

Multi Objective Mixed Integer Nonlinear Optimization Based on PSO with Application to Integrated Weed Management

Lucía Damiani¹, Franco A. Molinari², Mariano Frutos³, Guillermo R. Chantre², Aníbal M. Blanco^{1*}

¹Planta Piloto de Ingeniería Química, Universidad Nacional del Sur (PLAPIQUI-CONICET),

²Departamento de Agronomía, Universidad Nacional del Sur y CERZOS UNS-CONICET

³Departamento de Ingeniería, Universidad Nacional del Sur e IIESS UNS-CONICET
8000 Bahía Blanca, Argentina.

*ablanco@plapiqui.edu.ar

Abstract. An optimization tool for solving nonlinear multi-objective mixed-integer problems is presented. The algorithm is based on the particle swarm metaheuristic (PSO). As PSO was designed to be applied to box-constrained continuous problems, a technique based on the total of the violations of the restrictions of each particle was incorporated to address restricted problems of the general type. Additionally, to treat binary variables, the "Angle Modulation" method was adopted, which through the inclusion of four additional continuous variables and the evaluation of a trigonometric function, generates the values of all the binaries along the search. Finally, to address multi-objective problems, a methodology was incorporated to identify the Pareto front. The developed algorithm was tested on different benchmark functions of two and three objectives, obtaining satisfactory results. The potential of the developed tool is illustrated through a case study of agronomic interest: the design of strategies for integrated weed management.

Keywords: optimization, PSO, MINLP-MO, Integrated Weed Management

Optimización mixto entera no lineal multi-objetivo basada en PSO con aplicación al manejo integrado de malezas

Lucía Damiani¹, Franco A. Molinari², Mariano Frutos³, Guillermo R. Chantre², Aníbal M. Blanco^{1*}

¹Planta Piloto de Ingeniería Química, Universidad Nacional del Sur (PLAPIQUI-CONICET),

²Departamento de Agronomía, Universidad Nacional del Sur y CERZOS UNS-CONICET

³Departamento de Ingeniería, Universidad Nacional del Sur e IIESS UNS-CONICET
8000 Bahía Blanca, Argentina.

*ablanc@plapiqui.edu.ar

Resumen. Se presenta una herramienta de optimización para la resolución de problemas mixto entero no lineales multi-objetivo. El algoritmo se basa en la metaheurística de enjambre de partículas (PSO). Como PSO está diseñado para aplicarse a problemas continuos sin restricciones, para poder abordar problemas restringidos se le incorporó una técnica basada en el total de las violaciones a las restricciones de cada partícula. Adicionalmente, para tratar variables binarias, se adoptó el método “Angle Modulation”, el cual agrega cuatro variables continuas adicionales que, a través de una función trigonométrica, van proporcionando los valores de todas las variables binarias a lo largo de la exploración. Finalmente, para abordar problemas multi-objetivo, se incorporó una metodología para identificar el frente de Pareto. El algoritmo desarrollado se probó sobre diferentes funciones benchmark de dos y tres objetivos, obteniéndose resultados satisfactorios. Las prestaciones de la herramienta desarrollada se ilustran mediante un caso de estudio de interés agronómico: el diseño de estrategias para el manejo integrado de malezas.

Palabras Claves: optimización, PSO, MINLP-MO, Manejo Integrado de Malezas

1 Introducción

En las ciencias e ingenierías se pueden encontrar una gran variedad de problemas que pueden representarse matemáticamente a través de modelos de optimización mixto entera no lineal multi-objetivo (MINLP-MO). En este tipo de problemas, Ec. (1), se intenta hallar la mejor combinación de variables continuas (\mathbf{x}) y binarias (\mathbf{y}) que maximice una dada cantidad (NFO) de criterios de desempeño o funciones objetivo ($\mathbf{f}(\cdot)$) de acuerdo a ciertas restricciones de igualdad ($\mathbf{h}(\cdot)$) y desigualdad ($\mathbf{g}(\cdot)$). Asimismo, en el caso de las variables continuas, también se debe considerar sus límites superiores (\mathbf{x}^{up}) e inferiores (\mathbf{x}^{lo}).

$$\text{Minimizar } \mathbf{f}(\mathbf{x}, \mathbf{y}) = (f_1(\mathbf{x}, \mathbf{y}), \dots, f_k(\mathbf{x}, \mathbf{y}), \dots, f_{NFO}(\mathbf{x}, \mathbf{y}))$$

Sujeto a:

$$\mathbf{h}(\mathbf{x}, \mathbf{y}) = 0; \mathbf{g}(\mathbf{x}, \mathbf{y}) \leq 0; \mathbf{x}^{lo} - \mathbf{x} \leq 0; \mathbf{x} - \mathbf{x}^{up} \leq 0; \mathbf{y} \in (0,1) \quad (1)$$

Los problemas que presentan no linealidades en sus funciones objetivo o en sus restricciones pueden introducir no convexidades que ocasionan regiones factibles irregulares. Esto conduce a múltiples óptimos locales, dificultando encontrar la mejor solución o, incluso, simplemente una factible. Asimismo, la presencia de variables binarias conlleva una “complejidad combinatoria” debido a la gran cantidad de posibilidades que se pueden enumerar, incluso para un número modesto de decisiones. Por todas estas razones, este tipo de problemas es mucho más complejo de resolver que los que solo comprenden la versión continua y lineal. Otra complicación adicional es la necesidad de optimizar dos o más funciones objetivo en simultáneo, las cuales, en general, se encuentran en conflicto entre sí. Esto ocasiona que una única solución no optimice a la vez todos los objetivos considerados y que la mejora en uno de ellos conlleva el detrimento de los restantes. Esta característica acrecienta aún más los esfuerzos computacionales requeridos para encontrar soluciones de compromiso que resuelvan esta clase de problemas.

Por todas estas razones, la resolución de problemas de optimización MINLP-MO en tiempos de cómputo compatibles con aplicaciones prácticas es uno de los mayores desafíos dentro de la matemática aplicada moderna. Existen numerosas herramientas, libres y comerciales, para programar y resolver modelos MINLP. Las herramientas comerciales GAMS, gProms y Matlab, por ejemplo, proporcionan entornos de programación amigables y solvers muy eficientes, pero poseen costos elevados en concepto de licencias de software. Existen plataformas no-comerciales con buenas prestaciones para ciertos tipos de modelos (LP, NLP, MILP), pero no abundan los algoritmos MINLP generales en un entorno de programación lo suficientemente flexible para adaptarlo a aplicaciones diversas. Respecto de la optimización MO, habitualmente se requiere una programación ad-hoc en la plataforma de desarrollo adoptada. Por todo lo anterior, el objetivo principal de este trabajo es presentar una implementación propia de un solver MINLP-MO para abordar diferentes problemas de investigación y transferencia a desarrollar en el ámbito de nuestro grupo de investigación, e ilustrar su desempeño mediante problemas benchmark y una aplicación de interés agronómico.

2 Herramienta de optimización MINLP-MO

Existe una gran cantidad de algoritmos de optimización numérica para resolver problemas MINLP-MO. Una clasificación básica se establece entre los determinísticos y los metaheurísticos. La optimización determinística es completamente predictiva, es decir, si se especifican los mismos puntos iniciales de las variables de entrada del algoritmo, éste convergerá siempre a la misma solución. Su implementación es matemáticamente sofisticada, depende de derivadas de las funciones no lineales y de exploraciones de

tipo “branch and cut” para los árboles de decisiones, y bajo ciertas condiciones, pueden garantizar la convergencia al óptimo (local e incluso global).

Por otro lado, se encuentran las metaheurísticas, las cuales cuentan con elementos aleatorios para guiar la exploración del espacio de soluciones, por lo cual pueden no necesariamente converger exactamente a la misma solución mediante la misma secuencia de cálculo. A diferencia de los métodos determinísticos, cuentan con la gran ventaja de no necesitar puntos iniciales próximos a los óptimos ni verificar propiedades matemáticas especiales, tornándolas muy populares para resolver problemas mixto-entero no lineales de tipo general [1].

En particular, en este trabajo se adoptó la metaheurística basada en el enjambre de partículas (PSO, por sus siglas en inglés) debido principalmente a: (i) un buen rendimiento reportado para una gran variedad de problemas; (ii) su relativa sencillez de implementación; (iii) su flexibilidad para anexarle nuevas técnicas que amplíen su campo de aplicación.

3.1 Optimización continua por Enjambre de Partículas (PSO)

La optimización continua por enjambre de partículas es un método propuesto originalmente en [2], basándose en la conducta de comunidades de organismos vivos, como por ejemplo los enjambres de abejas en la búsqueda de alimento.

El PSO parte de una población o enjambre de tamaño N de soluciones candidatas denominadas “partículas” que se van moviendo por el espacio de D dimensiones o variables (j), en la búsqueda del óptimo. La exploración se realiza principalmente en base a dos ecuaciones matemáticas sencillas en las cuales se va actualizando, en cada iteración (k), la posición (\mathbf{x}) (2) y la velocidad (\mathbf{v}) (3) de cada partícula (i) del enjambre. A medida que la población se mueve, los individuos van registrando la mejor posición encontrada por cada uno de ellos, a la que se denomina “mejor local” (\mathbf{p}), así como la mejor lograda por todos ellos, llamada “mejor global” (\mathbf{q}). Al hablar de la “mejor solución” se hace referencia a aquella que logra el menor valor de la función objetivo (\mathbf{f}) si se trata de un problema de minimización del objetivo.

$$x_{ij}^{k+1} = x_{ij}^k + v_{ij}^{k+1} t^k \quad \forall i \in [1, N], \forall j \in [1, D] \quad (2)$$

$$v_{ij}^{k+1} = w^k v_{ij}^k + c_1 r_{1ij} (p_{ij}^k - x_{ij}^k) + c_2 r_{2ij} (q_j - x_{ij}^k) \quad \forall i \in [1, N], \forall j \in [1, D] \quad (3)$$

Finalmente, cuando se cumple el criterio de terminación impuesto (típicamente un número máximo de iteraciones, k_{max}), el algoritmo devuelve la mejor solución (\mathbf{q}) encontrada hasta ese momento.

3.2 Tratamiento de restricciones

El PSO básico [2] sólo tiene en cuenta los límites superiores (x_j^{up}) e inferiores (x_j^{lo}) de cada dimensión (j) del problema, pero no cuenta con un tratamiento explícito para manipular restricciones de igualdad ($\mathbf{h}(\cdot)$) y de desigualdad ($\mathbf{g}(\cdot)$). Para poder manipular

estos tipos de restricciones se incorporó al PSO descrito por (1) y (2) la técnica propuesta en [3] basada en el concepto de violación absoluta de las restricciones. La misma se adoptó por ser bastante general, no requerir la reformulación o reducción de restricciones y requerir pocos parámetros adicionales, lo que la vuelve, a nuestro entender, más práctica que otros métodos alternativos.

La metodología elegida consiste en agrandar temporalmente la región factible original, obteniendo una zona más relajada (Fig. 1). Esta región relajada se va reduciendo progresivamente a medida que las partículas comienzan a ingresar a ella hasta convertirse nuevamente en la región factible del problema original. Para ello, en cada iteración (k) se debe calcular el valor del total de violaciones a las restricciones (TAV por sus siglas en inglés) por parte de cada partícula i (4), y el valor de relajación μ (5), el cual establece si una solución es temporalmente factible (cuando $TAV_i^k < \mu^k$). Cabe mencionar que, en este trabajo, los límites de cada variable (límites de caja) se consideraron como restricciones de desigualdad y se tratan según el mismo procedimiento. En (5) F_F representa el número de individuos factibles en una dada iteración.

$$TAV_i = \sum_{s=1}^{m1} |h_s(x)| + \sum_{s=1}^{m2} \max(0, g_s(x)) + \sum_{j=1}^D \max(0, (x_j^{lo} - x_{ij}^k)) + \sum_{j=1}^D \max(0, (x_{ij}^k - x_j^{up})) \quad \forall i \in [1, N], \forall j \in [1, D] \quad (4)$$

$$\mu^{k+1} = \mu^k \left(1 - \frac{F_F}{N}\right) \quad (5)$$

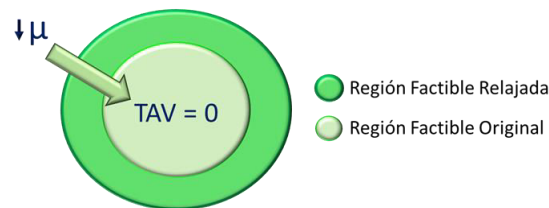


Fig. 1. Relajación de la región factible para el tratamiento de restricciones

Dado que ahora se tienen en cuenta las restricciones, cambia la forma de determinar cuáles son los mejores global y locales en cada iteración. Según el criterio de factibilidad, se considera que:

- (i) Una solución factible ($TAV_i^k < \mu^k$) es preferible a una infactible ($TAV_i^k > \mu^k$);
- (ii) Entre dos soluciones factibles ($TAV_i^k < \mu^k$), aquella que alcanzó un mejor valor de la función objetivo (f_i) es la preferida;
- (iii) Entre dos soluciones infactibles ($TAV_i^k > \mu^k$), se selecciona la de menor TAV_i .

Esta técnica favorece la exploración del espacio de búsqueda al admitir violaciones transitorias a las restricciones mientras conduce a la población hacia la región factible de manera progresiva. La implementación de esta prestación se testeó sobre diversas funciones benchmark que difieren en su complejidad de resolución debido a la presencia de diversos tipos de no linealidad y restricciones, así como de distintos números de

ecuaciones y variables en [4]. Los resultados resultantes de optimizar este set de problemas demuestran un buen rendimiento general en todos ellos, alcanzándose el óptimo global en la mayoría de los casos y subóptimos factibles en el resto.

3.3 Tratamiento de variables binarias

Con el fin de ampliar aún más la gama de problemas que podrían resolverse con esta herramienta, al PSO descrito anteriormente se le incorporó la metodología “Angle Modulation” [5] para tratar variables binarias. La misma fue seleccionada principalmente porque permite trabajar directamente con la metaheurística sin necesidad de realizar modificaciones especiales al algoritmo. Se trata de una técnica sencilla y fácil de anexar al PSO y para la cual se han reportado buenos resultados en distintas aplicaciones.

Básicamente, este método consiste en sumar 4 variables continuas adicionales (a , b , c , d) al problema a optimizar, las cuales son coeficientes de una función trigonométrica simple $G(\cdot)$ (6). Dependiendo del valor que tome esta función, el valor de cada variable binaria (j) será igual a uno o cero según el criterio (7).

$$G(I) = \sin(2\pi(I - a) \times b \times \cos(2\pi \times c(I - a))) + d \quad (6)$$

$$\begin{cases} \text{Si } G(I_i) > 0 \rightarrow y_i = 1 \\ \text{Si } G(I_i) \leq 0 \rightarrow y_i = 0 \end{cases} \forall i \in [1, n_b] \quad (7)$$

Los coeficientes a , b , c , d , manipulados como variables por el PSO, controlan la forma de la función sinusoidal $G(I)$. I representa un intervalo que se divide en tantos subintervalos regulares como cantidad de variables binarias (n_b) posea el problema.

De esta manera, la técnica “Angle Modulation” no solo resulta muy compatible con la mecánica de optimización del PSO al manipular un número pequeño de variables continuas adicionales, sino que también vuelve más sencillo el problema a resolver, ya que evita la complejidad combinatoria asociada a explorar el espacio entero correspondiente al problema binario original. Cabe aclarar que, si bien esta técnica proporciona una suerte de “memoria” en la búsqueda con el objetivo de preservar buenas combinaciones de variables binarias, no asegura una exploración exhaustiva del árbol de opciones al estilo “Branch & Cut”, con lo cual es imposible asegurar la convergencia a soluciones globalmente óptimas.

3.4 Tratamiento de objetivos múltiples

Para extender aún más las prestaciones de la herramienta con el objeto de optimizar una amplia gama de problemas de diferentes características, también se incorporó al PSO una técnica para tratar objetivos múltiples. Si bien existe una gran variedad de métodos para resolver esta clase de problemas [6], en este trabajo se implementó una técnica que detecta y expone el frente de Pareto. Esta metodología consiste en que el algoritmo identifique aquellas soluciones pertenecientes a la frontera de Pareto, la cual representa un conjunto de soluciones donde cada una es mejor que las restantes en al menos uno de los objetivos optimizados. Este frente está representado por una curva en caso de problemas de optimización bi-objetivo y por una superficie en caso de problemas tri-

objetivo. Con esta técnica se reduce la subjetividad y arbitrariedad en el tratamiento de los diferentes objetivos, a diferencia de otros métodos populares en los cuales se deben asignar prioridades. En este caso, el tomador de decisión tiene a su disposición diversas soluciones y decide cuál de ellas aceptar de acuerdo a su conocimiento e intuición, al priorizar determinado criterio por sobre los demás [7].

Para poder establecer el frente de Pareto, es necesario primero definir los términos “soluciones dominadas” y “soluciones no dominadas”, los cuales se utilizan para ordenar las distintas soluciones encontradas. Se dice que una solución S_1 domina a otra solución S_2 si se satisfacen dos condiciones (en problemas de minimización de todos los objetivos):

1. La solución S_1 no es peor que S_2 en todos los objetivos (f_k). Es decir, se debería cumplir que: $f_k(S_1) \leq f_k(S_2)$ para todo $k = \{1, \dots, N_{FO}\}$.
2. La solución S_1 es estrictamente mejor que S_2 al menos en un objetivo (f_k). Esto quiere decir que $f_k(S_1) < f_k(S_2)$ para al menos un objetivo $k = \{1, \dots, N_{FO}\}$.

En otras palabras, esta definición indica que la solución S_1 se considera un óptimo de Pareto si es igualmente buena en todos los objetivos y mejor en al menos uno de ellos. De esta manera, entre dos soluciones existen tres posibilidades:

- S_1 domina a S_2 ;
- S_1 es dominada por S_2 ;
- S_1 y S_2 son no dominadas entre sí.

A medida que se identifican las soluciones no dominadas, éstas se almacenan en un archivo externo (A). Se debe limitar el tamaño de este archivo para que no se torne demasiado extenso y se disminuya la velocidad del algoritmo con excesivas operaciones de almacenamiento. Una vez que se establece el número máximo de soluciones a almacenar, se requiere un criterio para establecer qué soluciones se descartan una vez que el archivo está completo. Para ello, en este trabajo se utilizó la metodología propuesta por [8], la cual ordena las soluciones del archivo externo según la distancia de hacinamiento que existe entre ellas.

Para determinar la distancia de hacinamiento (dh) se inicializa su valor en cero para todas las soluciones presentes en A . Posteriormente, se clasifica el conjunto de soluciones según valores de función objetivo en orden ascendente. Luego, se establece el valor de dh como la distancia promedio entre las dos soluciones vecinas. A las soluciones límite, que son aquellas que tienen el menor y mayor valor de la función objetivo, se les asigna, arbitrariamente, un valor infinito en la distancia de hacinamiento para que siempre queden seleccionadas. Este proceso se realiza para cada función objetivo del problema de optimización. Finalmente, se calcula el valor de la distancia de hacinamiento de cada solución, como la suma de todas las dh de cada función objetivo.

Una vez obtenida la distancia de hacinamiento para cada partícula, se las ordena de mayor a menor en el archivo externo A . Si resulta que éste supera el límite de partículas establecido, entonces se eliminan tantas soluciones de la parte inferior de A como sea necesario para cumplir con la dimensión especificada para éste. De esta manera, sólo permanecen las partículas que mayor diversidad aportan, en cuanto a su valor en las

funciones objetivo. El pseudocódigo para el cálculo de la distancia de hacinamiento se presenta en la Fig. 2.

Cabe mencionar que, con esta metodología, cambia la forma de determinar el mejor global (q) del PSO, que antes era un valor único, y ahora puede ser igual a cualquiera de las soluciones presentes en A . Entonces, para establecer q , siguiendo la propuesta de [8], a cada partícula se le asignó aleatoriamente cualquiera de las soluciones correspondientes al 10% de la parte superior de archivo externo (una vez que éste ya se ordenó de mayor a menor dh). De esta manera, se promueve el movimiento del enjambre hacia los puntos más dispersos entre sí.

En cuanto al mejor local (p), sólo es necesario comparar la nueva solución encontrada en esa iteración (x) con la guardada en p . Si x domina a p , entonces se actualiza el mejor local, si no, mantiene su valor.

```

1) Se obtiene el número total de soluciones no dominadas en el repositorio externo  $A$ .
    $n_A = \text{len}(A)$ 
2) Se inicializa la distancia de hacinamiento  $dh$  de cada solución  $s$  en cero.
    $dh[S_i] = 0$ .
3) Se calcula la distancia de hacinamiento de cada solución.
   Para cada función objetivo  $k = \{1, \dots, N_{FO}\}$ :
     Se ordenan las soluciones de mayor a menor valor de la función objetivo  $k$  y
     se calcula la distancia de cada solución según el objetivo  $k$ :
     Para cada  $k$  desde 1 hasta  $N_{FO}$ :
        $S = \text{sort}(S, k)$ 
     Para cada  $S_i$  desde 1 hasta  $n_A$ :
        $dh[S_i] = dh[S_i] + dh_k[S_{i+1}] - dh_k[S_{i-1}]$ 
     Se asigna a la distancia de hacinamiento de las soluciones límite un valor
     infinito.
        $dh[S_0] = dh[S_{n_A}] = \infty$ 

```

Fig. 2. Pseudocódigo del cálculo de la distancia de hacinamiento

3.5 Pseudocódigo PSO MINLP-MO: Implementación propia

La implementación propia del algoritmo con los elementos descritos en las secciones anteriores queda definida según el diagrama de la Fig. 3. En la Fig. 4 se proporciona el pseudocódigo correspondiente. A modo de resumen el optimizador se basa en un PSO básico [2], al cual se le adicionó: (i) el cálculo de “TAV” [3] para permitir la inclusión de restricciones; (ii) la técnica “Angle Modulation” [5] para manipular variables binarias y (iii) una metodología basada en la distancia de hacinamiento para identificar el frente de Pareto [8] a fin de optimizar problemas multi-objetivo. Como plataforma de desarrollo se adoptó el lenguaje de código abierto Python. Se seleccionó esta alternativa

por su enorme popularidad en computación científica, gran disponibilidad de complementos, versatilidad, agilidad de lectura, flexibilidad y transparencia de sintaxis.

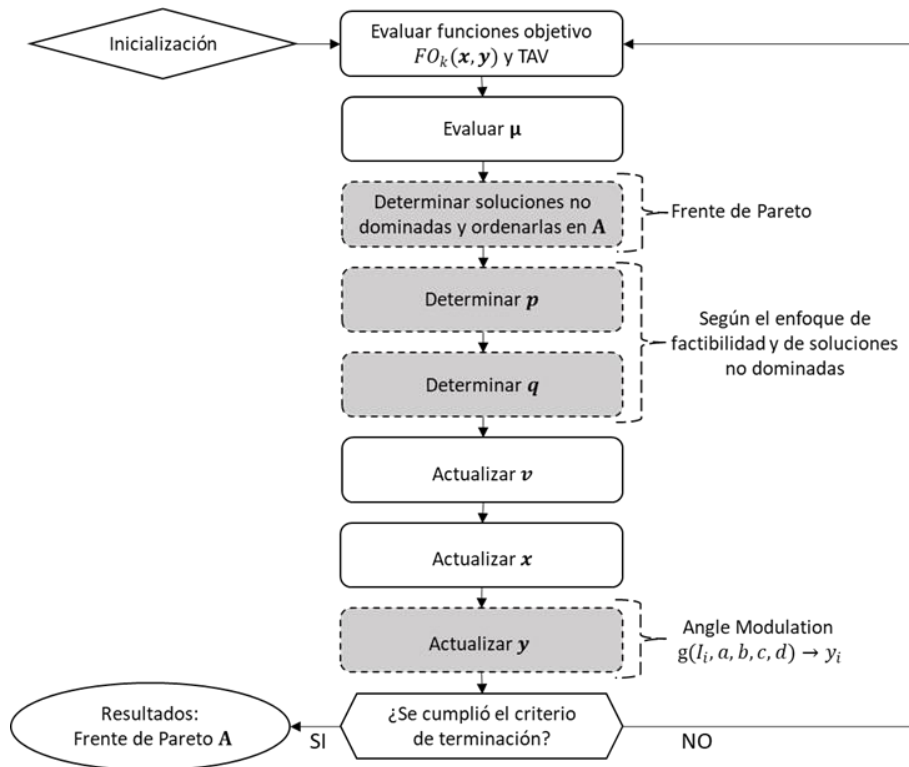


Fig 3. Esquema del PSO con variables continuas, binarias y multi-objetivo

1. Inicialización de cada partícula del enjambre:
 - a. Inicializar la posición aleatoriamente: $x_{ij}^0 = U(x_{ij}^{lo}, x_{ij}^{up})$ donde $x_{ij}^0 = [a, b, c, d, x_1, \dots, x_D]$
 - b. Calcular $g(I_i) = \sin(2\pi(I_i - a) \times b \times \cos(A)) + d$, donde $A = 2\pi \times c(I_i - a)$.
 - c. Determinar las variables binarias: $\begin{cases} Si\ g(I_i) > 0 \rightarrow y_i = 1 \\ Si\ g(I_i) \leq 0 \rightarrow y_i = 0 \end{cases} \forall i \in [1, n_b]$
 - d. Inicializar la velocidad en cero: $v_{ij}^0 = 0$
 - e. Inicializar la mejor posición local: $p_{ij}^0 = x_{ij}^0$.
 - f. Calcular el valor de la función $f_{N_{obj}}(x_i)$ y de $TAV_i(x_i)$.
 - g. Determinar las soluciones no dominadas del óptimo de Pareto: S_i .
 - h. Ordenar S_i según su dh en orden decreciente.
 - i. Almacenar S_i en el archivo externo A .
 - j. Determinar el mejor global: $q_i = U(10\%A)$.
 - k. Calcular el valor de relajación inicial: $\mu^0 = Promedio(TAV)$.
2. Repetir hasta que se cumpla el/los criterios de terminación elegidos:
 - a. Actualizar la velocidad de cada partícula:

$$v_{ij}^{k+1} = w^k v_{ij}^k + c_1 r_{1ij} (p_{ij}^k - x_{ij}^k) + c_2 r_{2ij} (q_j - x_{ij}^k)$$
 - b. Actualizar la posición de cada partícula:

$$x_{ij}^{k+1} = x_{ij}^k + v_{ij}^{k+1}$$
 - c. Calcular $g(I_i) = \sin(2\pi(I_i - a) \times b \times \cos(A)) + d$, donde $A = 2\pi \times c(I_i - a)$.
 - d. Determinar las variables binarias: $\begin{cases} Si\ g(I_i) > 0 \rightarrow y_i = 1 \\ Si\ g(I_i) \leq 0 \rightarrow y_i = 0 \end{cases} \forall i \in [1, n_b]$
 - e. Evaluar el valor de las funciones objetivo $f_{N_{obj}}(x_i^{k+1})$ y de $TAV_i(x_i^{k+1})$.
 - f. Determinar las soluciones no dominadas del óptimo de Pareto: S_i .
 - g. Ordenar S_i según su dh en orden decreciente.
 - h. Almacenar S_i en el archivo externo A .
 - i. Evaluar el valor de relajación: $\mu^{k+1} = \mu^k \left(1 - \frac{FF}{N}\right)$,
 - j. Actualizar el mejor local: $p_{ij} = x_{ij}^{k+1}$.
 - k. Actualizar el mejor global: $q_i = U(10\%A)$.
3. Frente Pareto = A .

Fig. 4. Pseudocódigo del PSO con variables continuas, binarias y multi-objetivo

4 Resultados

Como se comentó anteriormente, en [4] se testeó el PSO continuo con el tratamiento de restricciones basado en TAV, optimizando diversas funciones benchmark con diferentes números de ecuaciones, variables y no linealidades, lográndose identificar el óptimo global en la mayoría de los casos estudiados y buenas soluciones factibles en los restantes.

En este trabajo nos limitaremos el testeo del algoritmo en lo que respecta a la identificación de la frontera de Pareto. Para ello se tomaron ocho problemas benchmark pequeños, los cuales cuentan con dos o tres funciones objetivo y distintas restricciones, empleados típicamente en estudios de algoritmos de optimización multi-objetivo. En particular, el algoritmo se probó sobre cuatro problemas con dos funciones objetivo: MO1 [9] (Fig. 5), MO2 [10] (Fig. 6), MO3 [11] (Fig. 7) y MO4 [11] (Fig. 8); y otros cuatro con tres funciones objetivo: MO5 [12] (Fig. 9), MO6 [12] (Fig. 10), MO7 [13] (Fig. 11) y MO8 [13] (Fig. 12). Para todos ellos se presenta la correspondiente formulación y una comparación gráfica entre el frente de Pareto reportado en la literatura para el problema en cuestión, y el hallado por el PSO propio, con el fin de establecer la calidad de la solución encontrada.

En todos los casos, se utilizaron los siguientes parámetros del PSO: cantidad de partículas (N) = 100, constante de aceleración cognitiva (c₁) = 1,5, constante de aceleración social (c₂) = 1,5, peso de inercia (w) = 0,75, iteraciones máximas (k_{máx}) = 5000 y la dimensión del archivo externo A (dim_A) = 100. Estos valores se tomaron de [4], quienes testearon el PSO con estos parámetros sobre diferentes funciones benchmark de distinta complejidad, reportándose en todos los casos buenos rendimientos.

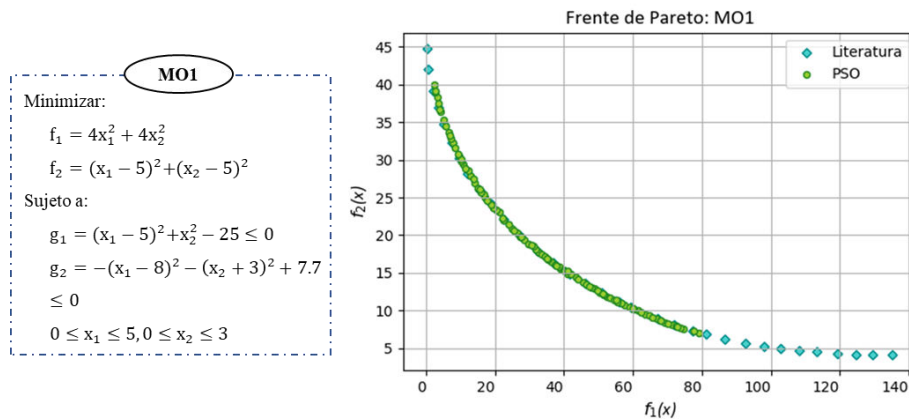


Fig. 5. Fronte de Pareto MO1 literatura (rombos celestes) vs PSO propio (círculos verdes)

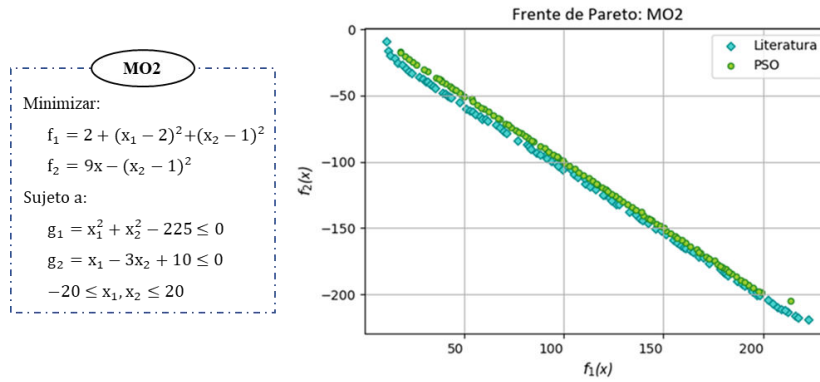


Fig. 6. Frente de Pareto MO2 literatura (rombos celestes) vs PSO propio (círculos verdes)

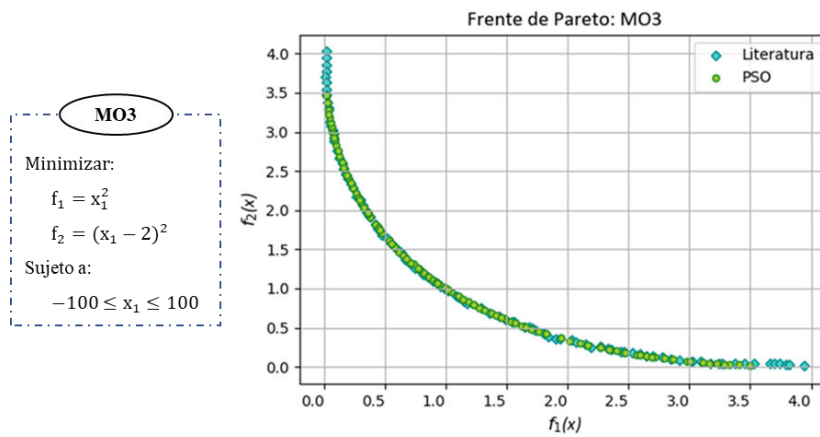


Fig. 7. Frente de Pareto MO3 literatura (rombos celestes) vs PSO propio (círculos verdes)

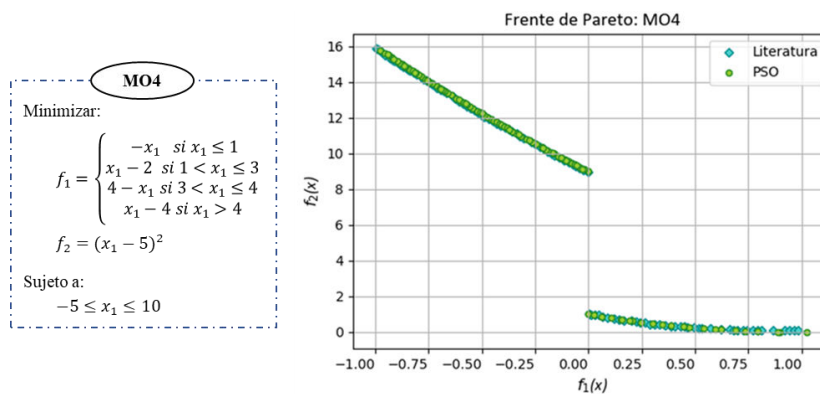


Fig. 8. Frente de Pareto MO4 literatura (rombos celestes) vs PSO propio (círculos verdes)

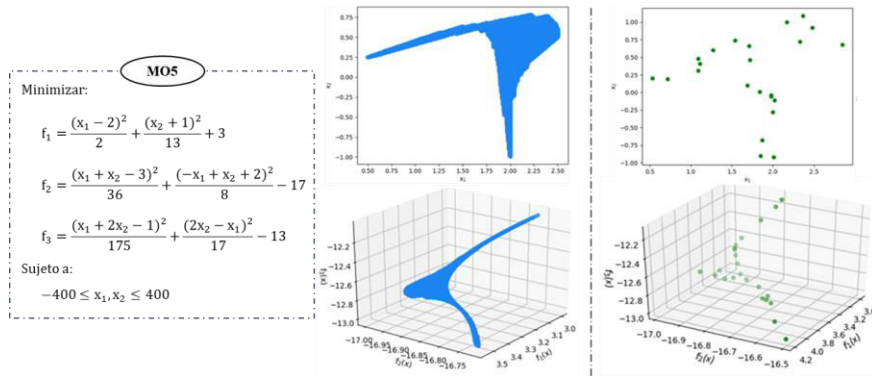


Fig. 9. Frente de Pareto MO5 literatura (izquierda) vs PSO propio (derecha)

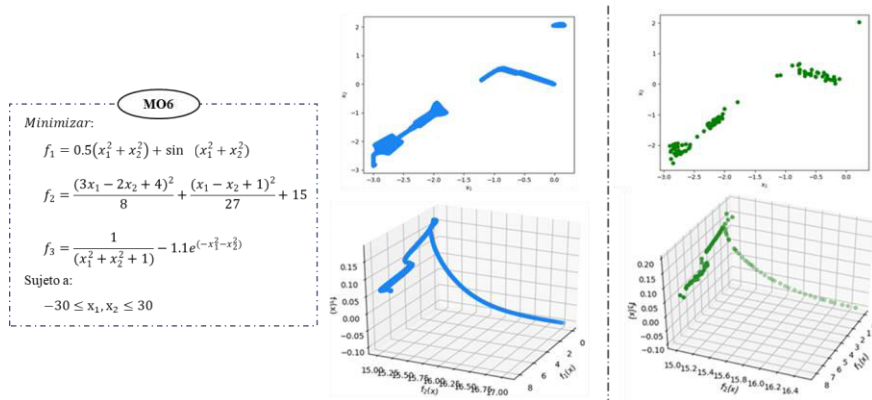


Fig. 10. Frente de Pareto MO6 literatura (izquierda) vs PSO propio (derecha)

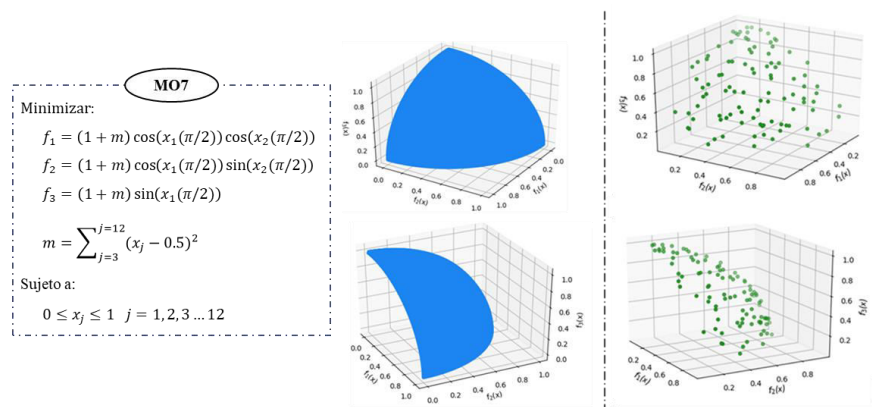


Fig. 11. Frente de Pareto MO7 literatura (izquierda) vs PSO propio (derecha)

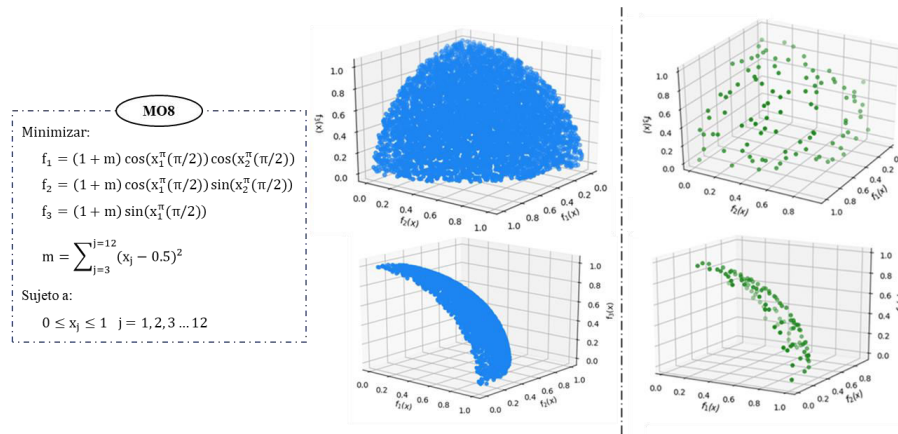


Fig. 12. Frente de Pareto MOS literatura (izquierda) vs PSO propio (derecha)

Como puede observarse en las figuras anteriores, en todos los casos de testeo propuestos, los resultados obtenidos con la herramienta desarrollada son muy similares a la solución reportada. Asimismo, cabe mencionar que todas las soluciones son factibles. Esto se concluye ya que los valores de TAV de cada frontera de Pareto (datos no mostrados) son siempre igual a cero, es decir, no se viola ninguna restricción. Los experimentos se realizaron en una computadora Intel(R) Core (TM) i7-3537U CPU @ 2.00GHz 2.50GHz con una memoria RAM de 8GB, demandando, en todos los casos, unos pocos minutos de cómputo para completar el número de iteraciones adoptado.

5 Aplicación: Manejo Integrado de Malezas

En esta sección se propone optimizar un modelo matemático de simulación multianual de escenarios de Manejo Integrado de Malezas (MIM). Por una descripción completa de este problema ver [14]. El MIM implica combinar distintas prácticas (químicas, culturales, mecánicas, etc.) para el realizar el control de las malezas con el objeto de reducir la dependencia exclusiva en los herbicidas, los cuales poseen un impacto ambiental considerable. Las principales variables de decisión son, en cada año del periodo estudiado: (i) el tipo de cultivo a sembrar; (ii) su densidad de siembra; (iii) el tipo de cultivar; (iv) la fecha de siembra; (v) las aplicaciones de herbicidas (pre-siembra y pos-emergencia) y (vi) las fechas de las aplicaciones. Las funciones objetivo consideradas son el impacto ambiental por aplicar los agroquímicos y el beneficio económico de la actividad. En particular, el modelo bio-económico propuesto en [14] permite simular los efectos económicos y medioambientales que podrían ocasionar diferentes estrategias de control sobre la dinámica demográfica de la maleza *Avena fatua* L. (AVEFA) cuando compite con trigo o cebada, un sistema productivo típico del sudoeste de la provincia de Buenos Aires, a lo largo de un horizonte de 5 años. Ese modelo permite

simular un conjunto finito de escenarios o casos de estudio que representan situaciones que se podrían llegar a dar en la práctica.

Sin embargo, la cantidad de estrategias MIM a lo largo de unos pocos años es enorme, incluso para un número modesto de cultivos y alternativas de control, como lo son los del caso bajo estudio. Para poder explorar sistemáticamente el extenso espacio de posibles opciones de manejo con el fin de hallar nuevas y mejores soluciones surge naturalmente la posibilidad de optimizar dicho modelo. Para poder hacerlo se requiere de un optimizador mixto entero no lineal multi objetivo como el descrito en las secciones anteriores, ya que el problema MIM cuenta con restricciones, variables binarias y múltiples objetivos: medioambiental y económico.

Este experimento persigue identificar posibles tácticas agronómicas que reflejen prácticas y objetivos aproximados a los tradicionales empleados en la zona bajo estudio, que utilizan típicamente aplicaciones de herbicidas para controlar la maleza tanto antes de la siembra (control pre-siembra) como durante el periodo de crecimiento del cultivo (control pos-emergencia) en cada cada temporada. La labranza es una práctica poco utilizada debido a los altos costos que insume y a los problemas de erosión de suelos que acarrea. Por esta razón se excluye esta posibilidad como práctica pre-siembra en este trabajo y se implementan en todos los casos, dos aplicaciones de glifosato (G) en el periodo de barbecho. Como indicador de impacto medioambiental se emplea el índice *EIQ* [16] y como indicador económico el valor presente neto correspondiente al periodo analizado, el cual tiene en cuenta el ingreso por venta del cultivo y todos los costos involucrados, incluyendo los de siembra, cosecha y controles, y las eventuales limpiezas del cereal por contaminación con semillas de maleza. Se considera el período 2014-2018, con los datos climáticos correspondientes, que condicionan tanto el perfil de emergencia de la maleza como el crecimiento del cultivo. De esta manera, el problema de optimización planteado queda definido según (8).

$$F \begin{cases} f_1 = -VPN \\ f_2 = EIQ \end{cases} \quad (8)$$

Sujeto a:

$$h_1 = Cantidad_{LabranzaTOTAL} = 0$$

La optimización se ejecutó en una computadora Intel(R) Core (TM) i7-3537U CPU @ 2.00GHz 2.50GHz con una memoria RAM de 8GB. Los parámetros del PSO para todas las optimizaciones realizadas son: $N = 100$, $c_1 = 1,5$; $c_2 = 1,5$; $w = 0,75$; $k_{m\acute{a}x} = 2500$ y $dim_A = 20$. Para la cantidad de iteraciones máximas adoptada el tiempo de optimización demandado fué de aproximadamente 11,20 h.

En este caso la frontera de Pareto (Fig. 13) se compone sólo de cuatro soluciones, dado que no se encuentran otras estrategias que provoquen alguna mejoría en al menos uno de los objetivos.

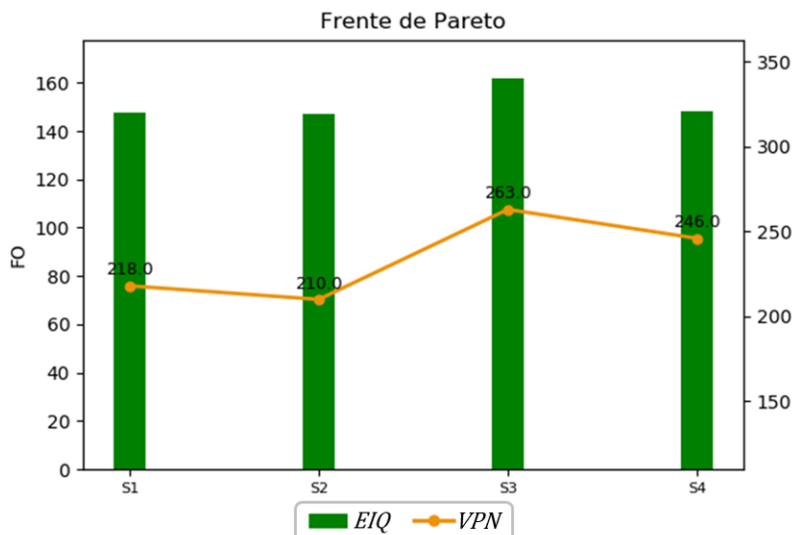


Fig. 13. Resultados optimización

La Tabla 1, que detalla cada una de las soluciones del frente de Pareto alcanzado luego de optimizar el problema (8), muestra que todas proponen la siembra de cebada (C) en una mayor cantidad de años que los de trigo. Además, en los casos en que se propone la siembra trigo en los primeros años, se selecciona un cultivar altamente competitivo (T-AC) (S3). Estas decisiones están claramente motivadas por la necesidad de efectuar un buen control de la maleza durante los primeros años (de mayor infestación) mediante el empleo de los cultivos con mayores habilidades competitivas (cebada y trigo competitivo). Los cultivares de trigo estándar (T-E), con mayor rendimiento potencial pero menor capacidad de competencia (S1 y S4) se dejan para las últimas temporadas con el objeto de compensar el valor presente neto. Asimismo, S4 permite visualizar cómo una mayor cantidad de aplicaciones de herbicidas (dos en pos-emergencia) no conduce directamente a obtener el mejor VPN (alcanza el 69% del mayor), pero sí repercute en un mayor impacto ambiental (aunque no el máximo, dado que una sola aplicación de diclofop-metil (D) incide más en el EIQ que dos de pinoxaden (P), como se verifica en el EIQ de S3).

Tabla 1. Resultados optimización

Soluciones	2014				2015				2016				2017				2018			
	Cultivo	Densidad	Control		Cultivo	Densidad	Control		Cultivo	Densidad	Control		Cultivo	Densidad	Control		Cultivo	Densidad	Control	
			Pre.	Post.			Pre.	Post.			Pre.	Post.			Pre.	Post.			Pre.	Post.
S1	C	257	G	P	C	258	G	-	C	260	G	-	T-E	310	G	-	T-E	242	G	-
S2	C	268	G	-	C	266	G	-	C	254	G	-	C	264	G	-	C	266	G	-
S3	T-AC	385	G	D	T-AC	301	G	-	C	254	G	-	C	262	G	-	C	272	G	-
S4	C	266	G	P	C	266	G	P	C	253	G	-	C	261	G	-	T-E	386	G	-

De las cuatro estrategias se presentan a continuación, la dinámica poblacional de AVEFA y el factor de competencia de los cultivos para las soluciones S3 y S2 que son las que logran el mayor *VPN* y menor *EIQ* respectivamente.

La solución S3 (Fig. 14) propone la siembra de trigo altamente competitivo los primeros dos años, mientras que en los últimos tres opta por cebada. Como control pos-emergente solo realiza una aplicación el primer año, el de mayor infestación. Esta estrategia controla eficazmente a la maleza, ya que en el período de competencia con el cultivo los picos de densidad se mantienen por debajo de 30 individuos/m² (a excepción del 2014, donde un pico potencialmente elevado es intervenido con el diclofop-metil, D), lo que permite alcanzar los mejores rendimientos del cultivo a costa de ser la propuesta con mayor *EIQ*.

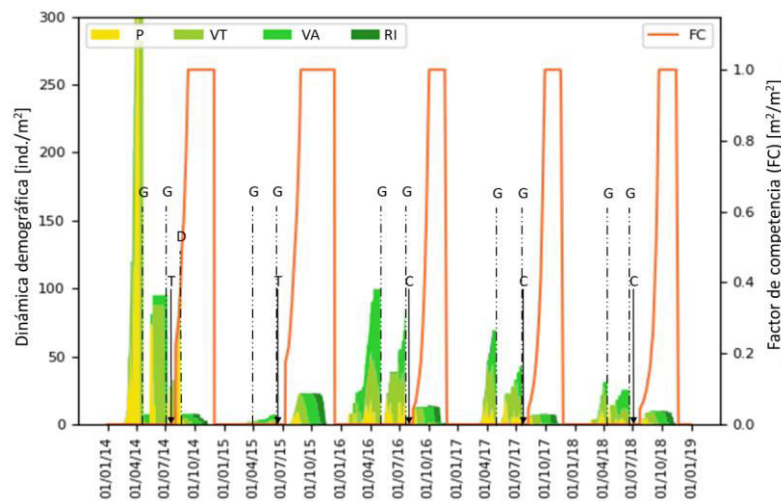


Fig. 14. Dinámica poblacional de AVEFA (*P*: Plántula, *VT*: Vegetativo Temprano, *VA*: Vegetativo Avanzado, *RI*: Reproductivo Inicial) y factor de competencia (*FC*) de S3

Por su parte, S2 implementa la siembra de cebada en todo el horizonte de planeamiento y no realiza ningún tipo de control pos-emergente, presentando el menor impacto ambiental de todas las soluciones halladas (Fig. 15). La combinación de estas medidas provoca un mayor crecimiento de la maleza, presentando un pico máximo de 255 ind/m² en 2015 y logrando, en general, un control más deficiente de AVEFA que en el caso anterior. Esto repercute en el rendimiento del cultivo y el margen bruto de la actividad, conduciendo al menor *VPN* del frente de Pareto.

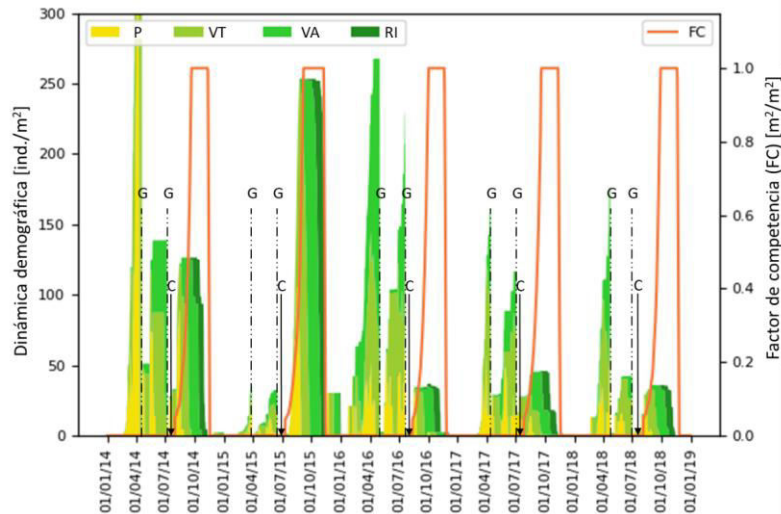


Fig. 15. Dinámica poblacional de AVEFA (*P*: Plántula, *VT*: Vegetativo Temprano, *VA*: Vegetativo Avanzado, *RI*: Reproductivo Inicial) y factor de competencia (*FC*) de S2

6 Conclusiones

En este trabajo se implementó un optimizador continuo basado en PSO. Para extender sus prestaciones, se incorporó al algoritmo básico una técnica para manipular restricciones, otra para tratar variables binarias y un método para optimizar objetivos múltiples.

El algoritmo se utilizó en este trabajo para optimizar diferentes problemas benchmark pequeños de dos o tres funciones objetivo, con diferentes restricciones y características. Los resultados obtenidos indican que la herramienta demuestra un buen desempeño general, ya que se alcanzan soluciones factibles y similares a las reportadas en la literatura. Si bien los ejemplos seleccionados son de pequeña dimensión, permiten comprobar el buen funcionamiento de la implementación.

Por estas razones, sumado a evidencia generada en trabajos previos [4], entendemos que el PSO MINLP-MO desarrollado puede adaptarse al estudio de problemas mixto entero no-lineales multi-objetivo resultantes de aplicaciones a la ingeniería en sus diversas ramas. Específicamente, en este trabajo se lo ha utilizado para sistematizar el estudio del manejo integrado de malezas, un problema de gran interés relacionado con la sostenibilidad medio-ambiental y económica de la actividad agronómica. Este problema es de considerable dificultad, tanto por la necesidad de representar adecuadamente el sistema cultivo-maleza bajo estudio mediante un modelo bio-económico apropiado, como por la necesidad de explorar un espacio de opciones grande a lo largo de un periodo de varios años.

Es importante destacar que con este solver no se pretende competir con plataformas comerciales de modelamiento y optimización. Lo que se busca principalmente es dis-

poner de un instrumento accesible para desarrollar proyectos de investigación y transferencia, que proporcione soluciones de aceptable eficiencia y no requiera presupuestos para adquisición y mantenimiento de licencias comerciales de software lo que, en la práctica, suele resultar una restricción importante para transferir y desarrollar colaborativamente.

7 Referencias

1. Woon, S. F., & Rehbock, V. (2010). A critical review of discrete filled function methods in solving nonlinear discrete optimization problems. *Applied mathematics and computation*, 217(1), 25-41.
2. Kennedy, J., Eberhart, R. (1995). Particle swarm optimization (PSO). In *Proc. IEEE International Conference on Neural Networks*, Perth, Australia (pp. 1942-1948).
3. Zhang, H., & Rangaiah, G. P. (2012). An efficient constraint handling method with integrated differential evolution for numerical and engineering optimization. *Computers & Chemical Engineering*, 37, 74-88.
4. Damiani, L., Diaz, A. I., Iparraquirre, J., & Blanco, A. M. (2020). Accelerated particle swarm optimization with explicit consideration of model constraints. *Cluster Computing*, 23(1), 149-164.
5. Pampara, G., Franken, N., & Engelbrecht, A. P. (2005). Combining particle swarm optimisation with angle modulation to solve binary problems. In *2005 IEEE congress on evolutionary computation (Vol. 1, pp. 89-96)*. IEEE.
6. Ngatchou, P., Zarei, A., & El-Sharkawi, A. (2005). Pareto multi objective optimization. In *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems (pp. 84-91)*. IEEE.
7. Selçuklu, S. B., Coit, D. W., & Felder, F. A. (2020). Pareto Uncertainty Index for Evaluating and Comparing Solutions for Stochastic Multiple Objective Problems. *European Journal of Operational Research*.
8. Raquel, C. R., & Naval Jr, P. C. (2005). An effective use of crowding distance in multiobjective particle swarm optimization. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation (pp. 257-264)*.
9. Binh, T. T., & Korn, U. (1997). Multiobjective evolution strategy for constrained optimization problems. In *Proceedings of the 15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics, Berlin (Vol. 357, p. 362)*.
10. Chankong, V., & Haimes, Y. Y. (2008). *Multiobjective decision making: theory and methodology*. Courier Dover Publications.
11. Schaffer, J. David (1984). *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. *Proceedings of the First Int. Conference on Genetic Algorithms*, Ed. G.J.E Grefenstette, J.J. Lawrence Erlbaum (PhD). Vanderbilt University.
12. Coello, C. A. C., Lamont, G. B., & Veldhuizen, D. A. V. (2007). *MOEA Test Suites. Evolutionary Algorithms for Solving Multi-Objective Problems: Second Edition*, 175-232.
13. Kaveh, A., & Mahdavi, V. R. (2019). Multi-objective colliding bodies optimization algorithm for design of trusses. *Journal of Computational Design and Engineering*, 6(1), 49-59.
14. Damiani L., Molinari, F. A., Blanco, A., & Chantre, G. R. (2020). Modelo de Simulación para el Manejo Integrado de Malezas: Análisis económico y medioambiental. In *XII Congreso de AgroInformática (CAI)-JAIHO 49*.

15. Kovach, J; Petzold, C; Degnil, J y Tette, J. (1992). A method to measure the environmental impact of pesticides. New York's Food and Life Sciences Bulletin 139: 1-8.