Optimización global

Raúl Medrano Millán[1]*

Resumen
El cálculo del valor mínimo o máximo global de una función puede hallarse de diferentes maneras. Por una parte, es posible hacer un estudio de la función para obtener este dato mediante métodos matemáticos. Otra forma de hallar este valor es mediante los diferentes algoritmos desarrollados con este propósito. En este trabajo se van a presentar, tanto el estudio de una función como algunos de los diferentes tipos de algoritmos existentes.

Palabras clave
optimización global; mínimo; algoritmo cúbico; algoritmo de evolución diferencial

[1] *Facultad de Ciencias, UNED, Madrid*
*rmedrano1@alumno.uned.es

Índice

Introducción

El objetivo de la optimización global es encontrar el valor mínimo, o el valor máximo, absoluto (global) de una función real de variable real o vectorial real en todo su dominio, así como el conjunto donde se alcanza ese valor (minimizadores). En general hablaremos del cálculo del mínimo de una función, puesto que el máximo puede obtenerse como el mínimo de la misma función cambiada de signo.

La optimización global es importante porque se aplica en diversas áreas, como por ejemplo:

- Ingeniería: Diseño óptimo de estructuras, control de procesos. Ejemplo: diseño de una antena con el objetivo de minimizar la relación señal-ruido, y por tanto la calidad de la señal emitida.
- Economía: Modelos de optimización de recursos. Ejemplo: diseño de carteras de inversión, maximizando la rentabilidad y minimizando los riesgos, con las restricciones que imponga del inversor.
- Ciencia: Resolución de problemas de optimización en campos como la física, la química y la biología. Ejemplo: búsqueda de fármacos cuyas moléculas minimicen la energía libre entre molécula y el objetivo.

Cuando se trata de calcular el mínimo global de una función continua y derivable (al menos a trozos), podemos aplicar un estudio matemático a dicha función y obtener tanto los minimizadores como el valor mínimo global de la función, e incluso analizar otros aspectos de la función como mínimos y máximos locales, zonas de crecimiento y decrecimiento de la función, etc.

Podemos tomar la definición de mínimo absoluto de una función de Marsden [1].

Definición: *sea* $f : A \to \mathbb{R}$ *una función definida en un conjunto* A *de* $\mathbb{R}^2$ *o* $\mathbb{R}^3$. *Se dice que un punto* $\vec{x}_0 \in A$ *es un punto de mínimo absoluto de* f *si* $f(\vec{x}) \geq f(\vec{x}_0)$ *para todo* $\vec{x} \in A$.

En el caso de funciones de una variable, la definición se reduce a:

Definición: *Sea $f : A \to \mathbb{R}$ una función definida en un conjunto A de $\mathbb{R}$. Se dice que un punto $x_0 \in A$ es un punto de mínimo absoluto de f si $f(x) \geq f(x_0)$ para todo $x \in A$.*

Cuando la búsqueda de mínimo se restringe a un conjunto cerrado y acotado, sabemos por el teorema de existencia de máximos y mínimos globales [1], que este mínimo existe.

Hay otros casos donde no podemos obtener ese mínimo global mediante el estudio de la función. E incluso hay casos prácticos donde el objetivo puede no tratarse de una función, sino de meras expresiones matemáticas. Es por esto que en optimización global se han ido elaborando diferentes algoritmos para cubrir todo tipo de posibles problemas.

Por tanto, hay varios posibles modos de obtención del mínimo de una función y una primera clasificación

- Mediante el estudio de la función.
- Mediante el uso de algoritmos de optimización, que según su naturaleza pueden ser:
  - Determinísticos: Utilizan un proceso definido para encontrar el mínimo, siempre con el mismo resultado para las mismas condiciones iniciales. No incluyen ningún tipo de aleatoriedad.
  - Probabilísticos: Incorporan aleatoriedad en los datos del problema (la función objetivo, las restricciones, etc) o en el propio algoritmo, lo que puede generar resultados diferentes en cada ejecución. Se pueden distinguir las siguientes clases:
    - Búsqueda aleatoria global (GRS), que implica decisiones aleatorias en el proceso de elección de los puntos de observación.
    - Aleatoriedad sobre la función objetivo, donde los supuestos estocásticos sobre la función objetivo se utilizan de manera similar a cómo se utiliza la condición de Lipschitz en algoritmos deterministas.
    - Heurístico o metaheurísticos, que en términos generales, significa 'encontrar' o 'descubrir mediante prueba y error'. Algunos de estos algoritmos están basados en analogías con procesos naturales, como la evolución o la física.

Los algoritmos actuales (o sus más recientes versiones) pueden ser híbridos, es decir comparten aspectos de las diferentes categorías expuestas en la clasificación anterior.

En las siguientes secciones se pretende dar una visión de tres tipos generales de solución al momento de abordar el problema de minimización de una función real: estudio analítico de la función, un ejemplo de algoritmo determinístico (algoritmo cúbico) y un ejemplo de algoritmo probabilístico (algoritmo de evolución diferencial). Y para ello vamos a utilizar como ejemplo la función real de dos variables $f(x,y) = 1 + |(x^2+y^2)^3 - 3x^2 - 3y^2|$.

1. Estudio de la función

Para encontrar los mínimos globales de la función $f(x,y) = 1 + |(x^2+y^2)^3 - 3x^2 - 3y^2|$ comenzamos su estudio, y para ello empleremos diferentes técnicas:

- Diferenciación: Cálculo de las derivadas para identificar puntos críticos.
- Análisis de intervalos: Estudio del comportamiento de la función en diferentes intervalos del dominio.
- Simetría: Aprovechamiento de la simetría de las funciones para simplificar el análisis.
- Graficación: Representación gráfica de las funciones para visualizar los mínimos.

Además de todo el dominio en el que está definida la función ($\mathbb{R}^{\not\in}$), también podemos restringir la búsqueda del mínimo a un conjunto acotado y cerrado de su dominio. En este último caso podemos seguir la siguiente estrategia, tal como se indica en [1]:

Sea f una función continua de dos variables definida en una región D en $\mathbb{R}^2$ cerrada y acotada, que está limitada por una curva cerrada suave. Para hallar el mínimo absoluto de f en D:

1. *Localizar todos los puntos críticos de f en D.*
2. *Hallar los puntos críticos de f considerada como una función definida sólo en δD.*
3. *Calcular el valor de f en todos esos puntos críticos.*
4. *Comparar todos esos valores y seleccionar el menor.*

Si D es una región limitada por un conjunto de curvas suaves (como un cuadrado), el procedimiento es el indicado anteriormente, añadiendo en el tercer paso los puntos donde se unen las curvas (en el caso del cuadrado, las esquinas).

[1] *Sea D cerrado y acotado en $\mathbb{R}^n$, y sea $f : D \to \mathbb{R}$ continua. Entonces existen puntos x_0 y x_1 de D donde f alcanza sus valores máximo y mínimo.*[1]

1.1 Estudio en todo su dominio

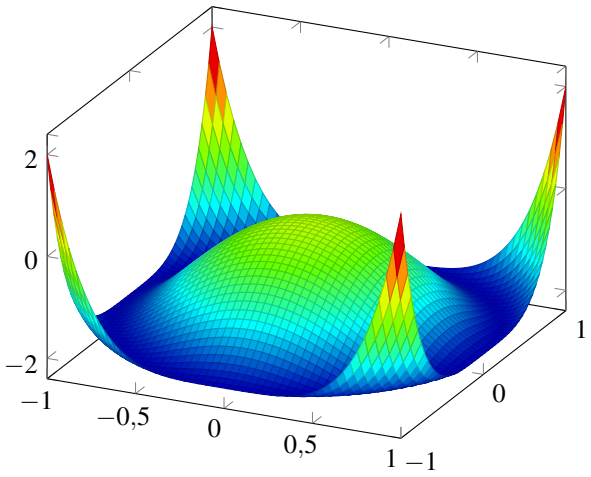
Podemos apreciar que lLa función $f(x,y)$ está formada por la suma de otras dos funciones: una función constante $g(x,y) = 1$ y otra función valor absoluto de $h(x,y) = (x^2+y^2)^3 - 3x^2 - 3y^2$. Como $g(x,y)$ y $h(x,y)$ son continuas la función $f(x,y)$ también es continua.

El mínimo (y también el máximo) de función $g(x,y)$ es 1, y se alcanza en todo su dominio, es decir en $\mathbb{R}^2$. La función $|h(x,y)|$, al tratarse de un valor absoluto, siempre tendrá un valor ≥ 0.

La estrategia a seguir para obtener el valor o valores mínimos globales de la función, será la siguiente:

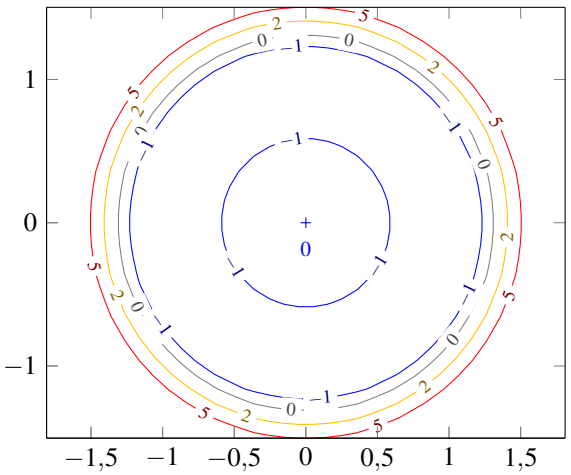
- Obtener los puntos en que la función $h(x,y) = (x^2+y^2)^3 - 3x^2 - 3y^2$ toma el valor 0. Si existen, estos serán los valores mínimos puesto que $|h(x,y)|$ no puede alcanzar valores menores, y por tanto serán los valores mínimos para $f(x,y)$.
- Si no existe ningún punto que cumpla la condición anterior, obtener los puntos donde $h(x,y)$ tenga el valor mínimo cuando $h(x,y) > 0$ y los valores máximos cuando $h(x,y) < 0$. Se trata de calcular los máximos y mínimos locales de $h(x,y)$. Del conjunto de todos estos valores, seleccionar los que hagan que $|h(x,y)|$ sea mínimo.

La gráfica de $h(x,y) = (x^2+y^2)^3 - 3x^2 - 3y^2$ en $[-1,1] \times [-1,1]$es la siguiente:



Para obtener los puntos en que la función $h(x,y) = (x^2+y^2)^3 - 3x^2 - 3y^2$ toma un determinado valor, calculamos las curvas de nivel [2] correspondientes a dicho valor.

Curvas de nivel



Para calcular analíticamente los puntos donde las curvas de nivel de la función $h(x,y)$ tiene valor 0, nos fijamos el la simetría de las variables x e y y en que sus valores se toman elevados al cuadrado. Esto sugiere un cambio de variable a coordenadas polares

$$x = rcos\theta,\ y = rsen\theta,\ r \geq 0,\ 0 \leq \theta < 2\pi$$

[2] Sea $f : U \subset \mathbb{R}^n \to \mathbb{R}$ y sea $c \in \mathbb{R}$. Entonces, el conjunto de nivel de valor c se define como el conjunto de los puntos $\vec{x} \in U$ en los cuales $f(\vec{x}) = c$. Si $n = 2$, hablaremos de curva de nivel (de valor c); y si $n = 3$, hablaremos de superficie de nivel. Con símbolos, el conjunto de nivel de valor c se escribe: $\{\vec{x} \in U | f(\vec{x}) = c\} \subset \mathbb{R}^n$. Nótese que el conjunto de nivel siempre está en el dominio de la función.[1]

de modo que

$$h(x,y) = (x^2+y^2)^3 - 3x^2 - 3y^2 \Rightarrow$$
$$\Rightarrow h(r,\theta) = r^6 - 3r^2 = r^2(r^4-3)$$

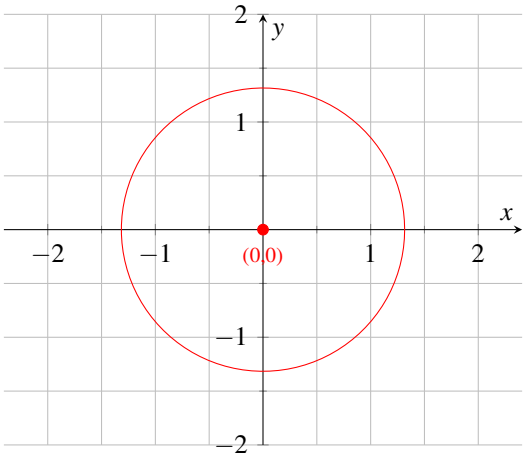
que toma el valor 0 cuando $r = 0$ y $r^4 = 3$, es decir, cuando $r = \{0, \sqrt[4]{3}\}$, ya que $r \geq 0$.

Si volvemos a coordenadas cartesianas, el valor $r = 0$ se corresponde con el valor $(0,0)$ y el valor $r = \sqrt[4]{3}$ se corresponde con la región $x^2+y^2 = \sqrt{3}$, que es la circunferencia centrada en $(0,0)$ y con radio $\sqrt[4]{3} \approx 1{,}316074$ (ya que no depende de θ).

Por tanto, los mínimos globales se alcanzan en:

$$\left.\begin{array}{l}\vec{x} = (0,0) \\ \vec{x} \in \{x,y \,|\, x^2+y^2 = \sqrt{3}\}\end{array}\right\} \Rightarrow f(x,y) = 1 \tag{1}$$

que, representados en un gráfico:



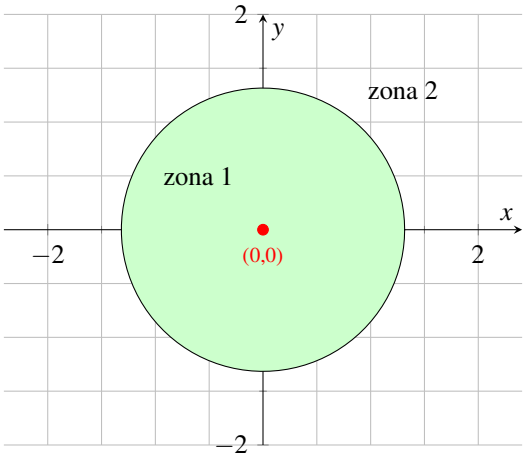
1.2 Estudio de la función en una caja cerrada

Estudiaremos los mínimos de la función restringidos a una caja cerrada con bordes paralelos a los ejes cartesianos, que definiremos como el conjunto de puntos que cumplen que $(x,y) \in [a,b] \times [c,d]$, con $a,b,c,d \in \mathbb{R}$, $a < b$ y $c < d$.

Del estudio anterior sabemos que $f(x,y)$ es mayor que 1 en todos los puntos de su dominio ($\mathbb{R}^2$), excepto en los puntos donde se alcanza su valor mínimo (circunferencia de radio $\sqrt[4]{3}$, más su centro $(0,0)$), que llamaremos minimizadores y donde toma valor 1.

Si la intersección de la caja cerrada con el conjunto de minimizadores no es vacía, entonces, por definición de mínimo absoluto, los minimizadores de de la caja cerrada se corresponderán con dicha intersección y tomarán valor 1.

Para el estudio de una caja cerrada que no incluya ninguno de los puntos de mínimo global, podemos analizarlo en las dos regiones en las que puede estar dicha caja. Por una parte está el círculo perforado en su centro con radio $0 < r < \sqrt[4]{3}$ y por otra la región externa a ese círculo, es decir cuando $r > \sqrt[4]{3}$.



También es importante constatar que las curvas de nivel se producen en circunferencias concéntricas con centro en el origen, tal como se estudió anteriormente. Por tanto, la simetría de esta función permite afirmar que todos los puntos con igual distancia al origen de coordenadas (igual radio) tienen el mismo valor.

La función $f(x,y)$ expresada en coordenadas polares es $f(r,\theta) = 1 + |r^6 - 3r^2|$. En la zona 1 podemos escribirla como $f(r,\theta) = f(r) = 1 - r^6 + 3r^2$ (ya que sólo depende de r), cuya derivada respecto a r es $f'(r) = -6r^5 + 12r$, que igualando a 0 nos dará los puntos críticos locales (descartando $r = 0$ ya que no está incluido en esta zona 1), $r = \sqrt[4]{2}$. Estos puntos son máximos ya que $f''(r) = -30r^4 + 12 \Rightarrow f''(\sqrt[4]{2} = -60 + 12 = -48 < 0$. Por tanto cuando $0 < r < \sqrt[4]{2}$, $f(x)$ es creciente, mientras que cuando $\sqrt[4]{2} < r < \sqrt[4]{3}$, la función es decreciente.

En la zona 2 podemos escribirla como $f(r,\theta) = f(r) = 1 + r^6 - 3r^2$ (ya que sólo depende de r), cuya derivada respecto a r es $f'(r) = 6r^5 - 12r$, que, para $r > \sqrt[4]{3}$ es mayor que cero y por tanto es creciente.

Para los dos posibles casos según que la caja está en la primera o la segunda zona definidas anteriormente.

- caja completamente incluida dentro de zona 1. En esta zona la función primero crece y luego decrece según la distancia al origen (radio). Por tanto el mínimo se alcanzará en el punto más cercano al origen y/o en el punto más lejano al origen. Si definimos por $r_i, i = 1,..,4$ la distancia de cada vértice al origen, el mínimo corresponderá al punto (x_i, y_i) cuyo valor $f(x_i, y_i) = f(r_i)$ sea menor.
- caja completamente incluida en la zona 2. Como fuera de la circunferencia $x^2 + y^2 = \sqrt{3}$ la función $f(x,y)$ es creciente, el mínimo estará en el punto más cercano a dicha circunferencia, es decir, el punto (x_i, y_i) cuyo valor $r_i = \sqrt{x_i^2 + y_i^2}$ sea menor y tendrá valor $f(r_i) = 1 + r_i^6 - 3r_i^2$.

2. Algoritmo cúbico (Cubic Algorithm)

2.1 Descripción del algoritmo

Uno de los métodos utilizados en la consecución de mínimos globales es el algoritmo Cubic, que fue presentado inicialmente por Efim A. Galperin en 1985 [2] y perfeccionado por ese mimo autor y Miguel Delgado Pineda [3], entre otros, en los años siguientes.

El algoritmo Cubic se parece a los métodos tradicionales de bisección o subdivisión de segmentos y lleva a cabo la enumeración de conjuntos globales con eliminaciones, de manera que nos permite identificar y eliminar aquellos conjuntos que no contienen al mínimo global.

Partimos del problema de encontrar el mínimo global de una función $f : \mathbb{R}^n \to \mathbb{R}$, Lipschitziana, en un cubo (n dimensional) cerrado $\overline{C} \subset \mathbb{R}^n$, de lado $c > 0$, tal que

$$\mathbf{s}^0 = min\, f(\mathbf{x}) \leq f(\mathbf{x})\ \forall \mathbf{x} \in \overline{C}$$

y el conjunto donde la función toma este valor

$$\overline{C}^0 = \{\mathbf{x} \,|\, f(\mathbf{x}) = \mathbf{s}^0, \mathbf{x} \in \overline{C}\}$$

Por ser Lipschitziana, la función $f(\mathbf{x})$ cumple la condición

$$|f(\mathbf{x}) - f(\mathbf{x}')| \leq L \parallel \mathbf{x} - \mathbf{x}' \parallel, L = \text{constante} > 0, \forall \mathbf{x}, \mathbf{x}' \in \overline{C}$$

y si se conoce la constante de Lipschitz sobre $\overline{C}$

$$L = max \parallel \nabla f(\mathbf{x}) \parallel, \mathbf{x} \in \overline{C}$$

El algoritmo Cubic es un algoritmo iterativo que en cada iteración se aproxima al mínimo buscado. Definamos como ε el error máximo que admitimos para acercarnos al valor mínimo. Se presentan los siguientes pasos.

Paso 1. Se define una partición de $\overline{C}$ como el conjunto de elementos disjuntos $\overline{C}_i \subset \overline{C}, \overline{C}_i \cap \overline{C}_j = \varnothing, i \neq j$, tal que $\cup \overline{C}_i = \overline{C}$. Si tomamos un número $N \geq 2$ y dividimos cada arista del cubo $\overline{C}$ en N partes, obtenemos una partición del cubo en N^n cubos de tamaño idéntico con longitud de cada arista igual a c/N. Al conjunto de los elementos de esta partición lo denominamos $\overline{C}^1$.

Paso 2. En uno de los elementos de la partición anterior tomamos un punto $\mathbf{x}_0^1$. Sea $\overline{C}_0^1$ el elemento de la partición al que pertenece $\mathbf{x}_0^1$. Aplicando traslaciones de longitud c/N en las n dimensiones de $\overline{C}$, obtendremos un conjunto de N^n puntos $\mathbf{x}_i^1$ donde cada uno de ellos pertenece a un $\overline{C}_i^1$ diferente. De este modo tenemos un conjunto de puntos representativo de la partición $\overline{C}^1$. Tomaremos, como punto correspondiente a cada uno de los elementos de la partición, el vértice con el menor valor de todas y cada una de sus coordenadas.

Paso 3. Para cada uno de los puntos anteriores calculamos $f(\mathbf{x}_i^1)$ y nos quedamos con el menor de estos valores, $\mathbf{s}_1 = min\, f(\mathbf{x}_i^1)$.

Paso 4. Se calcula la diagonal de los elementos de la partición $d_1 = \frac{c\sqrt{n}}{N}$ y se multiplica por L, la máxima pendiente de la función a estudiar, de modo que obtenemos la máxima variación que pueden tener los elementos de la partición, $r_1 = \frac{Lc\sqrt{n}}{N}$. Se descartan los elementos de la partición $\overline{C}^1 = \{\overline{C}_i^1\}$ tales que $f(\mathbf{x}_i^1) - s_1 > r_1$, es decir, aquellos elementos donde, a ciencia cierta, no puede estar el mínimo global de la función. Por tanto, nos quedaremos con un subconjunto de elementos de la partición $\overline{C}^1$.

Paso 5. Si $r_1 \leq \varepsilon$ entonces se ha alcanzado el mínimo con la precisión deseada. En caso contrario, al conjunto de elementos restantes, se les aplica una nueva división de sus aristas por N, de modo que obtenemos una nueva partición $\overline{C}^2$, con elementos cuya arista será c/N^2. A esta nueva partición le aplicamos los pasos 2 a 5 sustituyendo el superíndice 1 por el superíndice 2. De nuevo, al llegar a esta paso 5, comprobamos si $r_2 \leq \varepsilon$ para saber si hemos obtenido el resultado con la precisión deseada o, de nuevo, procedemos a una nueva iteración.

En cada iteración obtenemos un valor mínimo $\mathbf{s}_i$ para el conjunto de valores correspondiente a la partición $\overline{C}^i$, donde cada elemento tiene una diagonal $d_i = \frac{c\sqrt{n}}{N^i}$ y una variación máxima $r_i = \frac{Lc\sqrt{n}}{N^i}$. Estos elementos cumplen las condiciones siguientes:

$$s_1 \geq s_2 \geq s_3 \geq \ldots \geq s_m \geq \ldots$$

$$\overline{C} \supseteq \overline{C}^1 \supseteq \overline{C}^2 \supseteq \ldots \supseteq \overline{C}^m \supseteq \ldots$$

En [2] se define y demuestra el teorema de convergencia de los valores anteriores a la solución buscada:

$$\lim_{m\to\infty} s_m = s^* = \min_{\mathbf{x}\in\overline{C}} f(\mathbf{x})$$

$$\lim_{m\to\infty} \overline{C}^m = \cap\overline{C}^m = C^*$$

donde s^* es el mínimo global buscado y C^* es el conjunto donde se alcanza ese mínimo.

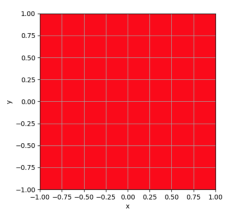
Los parámetros que controlan el algoritmo son:

- el valor de la constante de Lipschitz (L),
- el número de particiones a realizar en cada iteración (N),
- y el valor de la tolerancia máxima (ε).

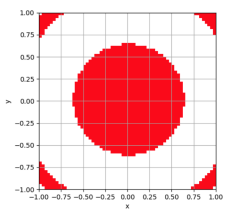
La terminación del algoritmo se produce cuando la variación en el valor de cubo es menor que el valor de la tolerancia ε.

2.2 Resultados obtenidos para la función test

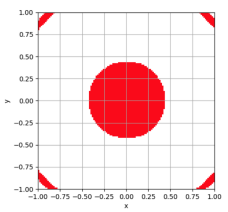
Se ha realizado una implementación del algoritmo en Python, en la que es posible modificar todos los parámetros del algoritmo, anteriormente descritos, en la llamada a la función. El las figuras siguientes se presentan los resultados obtenidos para la función $f(x,y) = 1 + |(x^2+y^2)^3 - 3x^2 - 3y^2|$ en el dominio definido por la caja $[-1,1]\times[-1,1]$, con $N = 2$ y mínimo valor de $\varepsilon = 0{,}01$ para diferentes iteraciones, con indicación del número de cubos que forman los minimizadores, el tamaño del lado de cada cubo, el error máximo en el valor del mínimo obtenido y el número de veces que se ha evaluado la función.



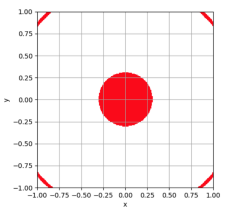
Iteración = 5
Nº de cubos = 1024
Lado del cubo = 0,0625
ε = 2,25
Nº evaluaciones = 1023



Iteración = 6
Nº de cubos = 1433
Lado del cubo = 0.03125
ε = 1,125 ε = 0,5625
Nº evaluaciones = 4095

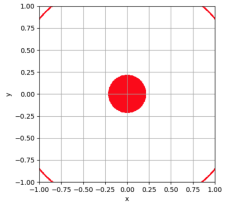


Iteración = 7
Nº de cubos = 2693
Lado del cubo = 0,015625
ε = 0,28125
Nº evaluaciones = 8394

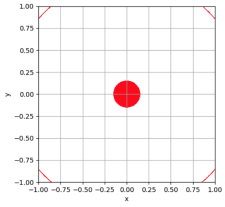


Iteración = 8
Nº de cubos = 5353
Lado del cubo = 0,0078125

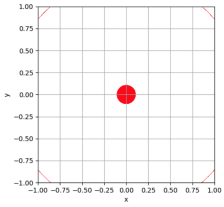
Nº evaluaciones = 16473



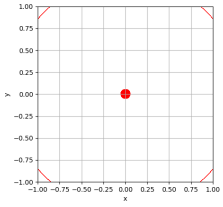
Iteración = 9
Nº de cubos = 10617
Lado del cubo = 0,00390625
ε = 0,140625
Nº evaluaciones = 32532



Iteración = 10
Nº de cubos = 21217
Lado del cubo = 0,001953125
ε = 0,0703125
Nº evaluaciones = 64383



Iteración = 11
Nº de cubos = 42393
Lado del cubo = 0,0009765
ε = 0,03515625
Nº evaluaciones = 128034



Iteración = 13
Nº de cubos = 169629
Lado del cubo = 0,00024414
ε = 0,008789
Nº evaluaciones = 509448

Para una precisión de al menos $\varepsilon = 0,001$ se obtienen, después de 17 iteraciones, 2715261 cubos finales con un tamaño de cada lado de 0,0000153.

3. Algoritmo de evolución diferencial (Differential Evolution Algorithm)

3.1 Descripción del algoritmo

El algoritmo Evolución Diferencial (Differential Evolution Algorithm) fue desarrollado por R. Storn y K. Price en 1996 [4]. Se trata de un algoritmo genético que no utiliza derivadas de la función objetivo, lo que le permite ser aplicado a funciones no lineales y no diferenciables.

El algoritmo simula el comportamiento de una población (de tamaño determinado n), que evoluciona generación tras generación hacia el mínimo de la función objetivo. Cada elemento de esta población (cromosoma o genoma en terminología de los algoritmos genéticos) es representado en cada generación t, mediante un vector de dimensión d correspondiente al espacio a estudiar:

$$\mathbf{x}_i^t = (x_{1,i}^t, x_{2,i}^t, ..., x_{d,i}^t)$$

La población inicial debe cubrir de forma uniforme y aleatoria todo el espacio de búsqueda definido por las restricciones del problema. Por lo tanto partimos de n vectores:

$$x_{j,i}^0 = x_{j,min} + rand_{i,j}[0,1] \cdot (x_{j,max} - x_{j,min}),\ i = 1,...,n;\ j = 1,...,d$$

donde $rand_{i,j}[0,1]$ es un número aleatorio uniformemente distribuido en el intervalo $[0,1]$ e independiente para cada componente del vector y donde $x_{j,min}, x_{j,max}$ son los valores mínimos y máximos de cada componente de las restricciones del problema.

El algoritmo evoluciona de una generación a otra mediante la aplicación de los pasos siguientes:

1. Inicialización de la población: Se crea un conjunto de vectores aleatorios (individuos) uniformemente distribuidos que representan posibles soluciones.
2. Mutación: Se modifica cada individuo mediante una operación de mutación que utiliza información de otros individuos.
3. Recombinación: Se combinan las características de los individuos mutados para crear nuevos individuos.
4. Selección: Se elige el mejor individuo de cada par, padre e hijo, para formar la siguiente generación.
5. Repetición: Se repiten los pasos 2-4 hasta que se alcanza un criterio de parada, como un número máximo de iteraciones o un mínimo global estable.

Tal como ocurre en la naturaleza, la mutación supone un cambio repentino en las características genéticas de un cromosoma. En el algoritmo, este cambio se simula mediante el esquema de mutación siguiente:

$$\mathbf{v}_i^{t+1} = \mathbf{x}_{r1}^t + F\,(\mathbf{x}_{r2}^t - \mathbf{x}_{r3}^t) \tag{2}$$

donde $r_1, r_2, r_3 \in \{1,2,...,n\}$, $r_1 \neq r_2 \neq r_3 \neq i$ y donde $F \in [0,2]$ es un parámetro, denominado peso diferencial, que controla la amplificación de la variación diferencial $(x_{r2}^t - x_{r3}^t)$. Por tanto, para que se produzca mutación, la población debe ser mayor o igual que 4 $(n \geq 4)$. En la práctica $F \in [0,1]$ es más eficiente y estable [5].

La recombinación se produce cuando hay un intercambio de partes de dos cromosomas entre sí y permite la mejora en la diversidad de la población. Esta recombinación se puede conseguir mediante dos métodos: binomial y exponencial. El

primero se realiza en cada una de las d variables siempre que un número generado aleatoriamente entre 0 y 1 sea menor o igual que el valor de Cr (probabilidad de recombinación) de modo que el número de componentes heredadas del donante tiene una distribución casi binomial. El esquema puede resumirse como:

$$u_{j,i}^{t+1} = \begin{cases} v_{j,i}^{t+1} & \text{if } r_i \leq Cr \text{ or } j = J_r \\ x_{j,i}^{t} & \text{if } r_i \geq Cr \text{ and } j \neq J_r \end{cases}, \; j = 1,2,...,d$$

donde $r_i \in [0,1]$ es un número aleatorio uniformemente distribuido y $J_r \in \{1,2,...,d\}$ es un índice generado aleatoriamente.

En el método exponencial, se selecciona un segmento del vector donante que comience en un punto aleatorio k y que tenga una longitud L aleatoria, por lo que puede contener más de un componente. El esquema matemático sería el siguiente:

$$u_{j,i}^{t+1} = \begin{cases} v_{j,i}^{t} & \text{para } j = k,...,k-L+1 \in [1,d] \\ x_{j,i}^{t} & \text{en otro caso} \end{cases}$$

La última etapa, la selección, consiste en calcular el valor de la función a analizar con el nuevo vector generado en las etapas anteriores y compararlo con el valor que se tenía con el vector original. Si es menor, sustituimos en vector original por el nuevo. Esquemáticamente:

$$\mathbf{x}_i^{t+1} = \begin{cases} \mathbf{u}_i^{t+1} & \text{si } f(\mathbf{u}_i^{t+1}) \leq f(\mathbf{x}_i^t) \\ \mathbf{x}_i^t & \text{en otro caso} \end{cases}$$

La eficiencia de todo el proceso está controlada por los parámetros F y C_r. También es posible modificar la ecuación (2) para obtener otro método de mutación denominado *best* (el método anterior se denomina *rand*), donde se utiliza el vector x_{best}^t en lugar del vector X_{r1}, es decir:

$$\mathbf{v}_i^{t+1} = \mathbf{x}_{best}^t + F\,(\mathbf{x}_{r2}^t - \mathbf{x}_{r3}^t)$$

donde $\mathbf{x}_{best}^t$ es el vector con el menor valor $f(\mathbf{x}_i^t)$.

También sería posible utilizar más de dos vectores para el cálculo de la diferencia en la mutación. Por ejemplo, si se utilizan en lugar de 2, cuatro vectores, los esquemas para el método *rand* y *best* serían:

$$\mathbf{v}_i^{t+1} = \mathbf{x}_{r1}^t + F\,(\mathbf{x}_{r2}^t - \mathbf{x}_{r3}^t + \mathbf{x}_{r4}^t - \mathbf{x}_{r5}^t)$$
$$\mathbf{v}_i^{t+1} = \mathbf{x}_{best}^t + F\,(\mathbf{x}_{r2}^t - \mathbf{x}_{r3}^t + \mathbf{x}_{r4}^t - \mathbf{x}_{r5}^t)$$

Como forma de clasificar las diferentes variantes del algoritmo, se utiliza la notación $DE/x/y/z$, donde

- x indica si en la mutación se utiliza el método de mutación *rand* o *best*
- y es el número de vectores diferencia utilizados
- z es el método de recombinación utilizado, es decir, *bin* o *exp*

La condición para la terminación del algoritmo puede ser definida de diferentes maneras. Algunas de ellas pueden ser: [6]

- determinar un máximo número de iteraciones dependiendo de la complejidad de la función a analizar,[3]
- cuando el mínimo alcanzado por la población no varía apreciablemente de una a otra iteración,
- se fija *a priori* un determinado objetivo a alcanzar.

3.2 Resultados obtenidos para la función test

Al igual que para el algoritmo cúbico, también se ha desarrollado una implementación propia del algoritmo en Python, correspondiente a la variante $DE/rand/1/bin$ con valores iniciales por defecto $n = 30, F = 0{,}5$ y $Cr = 0{,}7$. Como condición de terminación se ha incluido tanto la limitación del número de iteraciones (1000) y que la desviación estándar del conjunto de mínimos encontrados sea menor que una cantidad (10^{-9}) multiplicada por la media de dichos valores. Lo que primero ocurra marcará la parada del algoritmo. Todos los valores pueden ser modificados como parámetros de entrada en la llamada a la función del algoritmo.

Aplicado a la función $f(x,y) = 1 + |(x^2+y^2)^3 - 3x^2 - 3y^2|$ en el dominio definido por la caja $[-1,1] \times [-1,1]$, se obtiene el mínimo en 1,0, después de 32 iteraciones y con 1020 evaluaciones de la función.

[3] Esta es la opción definida en el trabajo inicial de Storn y Price. [4]

4. Conclusiones

A lo largo de este trabajo se han presentado diferentes enfoques para la resolución del problema de encontrar el mínimo global de una función, desde el estudio de la propia función hasta la aplicación de algoritmos tanto de tipo determinístico como de tipo probabilístico.

Posiblemente la primera conclusión a la que se llega es que, ante un problema de optimización global, bien vale la pena dedicar un tiempo de estudio para poder valorar la mejor opción en la consecución del mejor resultado en el menor tiempo posible.

El estudio analítico de la función objetivo nos aportará la mejor solución, que será exacta y obtendremos respuestas adicionales al objetivo principal: posibles mínimos locales, definición completa de su dominio, etc. En contra tenemos que son pocas las funciones a las que podemos aplicar un estudio analítico completo. La función estudiada en el presente trabajo presenta, además, simetrías que nos facilitan su estudio. Igualmente, cuando se trata de funciones de una y dos dimensiones, el uso de gráficos facilita su visualización y, por ende, su estudio. Para mayores dimensiones, no sería posible este estudio gráfico.

Como ejemplo de algoritmo determinístico se ha utilizado el Algoritmo Cúbico (Cubic Algorithm). Este algoritmo presenta varias ventajas: es aplicable a funciones de n dimensiones, las funciones no necesitan ser diferenciables sobre su dominio de definición (incluso puede tratarse de expresiones computacionales), es fácilmente implementable, permite controlar el error con el que conseguir los resultados, y obtenemos la región del dominio donde se alcanza el valor mínimo (minimizadores). Por otra parte, se demuestra la convergencia al valor mínimo. Entre algunas de las limitaciones de este algoritmo se encuentran que las funciones deban ser Lipschitzianas, que la reducción del error en la solución o el aumento de las dimensiones de la función implican un aumento exponencial en el número de cubos a definir y por tanto una ralentización del tiempo de respuesta conforme aumenta en número de iteraciones, y que el algoritmo se complica cuando el dominio de búsqueda no es una caja con lados paralelos a los ejes coordenados.[4]. Existen versiones evolucionadas del algoritmo donde se mitigan algunos de los puntos en contra, como el caso en el que no se conoce la constante de Lipschitz [7], o donde se mejora el rendimiento del algoritmo [8].

El algoritmo de Evolución diferencial (Differential Evolution), como ejemplo de algoritmo probabilístico, presenta varias ventajas. Entre ellas, que se trata de un algoritmo de fácil implementación, que es muy rápido, que es fácilmente adaptable a n dimensiones sin que aumente de forma importante el tiempo de respuesta, y que es aplicable no sólo a funciones sino también a expresiones computacionales. En contra, su naturaleza probabilística, que hace que cada vez que se ejecute se obtengan resultados diferentes (aunque dentro del rango de error prefijado). También que se obtiene el valor del mínimo alcanzado y uno de los puntos donde se alcanza, pero no siempre el conjunto de puntos donde se alcanza este valor mínimo (minimizadores). Y para algunos tipos de funciones (deceptive functions) se ha demostrado que este algoritmo no garantiza su convergencia[9].

Referencias

[1] Jerrold E. Marsden and Anthony J. Tromba. *Cálculo Vectorial*. Pearson Educación, Madrid, 5ª edition, 2004.

[2] Efim A. Galperin. The cubic algorithm. *Journal of Mathematical Analisys and Applications 112, 635-640*, 1982.

[3] Miguel Delgado Pineda. Optimización global de funciones continuas no diferenciables. *100cias@uned, Nº1 (Nueva época)*, 2008.

[4] Rainer Storn and Kenneth Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous. *Journal of Global Optimization 11, 341-359*, 1997.

[5] Xin-She Yang. *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, United Kingdom, 2ª edition, 2010.

[6] Swagatam Das and Ponnuthurai Nagaratnam Suganthan. Differential evolution: A survey of the state-of-art. *IEEE Transactions on Evolutionary Computacion. Vol 15, Nº 1. February 2011*, 2011.

[7] E. A. Galperin. The alpha algorithm and the appliocation of the cubic algorithm in case of unknown lipschitz constant. *Computers Math. Applic. Vol 25, No 10/11, pp, 71-78*, 1993.

[8] E. A. Galperin. The fast cubic algorithm. *Computers Math. Applic. Vol 25, No 10/11, pp, 147-160*, 1993.

[9] Zhongbo Hu, Qinghua Su, Xianshan Yang, and Zenggang Xiong. Not guaranteeing convergence of differential evolution on a class of multimodal functions. *Applied Soft Computing. January 2016*, 2016.

[4] O acorde con el tipo de coordenadas utilizadas: polares, esféricas, cilíndricas