tag{7}$$

donde $s_i^{G_{best}}$ es una variable binaria que permite saber si el activo i está presente en la mejor solución G_{best}. Si además sucede que el activo i está presente en $s_i^{G_{best}}$ y no lo está en la peor solución ($s_i^{G_{worts}}$) entonces:

$$v_i = v_i \times (1 - NEG\_LR) + s_i^{G_{best}} \times NEG\_LR. \tag{8}$$

En [16] los autores utilizaron el enfoque de mutación parcialmente guiada (PGM), en el cual en cada iteración del proceso evolutivo, cada dimensión del

vector de probabilidad se muta con una cierta probabilidad MP. Si el activo i es seleccionado se da igual oportunidad de mutarlo según un porcentaje de mutación (MR) o con el valor de la mejor solución $s_i^{G_{best}}$. En nuestro algoritmo decidimos utilizar el vector PBIL únicamente para guiar la selección de los activos que van a integrar las nuevas soluciones durante el proceso reproductivo como aparece en el Algoritmo 2.

Algoritmo 2 Reproducción con vector de probabilidad

```
1: para i ← 1 to N hacer
2:    si  rand[0, 1] < 0.5 y v_i > 0.5 entonces
3:       b_i = rand[0, 1] ∗ Capital
4:    sino
5:       si  bp_i > 0 entonces
6:          b_i = rand[0, 1] ∗ Capital
7:       fin si
8:    fin si
9:    Repara la bacteria b {Sección:3.5}
10: fin para
```

Reproducción con pesos aleatorios: Después de un proceso quimiotáctico viene un proceso de reproducción. En el algoritmo original de BFO cada bacteria se dividen asexualmente haciendo una copia idéntica de si misma, nosotros utilizamos un esquema de reproducción con el vector de probabilidad (v) para generar la mitad de la población faltante. El mecanismo de reproducción da igual oportunidad de seleccionar un activo presente en la bacteria padre o en el vector (v), el activo debe tener un peso mayor en v a 0.5 ó un peso mayor a 0 en la bacteria padre, si no se cumple alguno de estos criterios el activo se selecciona aleatoriamente cuando la bacteria es reparada. El peso asignado a los activos en las nuevas bacterias se distribuye aleatoriamente. Las nuevas bacterias estarán integradas por los activos de mayor calidad quedando ubicadas en regiones prometedoras del espacio de búsqueda.

Reinicialización aleatoria: Otra mejora incluida en nuestro algoritmo es la reinicialización de las bacterias después de un proceso quimiotáctico. Cada bacteria se evalúa para saber si logró modificar su valor de aptitud de manera significativa (con una diferencia de 10^{-5}). La idea es identificar a las bacterias que pueden estar atrapadas en un óptimo local. Con el objetivo de obtener una adecuada relación entre la *exploración* y *explotación*, si al llegar al número máximo de movimientos quimiotácticos durante un proceso de quimotaxis la bacteria no cambió de posición se reinicializa a una posición nueva aleatoria.

Algoritmo 3 Restriccion de cardinalidad

para $b \leftarrow 1$ **to** N_B **hacer**
  Ordena b según $f(b)$
  si $|b| > K$ **entonces**
    repetir
      Elimina el activo de menor peso
    hasta $|b| = K$
  fin si
  si $|b| < K$ **entonces**
    repetir
      Agrega un activo aleatoriamente
    hasta $|b| = K$
  fin si
fin para

3.5. Manejo de restricciones

Para cumplir con las restricciones de presupuesto, cardinalidad y piso-techo implementamos una proceso de reparación que evalúa y corrige cada bacteria. Primero se revisa que la cardinalidad de la solución sea igual a K según se expresa en el Algoritmo (3).

Posteriormente, una función disminuye hasta δ el peso de los activo que exceden el límite superior y aumenta hasta ϵ los que se encuentran por debajo de este valor.

$$x_i = \begin{cases} \delta, & \text{si } x_i > \delta \\ \epsilon, & \text{si } x_i < \epsilon. \end{cases} \tag{9}$$

Finalmente, una función de normalización de pesos es utilizada para cumplir con la restricción de capital. Esta función hace uso de un acumulador de capital excedente o sobrante en caso de que no sea posible decrementar o incrementar el peso de un activo sin violar la restricción de piso-techo. Después de la normalización se asigna el capital sobrante o faltante a los activos que pueden absorberlo. En la ecuación (10) el parámetro *Capital* representa el capital disponible por el inversionista, nx_i es el nuevo peso asignado al activo.

$$nx_i = \begin{cases} Capital \times \left(\frac{x_i}{\sum_i^N x_i} \right), & \text{si } \epsilon \leq nx_i \leq \delta \\ x_i, & \text{si } nx_i < \epsilon \text{ ó } nx_i > \delta. \end{cases} \tag{10}$$

4. Experimentación y resultados

4.1. Conjunto de datos

Para probar el desempeño del algoritmo utilizamos un conjunto de datos estándar propuesto inicialmente en [7]. Este conjunto de datos ha sido ampliamente utilizado y es reconocido como un marco de comparación para la

evaluación de algoritmos de optimización. Los archivos están disponibles en [15] y cada uno está conformado por el número de activos, el retorno estimado y la varianza de cada activo, y el coeficiente de correlación para cada pareja de activos i,j. Los activos incluidos en los archivos corresponde a los precios de cierre de cinco índices bursátiles: Hang Seng en Hong Kong (31 activos), DAX 100 en Alemania (85 activos), FTSE 100 en Reino Unido (89 acciones), S&P 100 en EE.UU. (98 activos) y Nikkei 225 en Japón (225 activos). Finalmente, para cada archivo de datos los autores proveen los puntos que conforman la frontera eficiente real.

4.2. Configuración del algoritmo

Los parámetros de configuración del algoritmo se establecieron en $Max_{\lambda} = 50$, los valores máximos y mínimos para los pasos quimiotácticos $C_{d_{max}} = 0.01$ y $C_{d_{min}} = 0.005$, el tamaño de población $N_B = 30$, el número de pasos de eliminación dispersión $MAX_{Elim-Disp} = 2$, reproductivos $MAX_{Reprod} = 20$, y una probabilidad de eliminación dispersión $Prob_{ED} = 0.25$. El número de pasos quimitácticos se fijo en $MAX_{Quim} = 30$, con un $Max_{nados} = 2$ después de un desplome. El *Capital* se fijó en $500,000$ con un límite inferior $\epsilon = 0.01$ y superior $\delta = 1$ para la restricción de piso-techo, para la de cardinalidad el valor de $K = 10$ según el enfoque de [7]. La velocidad de aprendizaje positivo fue $LR = 0.1$ y del negativo $NEG\_LR = 0.075$.

4.3. Resultados

Utilizamos el método de evaluación propuesto por [7] que mide la porcentaje de desviación horizontal y verticalmente de cada punto encontrado no dominado con la frontera eficiente real. Los resultados incluyen las siguientes medidas de desempeño: la media del porcentaje de desviación (MPD), la mediana del porcentaje de desviación (MedPD), el número de puntos no dominados y el tiempo total expresado en segundo. Se utilizó la misma configuración para cada conjunto de datos con los que se probó el algoritmo (Sección 4.2). Los resultados mostrados en la Tabla 1 son el promedio de veinte ejecuciones del algoritmo para los conjuntos de datos de 31, 85 y 89 activos resolviendo el PSP con restricciones de cardinalidad y piso-techo. En la Tabla 2 aparecen los mejores resultados obtenidos para el PSP sin restricciones.

En la Tabla 1 se presenta la comparación de BFO-PBIL contra PBILDE [16] que utiliza la técnica de evolución diferencial (DE) y tres heurísticas propuestas en [7] que incluyen un algoritmo genético (GA), búsqueda tabú (TS) y recocido simulado (SA). En la Figura 1 se muestra la frontera eficiente encontrada por nuestro algoritmo BFO-PBIL mejorado y la frontera eficiente real resuelta mediante programación cuadrática (QP). Los resultados que hemos obtenido hasta el momento para el PSP con restricciones son pobres comparados con las otras soluciones, creemos que esto es debido a una configuración deficiente en los parámetros del algoritmo. La razón por la que consideramos estos último, es que en las pruebas realizadas para el PSP sin restricciones el algoritmo mostró un

Tabla 1: Comparativa del desempeño para el PSP con restricciones

N	Medida	BFO-PBIL	PBIL-DE	Chang-GA	Chang-TS	Chang-SA
31	Puntos	276	**6367**	1317	1268	1003
	MPD(%)	4.3012789938	**0.6196**	0.9457	0.9908	0.9892
	MedPD(%)	4.4158656188	**0.4712**	1.1819	1.1992	1.2082
	Tiempo	759	113	172	**74**	79
85	Puntos	151	**3378**	1270	1467	1135
	MPD(%)	14.3790364757	**1.5433**	1.9515	2.5383	2.4675
	MedPD(%)	9.9511868431	**1.0986**	2.1262	3.0635	2.4299
	Tiempo	1406	1358	544	**199**	210
89	Puntos	203	**2957**	1482	1301	1183
	MPD(%)	7.9960532075	0.8234	0.8784	1.3908	**0.7137**
	MedPD(%)	7.2076477735	**0.5134**	0.5938	0.6361	1.1341
	Tiempo	1533	1496	573	246	**215**

Tabla 2: Comparativa del desempeño para el PSP sin restricciones

N	Medida	BFO-PBIL	PBIL-DE	Chang-GA	Chang-TS	Chang-SA
31	MPD(%)	0.510777	**0.0002**	0.0202	0.8973	0.1129
	MedPD(%)	0.000004	**0.000002**	1.1819	1.1992	1.2082
	Tiempo	223	**109**	621	469	476
85	MPD(%)	0.74099	**0.0052**	0.0136	3.5645	0.0394
	MedPD(%)	**0.00001**	0.0000211	0.0123	2.7816	0.0033
	Tiempo	**905**	1445	10332	9546	9412

comportamiento similar con las configuraciones que dieron un peor desempeño en las medidas de cantidad de puntos y el MPD. Para el problema sin restricciones, al probar diferentes configuraciones logramos identificar los mejores valores para los parámetros de configuración, sin embargo, hasta el momento aún no hemos realizado estas misma pruebas para el PSP con restricciones.

Para el problema formulado sin restricciones nuestro algoritmo produce soluciones de buena calidad que son competitivas con las heurísticas contra las que comparó el desempeño del algoritmo. Los resultado obtenidos para el PSP sin restricciones se presentan en la Tabla 2.

Establecimos la comparación de nuestro algoritmo BFO-PBIL mejorado contra PBILDE [16] y tres heurísticas propuestas en [7]. En [13] los autores emplearon medidas de desempeño diferentes por lo que no fue posible establecer una comparación con esta heurística. Con el objetivo de mostrar las mejoras que ofrece nuestra solución comparada con el algoritmo de BFO [13], se presenta en la Figura 2 las fronteras eficientes encontradas por las dos técnicas para el PSP sin restricciones. Como es posible observar, las mejoras introducidas al algoritmo permiten encontrar buenas soluciones ubicadas más cerca a la frontera eficiente real para los portafolios que ofrecen menor riesgo y menor retorno. Además, los

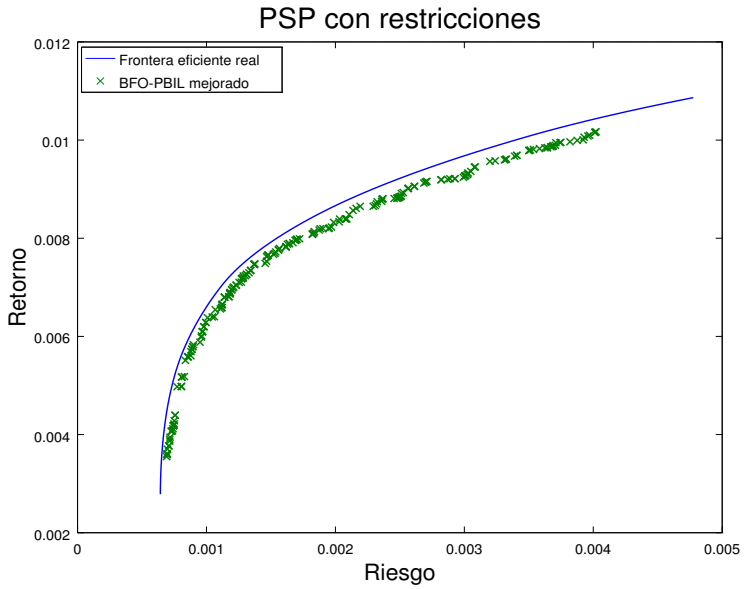

Fig. 1: BFO-PBIL mejorado para el problema (2). Conjunto de datos: 31 activos

portafolios encontrados por BFO-PBIL mejorado se aprecian mejor distribuidos sobre la frontera eficiente. Por otro lado, los dos algoritmos pudieron encontrar los mejores portafolios ubicados en el área de mayor riesgo y mayor retorno, para los cuales se observa una buena distribución sobre esta parte de la frontera eficiente.

5. Trabajo futuro

Los resultados aquí presentados son parte de un trabajo más amplio que aún se encuentra en curso. En dicho trabajo estamos analizando el algoritmo aquí propuesto con diferentes formulaciones del PSP y diferentes restricciones realistas que pocas veces son consideradas. En lo que respecta al algoritmo BFO-PBIL, es necesario probar los parámetros de configuración con distintos valores para el tamaño de la población N_B y el número de iteraciones de los pasos de eliminación-dispersión ($MAX_{Elim-Disp}$) y reproductivos (MAX_{Reprod}). Hemos visto que al utilizar el enfoque de PBIL es necesario aumentar el número de pasos reproductivos para dar tiempo al vector de obtener un aprendizaje significativo e incluirlo en las nuevas bacterias para llegar a buenos resultados. En este trabajo mostramos el potencial que tiene el algoritmo híbrido BFO-PBIL mejorado con una configuración estándar, sin embargo, es necesario realizar pruebas con conjuntos de datos más grandes, nuestro objetivo es proponer un algoritmo que sea robusto bajo un número grande de instancias como es el caso de los mercados bursátiles.

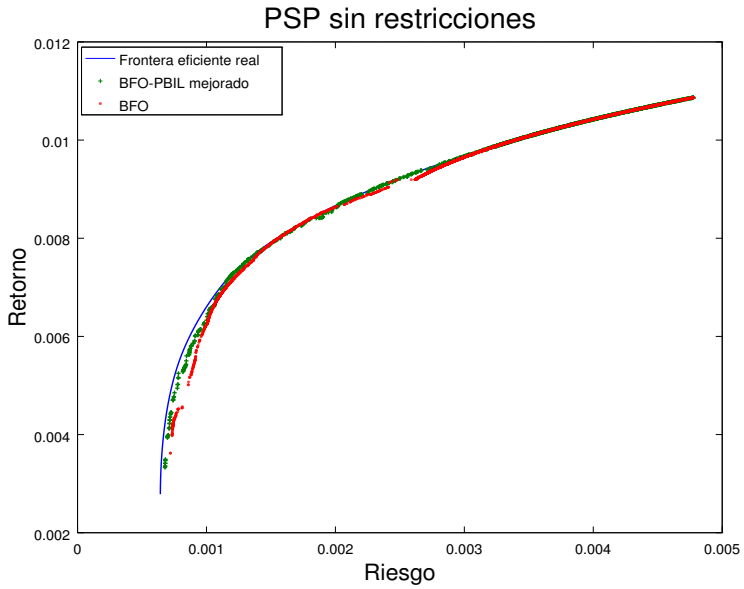

Fig. 2: Comparativa de BFO y BFO-PBIL mejorado para el problema (1). Conjunto de datos: 31 activos

6. Conclusiones

El algoritmo BFO es una de las heurísticas más novedosas en el área de Inteligencia de Enjambre. La técnica ha demostrado un gran potencial para resolver problemas de optimización en diferentes áreas y recientemente se ha empezado a utilizar para resolver el problema de la optimización de portafolios. En este trabajo hemos modificado el algoritmo original de BFO propuesto por Passino para mejorar algunas de las limitaciones que presentaba. Al incluir mejoras como una función lineal decreciente para el tamaño de los pasos quimiotácticos, reinicialización de las bacterias y asignación de pesos aleatorios durante el fase de reproducción hemos visto una mejora significativa en el desempeño del algoritmo. Además, hemos integrado y adaptado la técnica de aprendizaje incremental PBIL a BFO de manera exitosa agregando un componente que guía a las bacterias hacia buenas regiones con una adecuada exploración y explotación del espacio de búsqueda. Los resultados preliminares que hemos obtenidos hasta el momento mostraron que nuestro algoritmo es capaz encontrar soluciones de muy buena calidad que son competitivas con otros algoritmos de optimización.

Agradecimientos. El primer autor agradece el apoyo recibido por el CONACyT a través de una beca para estudios de posgrado.

Referencias

1. Markowitz, H.: Portfolio selection. The journal of finance, 1(7), 77–91 (1952)
2. Gupta, P., Mehlawat, M. K., Saxena, A.: Asset portfolio optimization using fuzzy mathematical programming. Information Sciences, 178(6), 1734–1755 (2008)
3. A. Ponsich, A.L. Jaimes, C.A.C. Coello: A Survey on Multiobjective Evolutionary Algorithms for the Solution of the Portfolio Optimization Problem and Other Finance and Economics Applications. IEEE Transactions on Evolutionary Computation, vol. 17, no.3, 321–344 (2013)
4. R. Ruiz-Torrubiano, A. Suarez, R. Moral-Escudero: Selection of optimal investment portfolios with cardinality constraints. In: Evolutionary computation. IEEE congress on CEC, pp. 2382–2388 (2006)
5. Vitoantonio Bevilacqua, Vincenzo Pacelli, Stefano Saladino: A novel multi objective genetic algorithm for the portfolio optimization. In: Advanced Intelligent Computing, Springer, pp. 186–193 (2012)
6. Hanhong Zhu, Yi Wang, Kesheng Wang, Yun Chen: Particle swarm optimization (pso) for the constrained portfolio optimization problem. Expert Systems with Applications, 38(8):10161–10169 (2011)
7. Chang, T.-J., Meade, N., Beasley, J.E., Sharaiha, Y.M.: Heuristics for cardinality constrained portfolio optimisation. Comp. & Opns. Res. 27, 1271–1302 (2000)
8. Doerner, K., Gutjahr, W., Hartl, R., Strauss, C., Stummer, C.: Pareto antcolony optimization: A metaheuristic approach to multiobjective portfolio selection. Annals of Operations Research, 131, 79–99 (2004)
9. Gaspero, L.D., Tollo, G., Roli, A., Schaerf, A.: Hybrid metaheuristics for portfolio selection problems. In: MIC 2007–Metaheuristics International Conference, Montreal (2007)
10. Passino, K. M.: Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Systems Magazine, 22, 52–67 (2002)
11. Tan, L., Niu, B., Wang, H., Huang, H., Duan, Q.: Bacterial foraging optimization with neighborhood learning for dynamic portfolio selection. Intelligent Computing in Bioinformatics, Springer International Publishing, pp. 413–423 (2014)
12. Niu, B., Xiao, H., Tan, L., Li, L., Rao, J.: Modified Bacterial Foraging Optimizer for Liquidity Risk Portfolio Optimization. Life System Modeling and Intelligent Computing, Springer Berlin Heidelberg, pp. 16–22 (2010)
13. Y. Kao, H.T. Cheng: Bacterial Foraging Optimization Approach to Portfolio Optimization. Computational Economics, vol. 42, num. 4, pp. 453–470 (2013)
14. S. G. Reid, K. M. Malan, A. P. Engelbrecht: Carry trade portfolio optimization using particle swarm optimization. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 3051–3058 (2014)
15. Beasley, J. E.: Or library dataset. (1999) http://people.brunel.ac.uk/ mastjjb/jeb/orlib/portinfo.html
16. K. Lwin, R. Qu: A hybrid algorithm for constrained portfolio selection problems. Applied intelligence, vol. 39, num. 2, pp. 251–266 (2013)