UNA INTRODUCCIÓN A LA OPTIMIZACIÓN

F. VADILLO

Resumen. En este documento se introducen los método numéricos para calcular el punto $\mathbf{v}_m \in \mathbb{R}^n$ donde una función $F : \mathbb{R}^n \to \mathbb{R}$ alcanza su valor mínimo. Los métodos y algoritmos presentados son los clásicos: descenso, gradiente y gradiente conjugado. Después se describe el algoritmo de Nelder-Mead para problemas no diferenciables y se finaliza referenciando los métodos de computo evolutivo. Los métodos de optimización con restricciones no estudiados en este documento se pueden aprender en textos clásicos como [6], [13], [2], [10], [3] y [4].

Índice

1. Introducción 2
1.1. Resultados teóricos 2
1.2. El método de Newton 3
2. Métodos de descenso 4
3. Métodos de descenso para sistemas lineales 5
3.1. SDM para un sistema lineal 5
3.2. CGM (Conjugate Gradient Method) para un sistema lineal 7
4. Métodos de descenso para problemas generales 8
5. Minimización sin diferenciabilidad 8
6. Métodos de cómputo evolutivo 10
6.1. Aceptación probabilística 10
6.2. Selección probabilística 11
6.3. Búsqueda aleatoria 11
6.4. Algorithms Genetics (AG) 12
6.5. Evolution Strategies (ES) 12
6.6. Particle Swarm Optimization (PSO) 12
6.7. Differential Evolution (DE) 12
6.8. Harmony Search Algorithm (HSA) 12
6.9. Artificial Immune Systems (AIS) 12
6.10. Electromagnetism-like Optimization (EMO) 12
6.11. Artificial Bee Colony (ABC) 12
Referencias 13

Received by the editors 22 de marzo de 2021.
www.ehu.es/~mepvaarf.
Departamento de Matemáticas de la UPV/EHU.

1. Introducción

Dada una función $F : \mathbb{R}^n \to \mathbb{R}$ se busca su mínimo, es decir, el punto $\mathbf{v}_m \in \mathbb{R}^n$ tal que

$$F(\mathbf{v}_m) \;=\; \min_{\mathbf{v}\in\mathbb{R}^n} \; F(\mathbf{v}). \tag{1.1}$$

Ejemplo 1.1. La resolución un sistema lineal $\mathbf{Av} = \mathbf{b}$ se puede reescribir como el problema del hallar el mínimo de la función

$$F(\mathbf{v}) \;=\; ||\mathbf{Av} - \mathbf{b}\,||.$$

Ejemplo 1.2. El problema de resolver el sistema de ecuaciones no lineales

$$\begin{aligned} x^2 + 4y^2 &= 1, \\ 4x^2 + y^2 &= 1, \end{aligned}$$

es también el problema de minimizar la función

$$F(x, y) \;=\; (x^2 + 4y^2 - 1)^2 + (4x^2 + y^2 - 1)^2,$$

cuya gráfica y curvas de nivel se dibujan en la Figura 1.2.

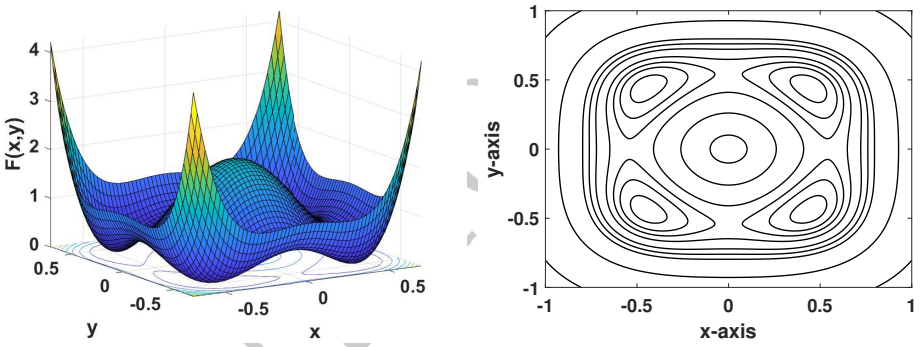

Figura 1.1. Gráfica y curvas de nivel de $F(x, y)$

Ejemplo 1.3. El famoso Traveling Salesman Problem (TSP) cuya versión más sencilla trata que hallar el camino más corta que conecta un conjunto dado de puntos del plano. Es un problema cuya solución requiere un tiempo que aumenta exponencialmente con el número de puntos (NP-complete problem) (Ver [8] y sus referencias).

1.1. Resultados teóricos.

Desde el punto de vista teórico los resultados clásicos más importantes son los siguientes:

Teorema 1.4. *El teorema de Taylor para funciones de varias variables Dado un punto* $\mathbf{v} = (v_1, \cdots, v_n)^T$ *y una dirección* $\mathbf{d} = (d_1, \cdots, d_n)^T$ *el desarrollo de Taylor establece que*

$$F(\mathbf{v} + \mathbf{d}) = F(\mathbf{v}) + \nabla F(\mathbf{v})^T \cdot \mathbf{d} + \frac{1}{2}\; \mathbf{d}^T \nabla^2 \mathbf{F}(\mathbf{v})\; \mathbf{d} + \mathcal{O}(||\mathbf{d}||^3), \tag{1.2}$$

donde $\nabla F(\mathbf{v})$ *es el vector gradiente de* F *en* $\mathbf{v}$, *y* $\nabla^2\mathbf{F}(\mathbf{v})$ *es la matriz Hessiana de las segundas derivadas de* F *en* $\mathbf{v}$ *definidas de la forma*

$$\nabla F(\mathbf{v}) = \begin{pmatrix} \frac{\partial F}{\partial v_1} \\ \frac{\partial F}{\partial v_2} \\ \vdots \\ \frac{\partial F}{\partial v_n} \end{pmatrix}, \quad \nabla^2 F(\mathbf{v}) = \begin{pmatrix} \frac{\partial^2 F}{\partial v_1^2} & \frac{\partial^2 F}{\partial v_1 \partial v_2} & \cdots & \frac{\partial^2 F}{\partial v_1 \partial v_n} \\ \frac{\partial^2 F}{\partial v_2 \partial v_1} & \frac{\partial^2 F}{\partial v_2^2} & \cdots & \frac{\partial^2 F}{\partial v_2 \partial v_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 F}{\partial v_n \partial v_1} & \frac{\partial^2 F}{\partial v_n \partial v_2} & \cdots & \frac{\partial^2 F}{\partial v_n^2} \end{pmatrix},$$

y el resto $\mathcal{O}(||\mathbf{p}||^3)$ *depende de las derivadas de* F *de orden mayor que 2.*

Demostración. Vea por ejemplo los clásicos [9, Capítulo 3] , [7, Chapter IV.4] o también [11, Capítulo 7]. □

El escalar

$$\nabla F(\mathbf{v})^T \cdot \mathbf{d} \;=\; \sum_{j=1}^{n} d_j \; \frac{\partial F}{\partial v_j},$$

es la derivada de F en el punto $\mathbf{v}$ en la dirección $\mathbf{d}$, y en el segundo término en el desarrollo (1.2) es la forma cuadrática

$$\mathbf{d}^T \nabla^2 F(\mathbf{v})\; \mathbf{d} \;=\; \sum_{i,j=1}^{n} \frac{\partial^2 F}{\partial v_i \partial v_j}\; d_i\; d_j.$$

Entonces, si $\mathbf{v}^*$ es un mínimo local si para cualquier dirección $\mathbf{d}$

$$F(\mathbf{v}^* + \mathbf{d}) = F(\mathbf{v}^*) + \nabla F(\mathbf{v}^*)^T \cdot \mathbf{d} + \frac{1}{2}\; \mathbf{d}^T \nabla^2 F(\mathbf{v}^*)\; \mathbf{d} + \mathcal{O}(||\mathbf{d}||^3) \geq F(\mathbf{v}^*),$$

con $0 < ||\mathbf{d}|| \ll 1$.

Teorema 1.5. *Condición de mínimo local Suponiendo que la función* F *es suficientemente regular,*

(a) Una condición necesaria para que F *tenga un mínimo local es un punto* $\mathbf{v}^*$ *es que sea un punto crítico, es decir,* $\nabla F(\mathbf{v}^*) = \mathbf{0}$ *y la matriz hessiana* $\nabla^2 F(\mathbf{v}^*)$ *sea semi-definida positiva.*

(b) Una condición suficiente para que F *tenga un mínimo local es un punto* $\mathbf{v}^*$ *es que sea un punto crítico y la matriz hessiana sea definida positiva.*

Demostración. Vea por ejemplo los clásicos [9, Capítulo 3] , [7, Chapter IV.4] o también [11, Capítulo 7]. □

1.2. El método de Newton. Según estos resultados clásicos, los puntos críticos son las soluciones del sistema no lineal

$$\mathbf{f}(\mathbf{v}) \;\equiv\; \nabla F(\mathbf{v}) \;=\; \mathbf{0}, \tag{1.3}$$

cuyo Jacobina es el Hessiano de la función F, es decir, $\mathbf{J}(\mathbf{v}) = \nabla^2 F(\mathbf{v})$. Por tanto, si se aplicara el método de Newton quedaría el siguiente algortimo

Algoritmo 1.6. Comentando en un $\mathbf{v}_0$, para $k = 0, 1, \cdots$

1. Se calcula $\mathbf{d}_k$ resolviendo el sistema $\nabla^2 F(\mathbf{v}_k)\; \mathbf{d}_k = -\nabla F(\mathbf{v}_k)$.
2. Se calcula $\mathbf{v}_{k+1} = \mathbf{v}_k + \mathbf{d}_k$.

Ejemplo 1.7. Ejemplo 9.5 [1, pp. 261].

Las dificultades prácticas de este método son evidentemente:

- Requiere la existencia del Hessiano que debe ser definido positivo y además necesita ser evaluado en cada iteración.
- El método necesita resolver un sistema lineal en cada paso.
- La convergencia es sólo local.

2. Métodos de descenso

Para cualquier dirección $\mathbf{d}$ en un punto $\mathbf{v}_k$ tal que $\nabla F(\mathbf{v}_k) \neq \mathbf{0}$ y si $||\mathbf{d}||$ es muy pequeño,

$$F(\mathbf{v}_k + \mathbf{d}) \approx F(\mathbf{v}_k) + \nabla F(\mathbf{v}_k)^T \cdot \mathbf{d}$$

y para que $F(\mathbf{v}_k + \mathbf{d}) < F(\mathbf{v}_k)$, la derivada direccional debe ser negativa, es decir,

$$\nabla F(\mathbf{v}_k)^T \cdot \mathbf{d} < 0,$$

lo que se denomina una dirección de descenso.

Por otra parte, la desigualdad de Cauchy-Schwarz establece que

$$| \nabla F(\mathbf{v}_k)^T \cdot \mathbf{d} | \leq ||\nabla F(\mathbf{v}_k)||_2 \, ||\mathbf{d}||_2,$$

es decir, que el descenso mas rápido será para

$$\mathbf{d}_k = -\nabla F(\mathbf{v}_k).$$

conocido como el método del gradiente.

Los Métodos iterativos para aproximar el punto mínimo comienza en una posición $\mathbf{v}_1$ y después construye una sucesión con una iteración del tipo

$$\begin{aligned} \mathbf{v}_{k+1} &= \mathbf{v}_k + \alpha_k \, \mathbf{d}_k, \\ \mathbf{d}_k &= -\mathbf{B}_k^{-1} \, \nabla F(\mathbf{v}_k), \end{aligned}$$

tal que si $\mathbf{B}_k$ es definida positiva, entonces $\mathbf{d}_k$ es una dirección de descenso porque

$$\nabla F(\mathbf{v}_k)^T \cdot \mathbf{d}_k = -\nabla F(\mathbf{v}_k)^T \mathbf{B}_k^{-1} \, \nabla F(\mathbf{v}_k) < 0.$$

Algunas posible elecciones:

1. En el método del gradiente $\mathbf{B}_k = \mathbf{I}$.
2. El método de Newton tomaría $\mathbf{B}_k = \nabla^2 F(\mathbf{v}_k)$ con las dificultades e inconvenientes comentadas.
3. Utilizar una combinación de las dos $\mathbf{B}_k = \nabla^2 F(\mathbf{v}_k) + \mu_k \mathbf{I}$ con $\mu_u \geq 0$ tal que $\mathbf{B}_k$ sea definida positiva y por tanto una dirección de descenso.

Por otra parte, en cada iteración se debe seleccionar el tamaño del paso α_k, en particular, para el método de Newton $\alpha_k = 1$. Evidentemente y si fuera posible, la mejor opción sería minimizar la función

$$\psi(\alpha) = F(\mathbf{v}_k + \alpha \, \mathbf{d}_k).$$

que son métodos de máximo descenso (SDM) steepest descent method.

Ejemplo 2.1. En este ejemplo de la referencia [5, pag.5] se busca el mínimo de la función

$$f(x_1, x_2) \;=\; 10 - \exp(-(x_1^2 + 3x_2^2)), \quad -1 \le x_1, x_2 \le 1, \tag{2.1}$$

representada en la parte izquierda de la Figura 2.1.

Utilizando un método gradiente con $\alpha_k = 0{,}05$ constante y aproximando los gradientes con diferencias finitas de paso $h = 0{,}001$, es decir

$$\begin{aligned} \frac{\partial f}{\partial x_1}(x_1, x_2) &\approx \frac{1}{h}(f(x_1 + h, x_2) - f(x_1, x_2)), \\ \frac{\partial f}{\partial x_2}(x_1, x_2) &\approx \frac{1}{h}(f(x_1, x_2 + h) - f(x_1, x_2)), \end{aligned}$$

en la parte derecha se representan 100 iteraciones con un punto inicial aleatorio.

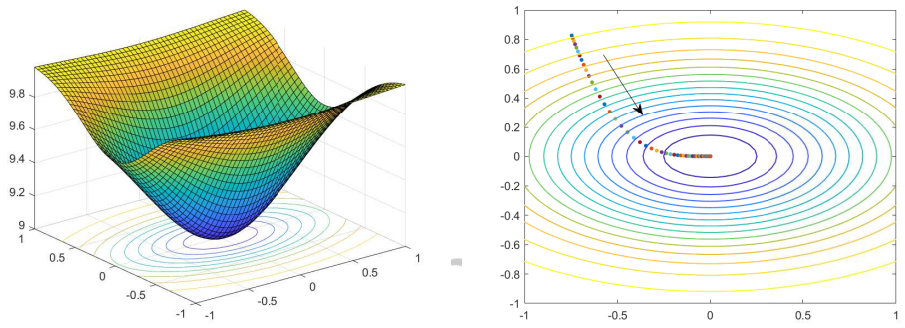

FIGURA 2.1. Gráfica del ejemplo (2.1) y el resultado de un algoritmo descendente.

3. MÉTODOS DE DESCENSO PARA SISTEMAS LINEALES

3.1. SDM para un sistema lineal. Se considera el problema de resolver el sistema lineal

$$\mathbf{A}\ \mathbf{v} = \mathbf{b}, \tag{3.1}$$

donde $\mathbf{A}$ es una matriz simétrica definida positiva y $\mathbf{b}$ un vector dado. Evidentemente resolver el sistema (3.1) es equivalente a minimizar la función

$$F(\mathbf{v}) \;=\; \frac{1}{2}\ \mathbf{v}^T \mathbf{A}\ \mathbf{v} - \mathbf{b}^T \cdot \mathbf{v},$$

cuyo gradiente es

$$\nabla F(\mathbf{v}) \;=\; \mathbf{A}\ \mathbf{v} - \mathbf{b},$$

con lo que el método del gradiente quedaría

$$\mathbf{v}_{k+1} \;=\; \mathbf{v}_k - \alpha_k\ (\mathbf{A}\ \mathbf{v}_k - \mathbf{b}) \;=\; \mathbf{v}_k + \alpha_k\ \mathbf{r}_k,$$

es decir, curiosamente el residuo $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\ \mathbf{v}_k$ sería la dirección de descenso.

Por otra parte, para calcular el tamaño en el desplazamiento se debería minimizar la función

$$\psi(\alpha) \;=\; \frac{1}{2}\,(\mathbf{v}_k+\alpha\mathbf{r}_k)^T\mathbf{A}\,(\mathbf{v}_k+\alpha\mathbf{r}_k)-\mathbf{b}^T\cdot(\mathbf{v}_k+\alpha\mathbf{r}_k),$$

cuyos puntos estaciones verifican

$$\alpha\;\mathbf{r}_k^T\mathbf{A}\;\mathbf{r}_k+\mathbf{r}_k^T\mathbf{A}\mathbf{v}_k-\mathbf{r}_k^T\cdot\mathbf{b}\;=\;\alpha\;\mathbf{r}_k^T\mathbf{A}\;\mathbf{r}_k-\mathbf{r}_k^T\cdot\mathbf{r}_k\;=\;0,$$

con lo que se concluye que

$$\alpha_k\;=\;\frac{\mathbf{r}_k^T\cdot\mathbf{r}_k}{\mathbf{r}_k^T\mathbf{A}\;\mathbf{r}_k}\;=\;\frac{\mathbf{r}_k^T\cdot\mathbf{r}_k}{\mathbf{r}_k^T\cdot\mathbf{q}_k}. \tag{3.2}$$

Es decir, en el método de máximo descenso para el sistema lineal (3.1) la dirección es el residuo $\mathbf{r}_k$ y el paso se calcula usando (3.2).

Ejemplo 3.1. En este primer ejemplo se aplica el SDM al sistema

$$\begin{pmatrix} 3 & -2 \\ -2 & 4 \end{pmatrix}\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \;=\; \begin{pmatrix} 1 \\ -1 \end{pmatrix},$$

con $\mathbf{v}_1=(1,2)^T$.

El test ó criterio de parada de la iteración utilizado es $||\mathbf{v}_{k+1}-\mathbf{v}_k||_\infty<10^{-4}$.

El SDM realiza 14 iteraciones que se puede observa en lado izquierdo de la Figura 3.1 con un residuo del orden de 4×10^{-5}. En este ejemplo se observa claramente como se nueve en direcciones perpendiculares a las curvas de nivel. En este ejemplo el número de condición $\kappa(A)_\infty\approx 4{,}5$.

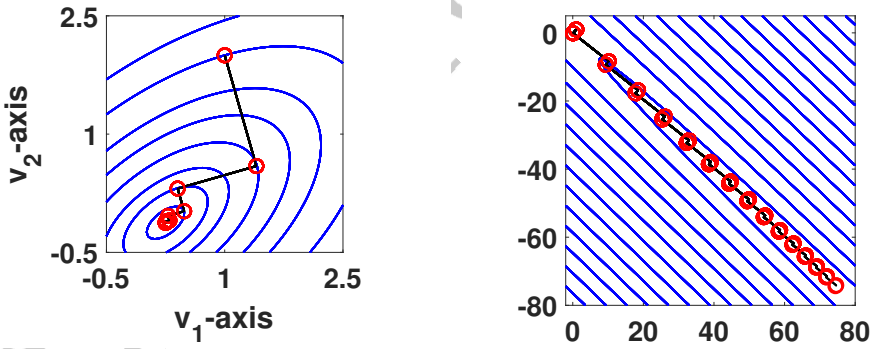

FIGURA 3.1. SDM para el ejemplo 3.1 a la izquierda y a la derecha el ejemplo 3.2.

Ejemplo 3.2. En el segundo ejemplo se aplica el SDM al sistema

$$\begin{pmatrix} 5 & 4{,}99 \\ 4{,}99 & 5 \end{pmatrix}\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \;=\; \begin{pmatrix} 1 \\ -1 \end{pmatrix},$$

con $\mathbf{v}_1=(1,1)^T$. El en lado derecho de la figura 3.1 se puede observar que las curvas de nivel son mas alargadas y después de 200 pasos el residuo es del orden de 9×10^{-3} con un número de condición $\kappa(A)_\infty\approx 999$.

3.2. CGM (Conjugate Gradient Method) para un sistema lineal. Para mejorar el SDM se necesita modificar la dirección de descenso, si antes $\mathbf{d}_k = \mathbf{r}_k$ ahora

$$\mathbf{d}_k \;=\; \mathbf{r}_k + \beta_{k-1}\,\mathbf{d}_{k-1}, \quad \text{para} \quad k = 2, 3, \cdots$$

y la cuestión ahora es cómo elegir el β_{k-1}.

Si se observa la Figura 3.1 es evidente que $\mathbf{r}_k$ es ortogonal a $\mathbf{r}_{k-1}$ y por tanto el SDM en realidad sólo tiene dos direcciones ortogonales, la dirección usada $\mathbf{r}_1$ es la misma que $\mathbf{r}_3, \mathbf{r}_5, \mathbf{r}_7, \cdots$ va saltando. Una forma de evitar esto es elegir β_{k-1} tal que $\mathbf{d_k}$ sea ortogonal a $\mathbf{d_{k-1}}$ en algún sentido.

Para entender como procede el algoritmo se consideran los dos primeros vectores $\mathbf{v}_1, \mathbf{v}_2$. Los residuos son

$$\begin{aligned} \mathbf{r}_1 &= \mathbf{d}_1 \\ \mathbf{r}_2 &= \mathbf{b} - \mathbf{A}\mathbf{v}_2 = \mathbf{b} - \mathbf{A}(\mathbf{v}_1 + \alpha_1 \mathbf{d}_1) = \mathbf{r}_1 - \alpha_1 \mathbf{A}\mathbf{r}_1, \end{aligned}$$

de donde se deduce que $\mathbf{r}_2^T \cdot \mathbf{r}_1 = 0$ por la expresión (3.2).

Por otra parte, para que

$$\mathbf{d}_2^T \mathbf{A}\mathbf{d}_1 = (\mathbf{r}_2 + \beta_1\, \mathbf{d}_1)^T \mathbf{A}\mathbf{r}_1 \;=\; 0$$

se debe tomar

$$\beta_1 \;=\; -\,\frac{\mathbf{r}_2^T \mathbf{A}\mathbf{r}_1}{\mathbf{r}_1^T \mathbf{A}\mathbf{r}_1} \;=\; \frac{\mathbf{r}_2^T \cdot (\mathbf{r}_2 - \mathbf{r}_1)}{\alpha_1\, \mathbf{r}_1^T \mathbf{A}\mathbf{r}_1} \;=\; \frac{\mathbf{r}_2^T \cdot \mathbf{r}_2}{\mathbf{r}_1^T \cdot \mathbf{r}_1}.$$

Siguiendo esta idea particular, el algoritmo quedará de la siguiente forma:

Algoritmo 3.3.

- Se comienza con un $\mathbf{v}_1$.
- Se calcula el residuo: $\mathbf{r}_1 = \mathbf{b} - \mathbf{A}\mathbf{v}_1$ llamando $\mathbf{d}_1 = \mathbf{r}_1$.
- Para $k = 1, 2, \cdots, n$
  - $\mathbf{q_k} = \mathbf{A}\mathbf{d}_k$.
  - $\alpha_k \;=\; \dfrac{\mathbf{r}_k^T \cdot \mathbf{r}_k}{\mathbf{d}_k^T \cdot \mathbf{q}_k}$.
  - $\mathbf{v}_{k+1} = \mathbf{v}_k + \alpha_k\, \mathbf{d}_k$.
  - $\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k\, \mathbf{q}_k$.
  - $\beta_k \;= \dfrac{\mathbf{r}_{k+1}^T \cdot \mathbf{r}_{k+1}}{\mathbf{r}_k^T \cdot \mathbf{r}_k}$.
  - $\mathbf{d}_{k+1} = \mathbf{r}_{k+1} + \beta_k\, \mathbf{d}_k$.

En la figura 3.2 se puede apreciar este algoritmo aplicados a los ejemplos 3.1 y 3.2. Para el primer caso en la tercera iteración el residuo es del orden de $5{,}6 \times 10^{-17}$, mientras que para el segundo ejemplo con sólo tres iteraciones es residuo es del orden de 7×10^{-12}.

El algoritmo CGM tiene una propiedad muy destacara que se demuestra en el siguiente teorema:

Teorema 3.4. *El método del gradiente conjugado cuando se utiliza para resolver sistema lineales* $\mathbf{A}\mathbf{v} = \mathbf{b}$ *con la matriz* $\mathbf{A}$ *definida positiva, alcanza la solución exacta en* m *iteraciones con* $m \leq n + 1$.

Demostración. Ver [6] o [10]. □

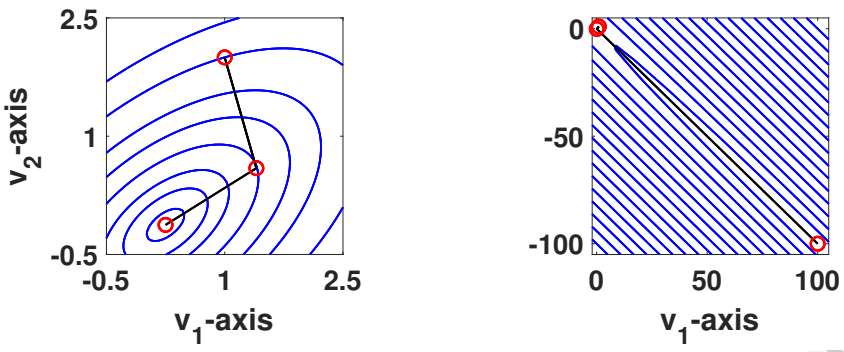

FIGURA 3.2. CGM para el primer ejemplo a la izquierda y a la derecha el segundo ejemplo.

Para ver algunos test de error y velocidad de convergencia del CGM ver referencia [8, pag. 358].

4. MÉTODOS DE DESCENSO PARA PROBLEMAS GENERALES

Volviendo al problema general (1.1), para aproximar el mínimo $\mathbf{v}_m$, los métodos iterativos eligen un valor inicial $\mathbf{v}_1$ y construyen una sucesión

$$\mathbf{v}_{k+1} = \mathbf{v}_k + \alpha_k \, \mathbf{d}_k, \quad k = 1, 2, \cdots$$

con una dirección de descenso $\mathbf{d}_k$ tal que $\mathbf{d}_k^T \cdot \mathbf{g}_k < 0$ donde $\mathbf{g}_k = \nabla \mathbf{F}(\mathbf{v}_k)$ es el gradiente.

Las dos opciones más utilizadas son:

SDM: En este caso $\mathbf{d}_k = -\mathbf{g}_k$.

CGM: Ahora $\mathbf{d}_k = -\mathbf{g}_k + \beta_{k-1} \, \mathbf{d}_{k-1}$ con $\beta_0 = 0$ y

$$\beta_k = \frac{\mathbf{g}_{k+1}^T \cdot \mathbf{g}_{k+1}}{\mathbf{g}_k^T \cdot \mathbf{g}_k}, \quad \text{para} \quad k = 1, 2, \cdots$$

conocido como el método de Fletcher-Reeves .

Una leve modificación calcula

$$\beta_k = \frac{\mathbf{g}_{k+1}^T \cdot (\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{g}_k^T \cdot \mathbf{g}_k}, \quad \text{para} \quad k = 1, 2, \cdots$$

que es el método de Polak-Ribière .

Para estudiar los detalles ver las referencias clásicas [6], [10] y [14]

5. MINIMIZACIÓN SIN DIFERENCIABILIDAD

Cuando la función $F(\mathbf{v})$ no tiene derivadas, evidentemente los métodos basados en el gradiente son inaplicables y se debe recurrir a otros algoritmo. El más conocido es el Nelder-Mead de 1965, y aunque hasta hace unos años apenas se había utilizado sus recientes implementaciones han ganado mucha eficacia.

La idea básica del algoritmo es simplemente geométrica: dados tres puntos en la superficie $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ es posible construir una dirección descendente primero eligiendo el punto entre tres de mayor valor F y buscando una dirección entre los otros dos.

Algoritmo 5.1. Algoritmo de Nelder-Mead en $\mathbb{R}^2$.

Paso 0.
- Se eligen $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$.
- Se toma una constante $\alpha > 0$, por ejemplo $\alpha = 1$.
- Se toma una constante $0 < \beta < 1$, por ejemplo $\beta = 1/2$.
- Se toma una constante $0 < \delta < 1$, por ejemplo $\delta = 1/2$.

Paso 1. Si fuera necesario se renumeran los vertices para que $F_1 \leq F_2 < F_3$.

Paso 2. Se calcule el centro entres $\mathbf{v}_1$ y $\mathbf{v}_2$, es decir, $\mathbf{v}_c = \frac{1}{2}(\mathbf{v}_1 + \mathbf{v_2})$ y la dirección $\mathbf{d} = \mathbf{v}_c - \mathbf{v}_3$.

Paso 3. Se calcula $\mathbf{v}_r = \mathbf{v}_c + \alpha\mathbf{d}$ y $F_r = F(\mathbf{v}_r)$.
- Si $F_1 \leq F_r < F_2$, entonces $\mathbf{v}_3 = \mathbf{v}_r$ y se vuelve al paso 1.
- Si $F_r < F_1$ se va la paso 4.
- Si $F_2 \leq F_r < F_3$ se va al paso 5.
- Si $F_3 < F_r$ se va al paso 6.

Paso 4. Expansion: se calcula $\mathbf{v}_e = \mathbf{v}_c + 2\alpha\mathbf{d}$ y $F_e = F(\mathbf{v}_e)$. Si $F_e < F_r$, entonces $\mathbf{v}_3 = \mathbf{v}_e$ o en caso contrario $\mathbf{v}_3 = \mathbf{v}_r$ y se vuelve al paso 1.

Paso 5. Dilatación: se calcula $\mathbf{v}_o = \mathbf{v}_c + \beta\mathbf{d}$ y $F_o = F(\mathbf{v}_o)$. Si $F_o < F_r$, entonces $\mathbf{v}_3 = \mathbf{v}_o$ y se vuelve al paso 1, en caso contrario se salta al paso 7.

Paso 6. Contracción: se calcula $\mathbf{v}_s = \mathbf{v}_3 + \beta\mathbf{d}$ y $F_s = F(\mathbf{v}_s)$ Si $F_s < F_3$, entonces $\mathbf{v}_3 = \mathbf{v}_s$ y se vuelve al paso 1, en caso contrario se salta al paso 7.

Paso 7. Se reemplaza $\mathbf{v}_i$ por $\mathbf{v}_1 + \delta(\mathbf{v}_i - \mathbf{v}_1)$ y se vuelve al paso 1.

El criterio de parada de este algoritmo estaría relacionado con el área del triángulo, cuando dicha área es menor que una tolerancia , la solución sería el punto $\mathbf{v}_1$.

Ejemplo 5.2. Se considera la función

$$F(x, y) = 10x^2 + y^2,$$

cuyo mínimo es el punto $(0, 0)$. En la Figura 5 se puede observar los seis primero paso.

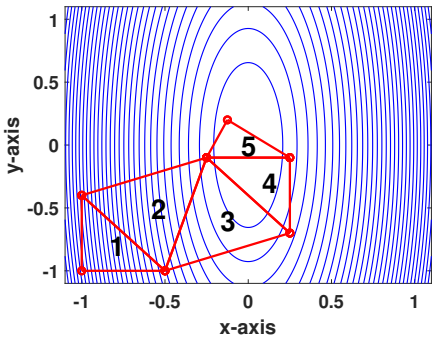

FIGURA 5.1. Algoritmo de Nelder-Mead para el ejemplo 5.2

Ejemplo 5.3. Se considera la función

$$F(x,y) \;=\; (x-y)^4 + 8xy - x + y + 3,$$

cuya solución se aprecia en ln la Figura 5 se puede observar los seis primero paso.

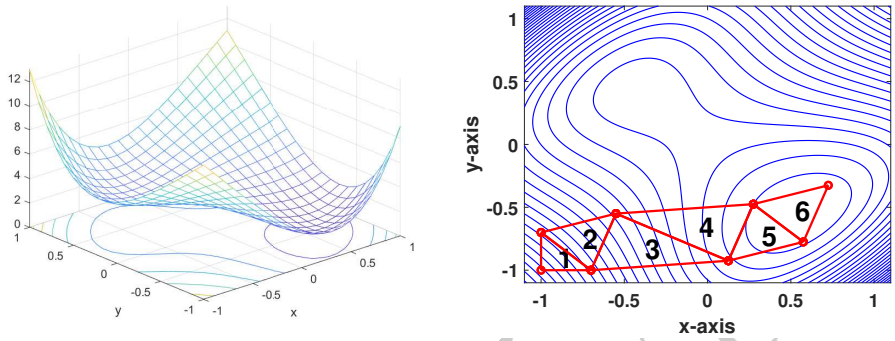

FIGURA 5.2. Algoritmo de Nelder-Mead para el ejemplo 5.3

6. MÉTODOS DE CÓMPUTO EVOLUTIVO

Los métodos de cómputo evolutivo según [12] o métodos meta-heurísticos en [14] son métodos de optimización que no utilizan el gradiente de la función objeto, esto permite aplicarlos a problemas con funciones más complicadas e irregulares, e incluso a funciones que resultan de simulaciones y modeles experimentales.

En general, son métodos que utilizan reglas heurísticas que generan patrones de búsqueda de soluciones, reglas basadas en simulaciones de diferentes procesos naturales. Los métodos de cómputo evolutivo son estocástico porque siempre utilizan procesos aleatorios para determinar las direcciones de búsqueda, por lo que generalmente es muy difícil obtener resultados teóricos y la mayoría de sus propiedades se descubren experimentalmente.

El problema trata de encontrar el vector $\mathbf{v}_m \in \mathbb{R}^n$ tal que

$$F(\mathbf{v}_m) \;=\; \min_{\mathbf{v}\in\mathbb{X}} F(\mathbf{v}). \tag{6.1}$$

En un algoritmo evolutivo, en cada iteración se dispone de una población de soluciones candidatas $\mathbf{P}^k = \{\mathbf{v}_1^k, \cdots, \mathbf{v}_N^k\}$ las cuales se van modificando según diferentes criterios o conceptos.

6.1. Aceptación probabilística. Es un concepto que puede formularse de la siguiente manera: una acción $\mathbf{A}$ tiene asociada una probabilidad $P_A \in [0,1]$, entonces se genera un número aleatoria $r_A \sim \mathcal{U}(0,1)$ y si $r_A \leq P_A$ se ejecuta la acción $\mathbf{A}$ y en caso contrario no tendrá efecto.

6.2. Selección probabilística. Se trata de seleccionar un elemento del conjunto posible $\mathbf{P}^k = \{\mathbf{v}_1^k, \cdots, \mathbf{v}_N^k\}$ pero de forma que los de mayor calidad tengan mayor probabilidad de ser elegidos. Por ejemplo, la probabilidad de elegir $\mathbf{v}_i$ pudiera ser

$$P_i \;=\; \frac{F(\mathbf{v}_i^k)}{\sum_{j=1}^N F(\mathbf{v}_j^k)},$$

y su probabilidad acumulada

$$P_i^A \;=\; \sum_{j=1}^{i} P_j.$$

Entonces, el proceso de selección genera un número aleatoria $r \sim \mathcal{U}(0,1)$ y comienza por $\mathbf{v}_1^k$ si $P_1^k > r$. Si esta condición no se cumple se sigue con la segunda solución y se continua hasta que $P_i^k > r$ en cuyo caso el $\mathbf{v}_i^k$ será el seleccionado.

6.3. Búsqueda aleatoria. En cada iteración el vector $\mathbf{v}^k$ se modifica añadiendo un vector aleatorio $\triangle\mathbf{v}$ tal que los nuevos candidatos serán

$$\mathbf{v}^{k+1} \;=\; \mathbf{v}^k + \triangle\mathbf{v},$$

donde $\triangle\mathbf{v} = (\triangle v_1, \cdots, \triangle v_n)^T$ con $\triangle v_j \sim \mathcal{N}(0, \sigma_j)$.

Una vez calculado $\mathbf{v}^{k+1}$ entonces

$$\mathbf{v}^{k+1} \;=\; \begin{cases} \mathbf{v}^{k+1}, & \text{si } F(\mathbf{v}^{k+1}) < F(\mathbf{v}^k), \\ \mathbf{v}^k, & \text{si } F(\mathbf{v}^{k+1}) \geq F(\mathbf{v}^k). \end{cases}$$

En esta búsqueda aleatoria, es importarte cuidar que $\mathbf{v}^{k+1} \in \mathbf{X}$ para lo cual lo mejor es asignar $F(\mathbf{v}^{k+1}) = +\infty$.

Ejemplo 6.1. Se considera el problema de maximizar la función

$$F(x,y) \;=\; 3(1-x_1)^2 e^{-x_1^2-(x_2+1)^2} + 10\Big(\frac{x_1}{5} - x_1^3 - x_2^5\Big)e^{-x_1^2-x_2^2} - \frac{1}{3}e^{-(x_1+1)^2-x_2^2},$$

para $-3 \leq x_1, x_2 \leq 3$, cuya gráfica se puede apreciar en la parte izquierda de la figura 6.1.

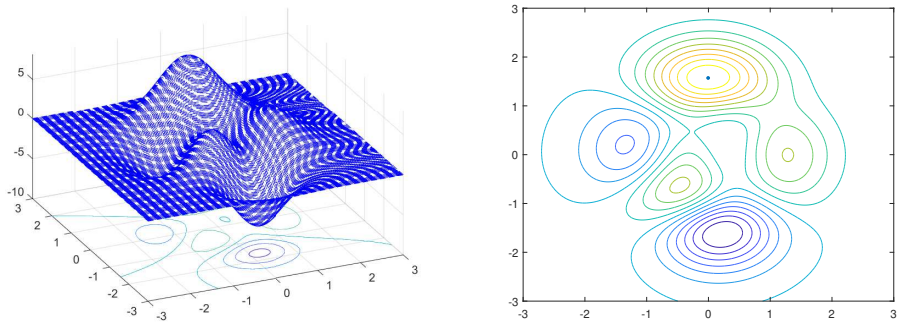

FIGURA 6.1. Gráfica y resultado del algoritmo de búsqueda aleatoria.

Ejemplo 6.2. Se considera el problema de maximizar la función de Eggholder

$$F(x,y) \;=\; -(x_2+47)\;\sin\left(\sqrt{|x_2+\frac{x_1}{2}+47)|}\right)-x_1\;\sin\left(\sqrt{|x_1-(x_2+47)|}\right),$$

para $-512 \leq x_1, x_2 \leq 512$, cuya gráfica se puede apreciar en la parte izquierda de la figura 6.2.

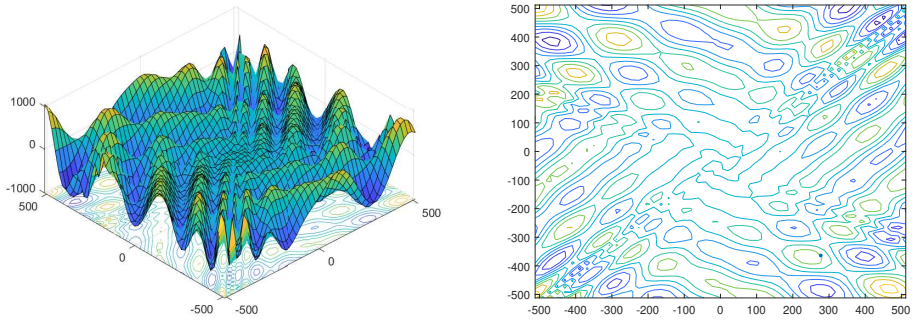

FIGURA 6.2. Gráfica y resultado para la función de Eggholder.

6.4. Algorithms Genetics (AG). [5, Cap. 2], [12, Chap. 3].

6.5. Evolution Strategies (ES). [5, Cap. 3], [12, Chap. 6].

6.6. Particle Swarm Optimization (PSO). [5, Cap. 4], [12, Chap. 11], [14, Chap. 10].

6.7. Differential Evolution (DE). [5, Cap. 5], [12, Chap. 12], [14, Chap. 10].

6.8. Harmony Search Algorithm (HSA). [5, Cap. 6], [12, Chap. 17].

6.9. Artificial Immune Systems (AIS). [5, Cap. 7], [12].

6.10. Electromagnetism-like Optimization (EMO). [5, Cap. 8].

6.11. Artificial Bee Colony (ABC). [5, Cap. 9], [12, Chap. 17].

Referencias

1. U.M. Ascher and C. Greif, *A Firts Course in Numerical Methods*, SIAM, 2011.
2. R. Baldick, *Applied Optimization. Formulation and Algorithms for Engineering Systems* , Cambridge University Press, 2006.
3. M. Bartholomew-Biggs, *Nonlineal Optimization with Engineering applications* , Springer, 2008.
4. S. Boyd and L. Vandenberghe, *Convex Optimization* , Cambridge International Science Publishing, 2009.
5. E.V. Cuevas, J.V. Osuna, D.A. Oliva, and M.A. Diaz, *Optimización. Algoritmos programados en MATLAB* , Alfaomega, 2017.
6. R. Fletcher, *Practical Methods of Optimization. Second Edition* , John Wiley and Sons, 2000.
7. E. Hairer and G. Wanner, *Analysis by its History* , Springer, 1996.
8. M.H. Holmes, *Introduction to Scientific Computing and Data Analysis*, Springer, 2016.
9. J.E. Marsden and A.J. Tromba, *Cálculo Vectorial. Cuarta edición*, Addison-Wesley Iberoamericana, 1998.
10. J. Nocedal and S.J. Wrigth, *Numerical Optimization. Second Edition* , Springer, 2006.
11. S.L. Salas, E. Hille, and G.J. Etgen, *Caculus. Una y varias variables. Cuarta edición* , Editorial Reverte, 2002.
12. D. Simon, *Evolutionary Optimization Algorithms. Biologically-Inspired and Population-Based Approaches to Computer Intelligence*, Wiley, 2013.
13. P. Ventkaratama, *Applied Optimization Algoritmos with MATLAB* , John Wiley & Sons, 2001.
14. Xin-She Yang, *Introduction to Mathematical Optimization* , Cambridge International Science Publishing, 2008.