Influencia del operador de cruce en la optimización de funciones con restricciones usando métodos de penalización *

D. Ortiz-Boyer, C. Hervás-Martínez
N. García-Pedrajas

Dept. de Informática y Análisis Numérico
Campus de Rabanales, Edif. Einstein
Universidad de Córdoba
14071 Córdoba
{dortiz, chervas, npedrajas}@uco.es

A. Martínez-Estudillo

Escuela Técnica Empresarial Agrícola
Escritor Castilla Aguayo s/n
Universidad de Córdoba
14071 Córdoba
acme@etea.es

Resumen

Los Algoritmos Genéticos (AG) son unas de las principales herramientas utilizadas para la optimización de funciones con restricciones. Una de las técnicas más comunes y versátiles para el manejo de restricciones consiste en penalizar aquellas soluciones cuyos valores no satisfacen las restricciones. De esta forma el problema se transforma en un problema de optimización sin restricciones en el que la forma de la función a optimizar de la zona no factible se ve alterada por el método de penalización. En este contexto el operador de cruce debería tender a generar individuos en entornos próximos a la zona factible que favorecieran la convergencia a soluciones factibles. En este trabajo vamos ha analizar la influencia del operador de cruce en este tipo de problemas. En el conjunto de pruebas hemos incluido problemas con restricciones lineales y no lineales. Los resultados obtenidos confirman la importancia del operador de cruce y ponen de manifiesto que la filosofía de cruce que usa CIXL2 es muy adecuada para este tipo de problemas.

1. Introducción

La mayoría de los problemas de optimización del mundo real se circunscriben a funciones no lineales sometidas a una serie de restricciones. Este tipo de problemas son conocidos como problemas de programación no-lineal (PPNL). Una de las herramientas más interesantes para abordar PPNL son los Algoritmos Genéticos, y especialmente los Algoritmos Genéticos con Codificación Real (AGCRs) en los que cada gen es una variable real del problema a resolver. Los AGCRs son algoritmos probabilísticos paralelos de búsqueda muy robustos que han sido ampliamente utilizados en la optimización de problemas sin restricciones. La utilización de estos en entornos con restricciones requiere la utilización de mecanismos que posibiliten la incorporación de las restricciones en el proceso evolutivo. Entre los más utilizados se encuentran los que siempre mantienen a la población dentro de la zona factible, y los que permiten la existencia de individuos fuera de ella, siendo éstos penalizados utilizando una función de coste [6].

Restringir la búsqueda sólo a las soluciones factibles hace que sea difícil encontrar un esquema que conduzca a la población hacia el óptimo ya que es frecuente que la solución óptima esté en el límite de la región factible, lo que conlleva que la mayoría de las soluciones más próximas al genotipo del óptimo sean no factibles. La aproximación más común en AGs para manejar restricciones consiste en penalizar aquellas soluciones cuyos valores no satisfacen las restricciones. De esta forma se

*Este trabajo ha sido financiado por el proyecto TIC2002-04036-C05-02 de la CICYT y fondos FEDER

amplia el espacio de búsqueda del problema al mismo tiempo que se distorsiona la función de aptitud de la zona no factible con el objetivo de que, si bien sea posible generar individuos en ella, la probabilidad de que estos sobrevivan sea menor.

En este contexto los mecanismos de generación de individuos como son el cruce y la mutación tienen una gran importancia ya que de ellos depende la generación de nuevos individuos y su factibilidad. Operadores adecuados en la optimización sin restricciones pueden no serlo tanto en entornos con restricciones. Es por ello que el concepto de exploración y explotación aplicado a problemas sin restricciones se ve reformulado cuando lo transladamos a PPNL. Así, operadores con un buen comportamiento en entornos sin restricciones pueden no ser tan apropiados en entornos con restricciones, de ahí la necesidad de hacer un estudio que arroje luz sobre este tema en cuestión.

En este trabajo nos centraremos en el estudio del operador de cruce ya que es el operador más importante de un AGCR [5]. Así, será fundamental la capacidad de dicho operador de establecer un equilibrio entre exploración y explotación [4], que aunque permita la generación de individuos en la zona no factible, prime la creación de estos en las zona factible.

En este trabajo estudiaremos diferentes operadores de cruce comúnmente usados en la optimización de problemas sin restricciones. Para evaluar su rendimiento usaremos diferentes métodos de penalización aplicados a un conjunto de funciones no lineales con restricciones tanto lineales como no lineales ampliamente utilizadas en la bibliografía especializada.

La estructura del artículo es la siguiente. En la Sección 2 explicamos en que consisten los métodos de penalización y cuales son los más importantes. En la sección 3 se describen brevemente los cruces empleados en el estudio. La sección 4 se dedica a la descripción de los experimentos realizados. En la sección 5 se muestran los resultados obtenidos. Finalmente, en la sección 6 se muestran las conclusiones a las que hemos llegado y las posibles futuras líneas de investigación.

2. Métodos basados en funciones de penalización

Un problema de satisfacción de restricciones se puede expresar como un PPNL de la forma:

$$\begin{array}{l} \text{Encontrar } \mathbf{x} \text{ que optimize } f(\mathbf{x}), \\ \mathbf{x} = (x_1, ..., x_p) \in \mathcal{F} \subset S \\ \text{sujeto a:} \\ \qquad g_i(\mathbf{x}) \leq 0, \ j = 1, ..., q \\ \qquad h_i(\mathbf{x}) = 0, \ j = q+1, ..., p \end{array} \tag{1}$$

donde g_i y h_i pueden ser funciones lineales o no lineales. $\mathcal{S}$ constituye el espacio de búsqueda total y se divide en dos subespacios disjuntos: el subespacio de búsqueda factible, $\mathcal{F}$, y el subespacio de búsqueda no factible, $\mathcal{U}$.

Los métodos basados en funciones de penalización son los más comúnmente usados por los AGs para la optimización de problemas con restricciones, especialmente cuando algunas de estas son no-lineales. Como hemos señalado anteriormente, las funciones de penalización se usan para transformar un problema de optimización con restricciones en un problema de optimización sin restricciones mediante la modificación de la función de aptitud de cada individuo de la siguiente manera:

$$f_p(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{, si } \mathbf{x} \in \mathcal{F} \\ f(\mathbf{x}) + Q(\mathbf{x}) & \text{, en otro caso} \end{cases} \tag{2}$$

donde $Q(\mathbf{x})$ representa la penalización para un individuo no factible o el costo de la reparación de ese individuo. Si el individuo es factible, es decir $\mathbf{x} \in \mathcal{F}$, entonces no sufre penalización. La mayoría de los métodos de penalización usan un conjunto de funciones, ϕ_i $(1 \leq i \leq p)$, para construir la función de penalización. La función f_i mide la violación de la restricción i de la siguiente forma.

$$\phi_i(\mathbf{x}) = \begin{cases} \text{máx}\{0, g_i(\mathbf{x})\}, & \text{if } 1 \leq i \leq q \\ |h_i(\mathbf{x})|, & \text{if } q+1 \leq i \leq p \end{cases} \tag{3}$$

La mayoría de las dificultades surgen como consecuencia de que la solución óptima yace frecuentemente en el límite de la región factible, lo que conlleva que la mayoría de las soluciones más próximas al genotipo del óptimo son no factibles. Así pues, el restringir la búsqueda sólo a las soluciones factibles o la implantación de penalizaciones muy severas hace que sea difícil encontrar un esquema que

conduzca a la población hacia el óptimo. De manera contraria, si la penalización no es suficientemente severa, entonces la búsqueda se realizaría sobre una región muy amplia y la mayor parte de la evolución se empleará para explorar regiones lejanas a la región factible. A continuación y para poder analizar los datos experimentales explicaremos brevemente algunos de los métodos más comúnmente usados en la bibliografía.

2.1. Penalizaciones Estáticas

En este método se utiliza una métrica basada en el número de restricciones que no se satisfacen [1]. La función objetivo penalizada de un individuo para un problema con m restricciones es $f_p(\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^{m} C_i \phi_i^k$, donde $f(\mathbf{x})$ es la función objetivo sin penalizar, el parámetro C_i es una constante impuesta por la violación de la restricción i (suele tomar los valores $\{0{,}5, 1, 100\}$), ϕ_i es la distancia métrica de una restricción i aplicada a la solución $\mathbf{x}$, definida por la ecuación 2, y k es un exponente definido por el usuario (suele fijarse a 1 o 2). En este trabajo utilizaremos $C_i = 100$ y $k = 1$.

2.2. Penalizaciones Dinámicas

Una variación de las funciones de penalización basadas en la distancia consiste en, incrementar la severidad de la penalización para una distancia dada conforme avanza el proceso de búsqueda. Algunos investigadores han argumentado que las penalizaciones dinámicas trabajan mejor que las penalizaciones estáticas. Sin embargo, en la practica, es tan difícil derivar una buena función de penalización dinámica como lo era determinar el valor de los parámetros en las penalizaciones estáticas.

Método de Joines y Houck Jeffrey A. Joines y Christopher R. Houck propusieron el siguiente método en el que los individuos son evaluados en la generación t utilizando la función $f_p(\mathbf{x}, t) = f(\mathbf{x}) + (C \times t)^{\alpha} \sum_{j=1}^{m} |\phi_j^{\beta}(\vec{x})|$, donde C, α y β son constantes que suelen tomar los valores $C = 0{,}5$ y $\alpha = \beta = 1$ [1].

GENOCOP II (GEnetic algorithm for Numerical Optimization for COnstrained Problems) [6] es una versión de GENOCOP I especialmente adaptada para trabajar con problemas con restricciones no-lineales. El algoritmo es el siguiente:

- Dividir todas las restricciones en cuatro subconjuntos: ecuaciones lineales (L_e), inecuaciones lineales (L_i), ecuaciones no lineales (N_e) e inecuaciones no lineales (N_i).
- Seleccionar un individuo que cumpla las restricciones lineales como punto de partida de la búsqueda. La población inicial consistirá en copias de este individuo. Este punto de partida no tiene porqué ser factible.
- Inicializar la temperatura inicial $\tau = \tau_0$
- Evolucionar la población utilizando GENOCOP I y la función de aptitud penalizada $f_p(\mathbf{x}, \tau) = f(\mathbf{x}) + \frac{1}{2\tau} \sum_{j \in \mathcal{A}} \phi_j^2(\mathbf{x})$
- Si $\tau < \tau_f$ detener la búsqueda; si no, decrementar la temperatura $\tau = 10^{-1} \dot{\tau}_{t-1}$, repetir los pasos previos del algoritmo y utilizar la mejor solución como punto de partida de la siguiente iteración.

$\mathcal{A}$ es el vector de restricciones activas. El conjunto de las restricciones activas está formado por las ecuaciones no lineales y las restricciones expresadas mediante inecuaciones no lineales que han sido violadas. Los valores sugeridos [6] son: $\tau_0 = 1$ y $\tau_f = 0{,}000001$.

Método de Smith y Tate La función de penalización adaptable propuesta por Smith y Tate usa tanto la longitud de la búsqueda como la severidad de las restricciones en su función de penalización [1]. Esta función de penalización involucra la estimación de un *umbral cercano a la factibilidad* (NFT) para cada restricción. Conceptualmente, el NFT se define como umbral de distancia de la región factible a la cual el usuario considera que la búsqueda está 'razonablemente' cercana a la región factible. La formulación de la función objetivo penalizada propuesta es: $f_p(\mathbf{x}, t) = f(\mathbf{x}) + (F_{feas}(t) - F_{all}(t)) \sum_{i=1}^{m} \left(\frac{\phi_i}{NFT_i} \right)^k$, donde $f_{all}(t)$ representa el valor sin penalizar de la mejor solución

encontrada por ahora, y $F_{feas}(t)$ representa el valor de la mejor solución factible hasta ahora. $NFT = \frac{NFT_0}{1+\Lambda}$, donde NFT_0 es un limite superior para NFT, Λ es un parámetro de búsqueda dinámica usado para ajustar NFT basado en la búsqueda histórica. Λ se suele definir de manera dinámica en función del número de la generación como $\Lambda = \lambda t$. Se debe de evitar que el NFT se aproxime a cero ni muy rápidamente ni muy lentamente. Además, no existe ningún método para poder calcular NFT_0, por lo que utilizaremos para cada función los valores recomendados en [1].

2.3. Método de Kuri

Este método fue desarrollado por Kuri y la aptitud de un individuo viene determinada por:

$$f_p(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{si factible} \\ K - \sum_{i=1}^{s} \frac{K}{m} & \text{en otro caso} \end{cases} \quad (4)$$

donde m es el número total de restricciones, s es el número de restricciones que se satisfacen y K es una constante grande que suele tomar el valor de 1×10^9 [1]. Nótese que todos los individuos que violan el mismo numero de restricciones reciben la misma penalización, debido a que sus aptitudes no se computan, independientemente de lo cercanos que estén de la región factible.

3. Operadores de Cruce

AGCRs son métodos de búsqueda de propósito general cuyos operadores de cruce y mutación establecen un equilibrio entre exploración y explotación del espacio de búsqueda. El operador de cruce juega el papel más importante. Dicho operador combina las características de dos o más individuos padres para generar uno o más hijos con el objetivo de que el intercambio de información entre buenos padres produzca mejores hijos.

Algunos operadores de cruce, como el aritmético [4], generan individuos dentro de los límites de los padres. De este modo, el operador de cruce implementa un búsqueda en profundidad o explotadora, dejando la búsqueda en amplitud o exploradora en manos del operador de mutación. Esta política, aunque intuitivamente natural, hace que la población converja a puntos centrales del espacio de búsqueda, produciendo una rápida disminución de la diversidad de la población que normalmente produce la convergencia prematura a un óptimo local.

Otros, como BLX-α [3], SBX [2] y UNDX [7] generan individuos tanto en la zona de exploración como en la de explotación. De esta forma es posible evitar la perdida de diversidad y evitar así la convergencia prematura. Sin embargo, en PPNL no está claro que este tipo de operadores suponga una ventaja, ya que podrían producir demasiados individuos en la zona no factible del problema.

Dentro de esta última categoría podemos encuadrar el cruce basado en intervalos de confianza CIXL2 [8] que utiliza las características estadísticas de localización y dispersión de la población de mejores individuos para dirigir el proceso de búsqueda. Esta filosofía puede resultar muy ventajosa cuando la función objetivo se ve sometida a un proceso de penalización cuyo objetivo es que los individuos de las zonas no factibles sean menos adecuados que los de las zonas factibles, favoreciendo que el proceso de búsqueda tienda a generar individuos factibles. Por tanto, si la función de penalización es adecuada, los mejores individuos se encontrarán en la zona factible o en sus proximidades con lo que la aplicación de CIXL2 intensificará la búsqueda en esta zona.

Para ver hasta que punto la argumentación anterior es valida realizaremos en este artículo un estudio de la influencia que el operador de cruce tiene en la resolución de PPNL utilizando funciones de penalización. Y para ello a continuación explicaremos cada uno de los operadores utilizados.

Sea $\beta^{f_1} = \{\beta_1^{f_1}, \beta_2^{f_1}, ..., \beta_i^{f_1}, ..., \beta_p^{f_1}\}$ y $\beta^{f_2} = \{\beta_1^{f_2}, \beta_2^{f_2}, ..., \beta_i^{f_2}, ..., \beta_p^{f_2}\}$ dos padres con p genes los cruce utilizados en este trabajo se definen como:

Cruce Aritmético. Dos hijos $\beta^{s_1} = \{\beta_1^{s_1}, \beta_2^{s_1}, ..., \beta_i^{s_1}, ..., \beta_p^{s_1}\}$ y $\beta^{s_2} = \{\beta_1^{s_2}, \beta_2^{s_2}, ..., \beta_i^{s_2}, ..., \beta_p^{s_2}\}$ son creados, siendo $\beta_i^{s_1} = \lambda\beta_i^{f_1} + (1-\lambda)\beta_i^{f_2}$ y $\beta_i^{s_2} = \lambda\beta_i^{f_2} + (1-\lambda)\beta_i^{f_1}$, donde λ es una constante [4]. Es cruce tiende a generar soluciones cerca del centro del dominio. En

nuestros experimentos, siguiendo las recomendaciones de la bibliografía, hemos usado un valor de $\lambda = 0{,}25$.

Cruce BLX-α. Crea un hijo $\beta^s = \{\beta_1^s, \beta_2^s, ..., \beta_i^s, ..., \beta_p^s\}$, donde β_i^s es elegido aleatoriamente en el intervalo $[\beta_{min} - I \cdot \alpha, \beta_{max} + I \cdot \alpha]$. Siendo $c_{max} = max(\beta_i^{f_1}, \beta_i^{f_2})$, $c_{min} = min(\beta_i^{f_1}, \beta_i^{f_2})$ y $I = c_{max} - c_{min}$ [3]. Para $\alpha = 0{,}5$, la probabilidad de que los genes de los hijos tomen valores dentro o fuera del intervalo definido por los padres es la misma. En [4] son probados diferentes valores de α obteniéndose los mejores resultados para $\alpha = 0{,}5$.

Cruce SBX. El cruce SBX (Simulated Binary Crossover) fue propuesto por Deb [2] e intenta emular el efecto del cruce en un punto usado con representación binaria. Este cruce genera dos hijos $\beta^{s_1} = \frac{1}{2}[(1 + B_k)\beta^{f_1} + (1 - B_k)\beta^{f_2}]$ y $\beta^{s_2} = \frac{1}{2}[(1 - B_k)\beta^{f_1} + (1 + B_k)\beta^{f_2}]$, donde $B_k \geq 0$ es un valor generado según la función de densidad

$$p(B) = \begin{cases} \frac{1}{2}(\eta + 1)B^{\eta}, & \text{si } 0 \leq B \leq 1 \\ \frac{1}{2}(\eta + 1)\frac{1}{B^{\eta+2}}, & \text{si } B > 1 \end{cases} \quad (5)$$

Esta distribución puede ser obtenida fácilmente a partir de una distribución uniforme $u(0, 1)$ mediante la transformación

$$B(u) = \begin{cases} (2u)^{\frac{1}{\eta+1}}, & if \eta \leq \frac{1}{2} \\ \left(\frac{1}{2(1-u)}\right)^{\frac{1}{\eta+1}}, & si \eta > \frac{1}{2} \end{cases} \quad (6)$$

Cruce UNDX. El cruce UNDX [7] es un operador en el que la distribución de los hijos generados mantiene el vector media y la matriz de covarianza de la población de padres, a lo que se le da el nombre de *preservación de los estadísticos*. El algoritmo de cruce es el siguiente:

1. Seleccionar aleatoriamente tres padres β^{f_1}, β^{f_2} y β^{f_3} de la población.
2. Calcular el punto medio de los padres β^{f_1} y β^{f_2} como $\beta^p = (\beta^{f_1} + \beta^{f_2})/2$.
3. Calcular el vector diferencia de los padres β^{f_1} y β^{f_2} como $d = \beta^{f_1} - \beta^{f_2}$.
4. Sea D la distancia del tercer padre β^{f_3} a la linea que une β^{f_1} y β^{f_2}. A esta línea se le llama linea de búsqueda primaria.
5. Obtener un descendiente β_s mediante la ecuación $\beta_i^s = \beta^p + \xi d + D\sum_{i=1}^{p-1} \eta_i e_i$, donde $\xi \sim N(0, \sigma_\xi^2)$, $\eta_i \sim N(0, \sigma_\eta^2)$, p es la dimensión del espacio de búsqueda y e_i es el vector base orto-normal del subespacio ortogonal a la linea de búsqueda primaria.

Los valores recomendados para los parámetros por el autor son $\sigma_\xi = 1/2$ y $\sigma_\eta = 0{,}35/\sqrt{n}$.

Cruce CIXL2. Sea $\boldsymbol{\beta}$ un conjunto de N individuos que forman una población y $\boldsymbol{\beta^*} \subset \boldsymbol{\beta}$ el conjunto de los n mejores individuos con respecto a su aptitud. Si consideramos que cada uno de los genes de los individuos de $\boldsymbol{\beta^*}$ están normalmente distribuidos, podemos definir tres individuos asociados al extremo inferior (CILL), superior (CIUL) y media (CIM) del intervalo de confianza de los valores de sus genes:

$$CILL_i = \overline{\beta}_i - t_{n-1,\alpha/2}\frac{S_{\beta_i}}{\sqrt{n}};$$
$$CIUL_i = \overline{\beta}_i + t_{n-1,\alpha/2}\frac{S_{\beta_i}}{\sqrt{n}};$$
$$CIM_i = \overline{\beta}_i$$

siendo $\overline{\beta}_i$, $i = 1, \ldots, n$, la media de cada gen, S_{β_i} la desviación típica, $t_{n-1,\alpha/2}$ un valor obtenido de a partir de la distribución T de Student con $n - 1$ grados de libertad y $1 - \alpha$ la probabilidad de que la media pertenezca a su intervalo de confianza.

Los individuos CILL y CIUL dividen el dominio de cada gen, $D_i \equiv [a_i, b_i]$, en tres subintervalos I^L, I^{CI} y I^R, tal que $D_i \equiv I^L \cup I^{CI} \cup I^R$ y $I^L \equiv [a_i, CILL_i)$; $I^{CI} \equiv [CILL_i, CIUL_i]$; $I^R \equiv (CIUL_i, b_i]$. Este cruce crea a partir de un individuo de la población $\beta^f = (\beta_0^f, \beta_1^f, \ldots, \beta_i^f, \ldots \beta_p^f) \in \boldsymbol{\beta}$, y un individuo CILL, CIUL y CIM, un único hijo β^s de la siguiente forma:

- $\beta_i^f \in I^L$: Si la aptitud de β^f es mayor que la de CILL entonces $\beta_i^s = r(\beta_i^f - CILL_i) + \beta_i^f$, sino $\beta_i^s = r(CILL_i - \beta_i^f) + CILL_i$.
- $\beta_i^f \in I^{CI}$: Si la aptitud de β^f es mayor que la de CIM entonces $\beta_i^s = r(\beta_i^f -$

$CIM_i) + \beta_i^f$, sino $\beta_i^s = r(CIM_i - \beta_i^f) + CIM_i$.

- $\beta_i^f \in I^R$: Si la aptitud de β^f es mayor que la de CIUL entonces $\beta_i^s = r(\beta_i^f - CIUL_i) + \beta_i^f$, sino $\beta_i^s = r(CIUL_i - \beta_i^f) + CIUL_i$ (figure 1).

donde r es un número aleatorio perteneciente al intervalo [0,1]. Ambos parámetros $1-\alpha$ y n, son los encargados de establecer un adecuado equilibrio entre exploración y explotación en función de las características de localización y dispersión de los mejores individuos de la población. En un trabajo anterior [8] hemos encontrado que valores de $(1-\alpha) = 0{,}7$ y $n = 5$ garantizan un equilibrio muy adecuado para problemas de diferente tipología.

4. Experimentos.

Los métodos de penalización son principalmente usados en problemas en los que la no-linealidad tanto de su función de aptitud como de sus restricciones hacen imposible aplicar otros métodos. De entre los numerosos problemas existentes en la bibliografía [1] [6] hemos seleccionado tres cuya formulación y su valores óptimos son mostrados en la tabla 1. f_1 y f_2 solo tienen restricciones no-lineales mientras f_3 tiene también restricciones lineales.

El AGCR usado tendrá una población de 100 individuos, una probabilidad de cruce de $p_c = 0{,}6$, una probabilidad de mutación $p_g = 0{,}05$, y una selección por torneo de dos con elitismo de un individuo. Como operador de mutación usaremos la mutación no uniforme. Para cada problema haremos 30 experimentos. El criterio de parada será un número fijo de generaciones $t_{max} = 5000$.

5. Resultados

En la tabla 2 se muestra por columnas la siguiente información: Af_p, aptitud media penalizada del mejor individuo en 30 experimentos; SDf_p, desviación típica de Af_p; Af, aptitud media sin penalizar del mejor individuo en 30 experimentos; SDf, desviación típica de Af; Bf, aptitud sin penalizar del mejor individuo factible en 30 experimentos; y Gen, generación en la que fue encontrado el mejor individuo.

Para la función f_1 los mejores individuos factibles se obtienen para los cruces BLX y CIXL2. El valor mínimo para f_1 obtenido usando BLX y CIXL2 es $-6961{,}81387$ y difiere del valor óptimo en $5e-5$. Para CIXL2 en tres de los cinco métodos de penalización $Af_p = Af$, es decir, en las 30 ejecuciones realizadas en cada caso el mejor individuo siempre es factible. Para BLX esto solo ocurre usando GENOCOP II. En el segundo problema en todos los casos se llega a la misma solución y siempre los mejores individuos son factibles ($Af_p = Af$).

En la figura 2 se muestra para f_3, por ser esta la función que presenta más dificultad, la evolución usando GENOCOP II de Af. Como podemos ver, la curva descrita por el AGCRs con CIXL2 se mantiene muy estable lo que indica que los mejores individuos de cada generación se encuentran o en la zona factible o en sus proximidades. En los otros casos se producen en mayor o menor medida oscilaciones que hacen converger al algoritmo hacia la zona no factible del dominio. Los resultados de la tabla 2 muestran como con CIXL2 y GENOCOP II se obtienen los mejores resultados. Ningún otro cruce proporciona individuos factibles, a excepción de BLX en combinación con el método de penalización de Joines y Houck, lo que indica lo acertado del uso de la filosofía de cruce del CIXL2 para esta función.

6. Conclusiones

De los resultados anteriores podemos concluir que la filosofía de cruce subyacente en CIXL2 es muy adecuada para la optimización de PPNL usando métodos de penalización, especialmente indicados cuando las restricciones no son lineales.

También podemos ver como la complejidad del problema y el método de penalización usado para modificar la función de aptitud en la zona no factible influye de manera decisiva en el proceso evolutivo. Este hecho abre la puerta a estudios futuros donde se analice en función del tipo de PPNL cual es el método de penal-

Cuadro 1: Problemas de programación no-lineal abordados.

Función	**Óptimo**
$minf_1(\mathbf{x},y) = (x_1-10)^3 + (x_2-20)^3$	$f_1(\mathbf{x}^*)$=-6961.81381
sujeto a:	
$(x_1-5)^2 + (x_2-5)^2 - 100 \geq 0$ $\quad -(x_1-6)^2 - (x_2-5)^2 + 82{,}81 \leq 0$	
$13 \leq x_1 \leq 100$ $\quad 0 \leq x_2 \leq 100$	
$max f_2(\mathbf{x}) = \frac{\sin^3(2\pi x_1)\cdot\sin(2\pi x_2)}{x^3\cdot(x_1+x_2)}$	$f_2(\mathbf{x}^*)$=0.095825
sujeto a:	
$x_1^2 - x_2 + 1 \leq 0$ $\quad 1 - x_1 + (x_2-4)^2 \leq 0$	
$0 \leq x_1 \leq 10$ $\quad 0 \leq x_2 \leq 10$	
$minf_3 = 3x_1 + 0{,}000001x_1^3 + 2x_2 + 0{,}000002/3x_2^3$	$f_3(\mathbf{x}^*)$=5126.4981
sujeto a:	
$1000\sin(x_3 - 0{,}25) + 1000\sin(-x_4 - 0{,}25) + 894{,}8 - x_1 = 0$	
$1000\sin(x_3 - 0{,}25) + 1000\sin(x_3 - x_4 - 0{,}25) + 894{,}8 - x_2 = 0$	
$1000\sin(x_4 - 0{,}25) + 1000\sin(x_4 - x_3 - 0{,}25) + 1294{,}8 = 0$	
$x_4 - x_3 + 0{,}55 \geq 0$ $\quad x_3 - x_4 + 0{,}55 \leq 0$	
$0 \leq x_i \leq 1200, \quad i = 1,2$ $\quad -0{,}55 \leq x_i \leq 0{,}55, \quad i = 3,4$	

ización más adecuado, y en función de estos parámetros seleccionar el AG más adecuado para su resolución, poniendo especial interés en la selección del operador de cruce.

Referencias

[1] C. Coello. A survey of constraint handling techniques used with evolutionary algorithms. Technical report, Lania-RI-9904, Laboratorio Nacional de Informtica Avanzada, 1999.

[2] K. Deb and R. B. Agrawal. Simulated binary crossover for continuous search space. *Complex Systems*, 9:115–148, 1995.

[3] L. J. Eshelman and J. D. Schaffer. Real-coded genetic algorithms and interval-schemata. In L. Darrell Whitley, editor, *Foundation of Genetic Algorithms 2*, pages 187C3.3.7:1–C3.3.7:8.–202, San Mateo, 1993. Morgan Kaufmann.

[4] F. Herrera, M. Lozano, and J. L. Verdegay. Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. *Artificial Inteligence Review*, (12):265–319, 1998. Kluwer Academic Publisher. Printed in Netherlands.

[5] G. E. Liepins and M. D. Vose. Characterizing crossover in genetic algorithms. *Annals of Mathematics and Artificial Intelligence*, (5):27–34, 1992.

[6] Z. Michalewicz and M. Schoenauer. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, 4(1):1–32, 1996.

[7] I. Ono and S. Kobayashi. A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover. In *7th International Conference on Genetic Algorithms*, pages 246–253, Michigan, USA, July 1997. Michigan State University, Morgan Kaufman.

[8] D. Ortiz-Boyer. *Operadores de cruce basados en intervalos de confianza en algoritmos genéticos con codificación real.* PhD thesis, E.T.S.I. Informática, Málaga, Spain, 2001.

Cuadro 2: Resultados.

Cruce	Af_p	SDf_p	Af	SDf	Bf	Gen	Af_p	SDf_p	Af	SDf	Bf	Gen
	f_1						**f_2**					
	Penalización Estática						Penalización Estática					
Arit.	-7902.97440	1.94e+01	-7944.76787	2.13e+01	-6958.75554	1900	0.09583	8.78e-18	0.09583	8.78e-18	0.09583	4912
BLX	-7909.54226	0.00e+00	-7951.97246	3.96e-09	-6878.55428	2986	0.09583	4.39e-18	0.09583	4.39e-18	0.09583	4818
SBX	-7908.31050	2.44e+00	-7950.61485	2.68e+00	-6382.95489	368	0.09583	1.39e-17	0.09583	1.39e-17	0.09583	4985
UNDX	-7909.39841	4.03e-01	-7951.81529	4.44e-01	-6664.46302	3067	0.09583	1.19e-13	0.09583	1.19e-13	0.09583	4982
CIXL2	-7909.54226	0.00e+00	-7951.97246	3.59e-09	-6878.55428	2986	0.09583	1.24e-17	0.09583	1.24e-17	0.09583	4744
	Penalización de Joines y Houck						Penalización de Joines Houck					
Arit.	-6961.27083	5.38e-01	-6961.27083	5.38e-01	-6961.80552	4989	0.09583	9.81e-18	0.09583	9.81e-18	0.09583	4918
BLX	-6961.81362	7.10e-04	-6961.81410	6.90e-04	-6961.81387	4940	0.09583	1.81e-17	0.09583	1.81e-17	0.09583	513
SBX	-6269.64465	3.10e+02	-7528.34282	3.25e+02	-6783.38553	4990	0.09583	1.24e-17	0.09583	1.24e-17	0.09583	4961
UNDX	-6334.12880	3.40e+02	-7474.21272	3.43e+02	-6854.08014	691	0.09583	6.97e-12	0.09583	6.97e-12	0.09583	4945
CIXL2	-6961.81388	0.00e+00	-6961.81388	0.00e+00	-6961.81388	4951	0.09583	2.32e-17	0.09583	2.32e-17	0.09583	1126
	Penalización GENOCOP II						Penalización GENOCOP II					
Arit.	-6697.37820	1.49e+02	-6697.37820	1.49e+02	-6828.22982	4990	0.09583	1.46e-17	0.09583	1.46e-17	0.09583	4881
BLX	-6961.81384	3.00e-05	-6961.81384	3.00e-05	-6961.81387	4999	0.09583	7.60e-18	0.09583	7.60e-18	0.09583	145
SBX	-6533.90823	2.06e+02	-6533.90823	2.06e+02	-6817.03718	4892	0.09583	1.32e-17	0.09583	1.32e-17	0.09583	4907
UNDX	-6574.24726	1.68e+02	-6574.24726	1.68e+02	-6806.01032	4205	0.09583	9.66e-13	0.09583	9.66e-13	0.09583	4979
CIXL2	-6961.81387	1.73e-06	-6961.81387	1.73e-06	-6961.81388	4991	0.09583	1.39e-17	0.09583	1.39e-17	0.09583	433
	Penalización de Smith y Tate						Penalización de Smith y Tate					
Arit.	-6846.96130	8.71e+01	-6846.96130	8.71e+01	-6923.53805	4976	0.09583	1.16e-17	0.09583	1.16e-17	0.09583	286
BLX	-5.510e+10	1.65e+12	-7060.95405	2.97e+02	-6961.81387	4999	0.09583	1.16e-17	0.09583	1.16e-17	0.09583	78
SBX	2.580e+14	5.67e+14	-5844.63957	2.05e+03	-6793.88228	4989	0.09583	1.16e-17	0.09583	1.16e-17	0.09583	2535
UNDX	7.747e+13	2.28e+14	-6351.06428	1.76e+03	-6513.43048	4988	0.09583	9.81e-18	0.09583	9.81e-18	0.09583	286
CIXL2	-1.102e+12	2.20e+12	-7160.09428	3.97e+02	-6961.81387	4991	0.09583	9.81e-18	0.09583	9.81e-18	0.09583	286
	Penalización de Kuri						Penalización de Kuri					
Arit.	5.000e+09	0.00e+00	1.302e+06	9.70e+03	—	—	0.09583	1.46e-17	0.09583	1.46e-17	0.09583	4922
BLX	9.999e+08	2.00e+09	1.612e+06	3.42e+06	-6961.81387	5000	0.09583	1.81e-17	0.09583	1.81e-17	0.09583	3410
SBX	4.500e+09	1.50e+09	3.175e+06	3.49e+06	-6280.89059	4963	0.09583	1.32e-17	0.09583	1.32e-17	0.09583	4957
UNDX	4.500e+09	1.50e+09	5.541e+06	3.82e+06	-6539.62630	4991	0.09583	5.96e-12	0.09583	5.96e-12	0.09583	4978
CIXL2	-6961.81387	4.58e-06	-6961.81387	4.58e-06	-6961.81387	4990	0.09583	1.39e-17	0.09583	1.39e-17	0.09583	4984
	f_3											
	Penalización Estática											
Arit.	5147.62836	2.69e+01	5147.45233	2.69e+01	—	—						
BLX	5291.59506	1.41e+02	5291.40869	1.41e+02	—	—						
SBX	5312.11021	3.01e+02	5311.37972	3.01e+02	—	—						
UNDX	9456.21763	9.57e+03	5350.96311	2.33e+02	—	—						
CIXL2	5126.33324	1.49e-06	5126.16839	1.45e-06	—	—						
	Penalización de Joines y Houck											
Arit.	5145.88820	1.80e+01	5143.86759	1.87e+01	—	—						
BLX	5200.40070	1.16e+02	5193.64613	1.10e+02	5146.48500	4997						
SBX	8638.78859	5.09e+03	5216.65409	3.51e+01	—	—						
UNDX	20411.34313	2.53e+04	5311.12395	2.41e+02	—	—						
CIXL2	5245.58174	7.27e+01	5211.02190	4.44e+01	5142.04938	4541						
	Penalización GENOCOP II											
Arit.	4.618e+15	5.77e+15	5155.68967	3.21e+01	—	—						
BLX	2.086e+07	6.09e+07	5278.07682	1.24e+02	—	—						
SBX	1.300e+20	3.80e+20	5216.38843	3.10e+01	—	—						
UNDX	9.616e+24	1.52e+24	5282.72195	1.04e+02	—	—						
CIXL2	5126.56719	1.11e-01	5126.52397	2.61e-02	5126.51365	4610						
	Penalización de Smith y Tate											
Arit.	5149.59471	2.31e+01	5149.58658	2.31e+01	—	—						
BLX	5400.56093	3.49e+02	5387.79282	3.53e+02	—	—						
SBX	1.484e+04	1.89e+04	5339.60746	2.92e+02	—	—						
UNDX	7.618e+05	1.59e+06	5488.69093	3.12e+02	—	—						
CIXL2	5175.73846	9.87e+01	5166.14501	8.41e+01	—	—						
	Penalización de Kuri											
Arit.	6.000e+08	0.00e+00	3349.55454	7.90e+01	—	—						
BLX	6.000e+08	0.00e+00	3417.87828	2.12e+03	—	—						
SBX	6.000e+08	0.00e+00	3482.11846	2.33e+03	—	—						
UNDX	6.000e+08	0.00e+00	7742.64803	1.10e+03	—	—						
CIXL2	6.000e+08	0.00e+00	3374.40011	3.32e+03	—	—						

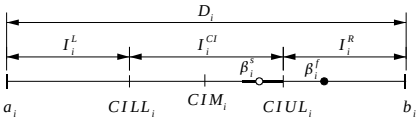

Fig. 1. CIXL2

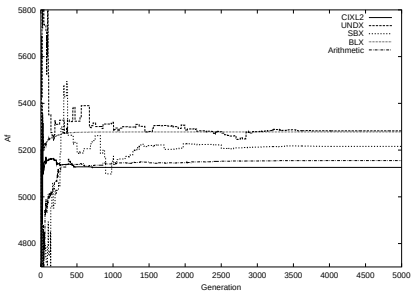

Fig. 2. Representación gráfica de Af para f_3