Método del descenso más pronunciado para aproximar la solución de un sistema de ecuaciones matricial de la forma $A\vec{x} = b$

Andrés Bolaños

***Introducción*— En este texto se muestra información acerca de cómo se puede solucionar un sistema de ecuaciones matricial sin necesidad de utilizar la inversa de A. Sabemos que al multiplicar A^{-1} por el vector solución b obtendremos exactamente las componentes de $\vec{x}$. Sin embargo, calcular esta matriz previa a la operación puede ser computacionalmente costoso. Utilizando el método iterativo del descenso más pronunciado podemos encontrar el mínimo de la función multivariable asociada a A$\vec{x}$ = b, al obtener dichas coordenadas, estas serán aproximadamente igual a las componentes de la solución del sistema, dependiendo de la tolerancia definida. Esta implementación puede ser generalizada a n dimensiones, pero por motivos de practicidad y representaciones gráficas se utilizará matrices $2x2$.**

I. PALABRAS CLAVE

Descenso más pronunciado, Gradiente, Ecuación matricial, Optimización multivariable, Tolerancia, Costo computacional, Error

II. MARCO TEÓRICO

A. *Optimización multivariable*

La optimización puede ser descrita como: encontrar la mejor manera de realizar algo. Este enunciado determina que debe encontrarse una combinación de factores que permite obtener un resultado utilizando estos recursos de la mejor manera posible. Este es un arte de dos pasos, el primero de estos es transcribir el problema y colocar todas las barreras necesarias. Y el segundo es encontrar esa combinación eficiente para dicho problema.

Cuando una función depende de más de una variable, es decir que la variable dependiente está sujeta a más de una condición, se le conoce como multivariable.

$$\min_{x \in C} F(x)$$

B. *Optimización convexa y diferenciable sin restricciones*

1) Convexidad: La convexidad de una curva o una superficie, es la zona que se asemeja al exterior de una circunferencia o una superficie esférica, es decir, que tiene su parte sobresaliente dirigida al observador. Es el concepto opuesto a la 'concavidad'.

Una parte C de un espacio vectorial real es convexa si para cada par de puntos de C, el segmento que los une está totalmente incluido en C; es decir, un conjunto es convexo si se puede ir de cualquier punto a cualquier otro en línea recta, sin salir del mismo.

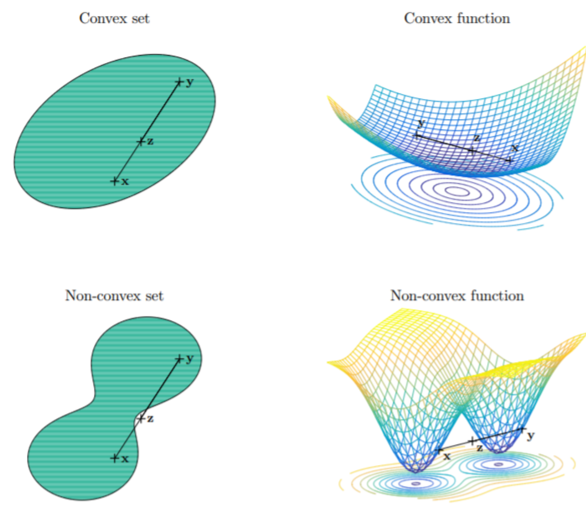


Fig. 1. Ilustración de un set convexo (arriba) y uno no convexo (abajo)

2) Función diferenciable: Junto con la convexidad, la diferenciabilidad es otra noción importante que permite distinguir entre los diferentes tipos de problemas de optimización. El concepto de función diferenciable es una generalización para el cálculo en varias variables del concepto más simple de función derivable. En esencia una función diferenciable admite derivadas en cualquier dirección y puede aproximarse al menos hasta primer orden por una aplicación afín.

La formulación rigurosa de esta idea intuitiva sin embargo es algo más complicada y requiere de conocimientos de álgebra lineal. Debe notarse que aunque una función de varias variables admita derivadas parciales según cada una de sus variables no necesariamente eso implica que sea una función diferenciable.

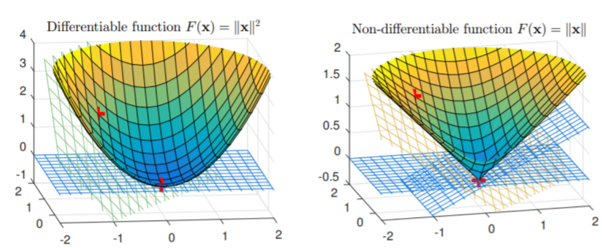


Fig. 2. Ejemplificación de una función no diferenciable y una diferenciable

C. Descenso más pronunciado

Utilizando este método podemos encontrar mínimos en una función multivariable. El gradiente negativo de la función en un punto indica la dirección de mayor decrecimiento. Es una dirección correcta para buscar el mínimo de la función.

Proceso:

1) Coordenada inicial: Este algoritmo requiere que en la entrada se coloque una coordenada inicial, comúnmente denominada "primera aproximación".

2) Definición de alpha: Dado un punto x y una dirección s (dada por el gradiente) se obtiene la siguiente función de una variable definida por:

$$\Phi(\alpha) = f(x + \alpha s)$$

3) Optimización de alpha: Ahora se optimiza $\Phi(\alpha)$ con alguno de los métodos de optimización de una variable, en este ejemplo se utilizará el método de la proporción áurea.

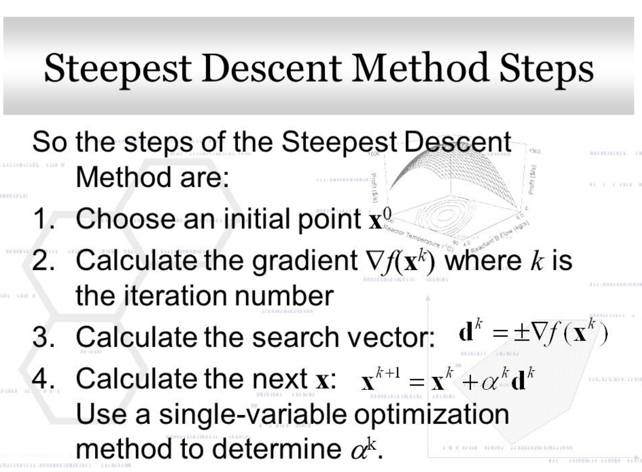
4) Salto: Alpha representa el salto que se dará en cada una de la iteraciones. Este salto se hará en la dirección que indique s, quien es determinada a partir del gradiente negativo valuado en x

5) Nueva X: La nueva X es calculada a partir de la anterior de la siguiente manera:

$$x_{n+1} = x_n + a_n s_n$$

6) Verificación: Se calcula el error entre la nueva coordenada x_{n+1} y x_n, esto es posible calculando la longitud del vector diferencia entre ambos. La magnitud de un vector a veces se llama la longitud de un vector, o la norma de un vector. Básicamente, la norma de un vector es una medida de distancia, simbolizada por una doble barra vertical.

7) Algoritmo: El algoritmo para utilizar el descenso más pronunciado es el siguiente:



III. Aplicación

A. Explicación

Teniendo todas las piezas juntas, podemos comenzar a hablar acerca de la aplicación que tiene este algoritmo en la solución de la ecuación matricial.

El descenso más pronunciado es un método que permite encontrar los máximos y mínimos de una función definida por varias variables. Utiliza el gradiente negativo para buscar dónde es que está el descenso más pronunciado. Optimiza alpha, que es el tamaño del salto en la dirección dada por el gradiente. Esta optimización puede ser hecha con cualquier método de optimización de una variable, para este ejemplo en específico se ha utilizado el método de *Golden Ratio*.

Cuando la matriz A es simétrica y positiva definida, tenemos una nueva clase de algoritmos para resolver $A\vec{x} = b$. Considere la función:

$$f(x) = \frac{1}{2}x^T Ax - x^T b$$

misma que surge de multiplicar la función orignal por x^T, despejándola para dejar todos los términos de un solo lado. Nótese que se multiplica uno de los términos por $\frac{1}{2}$. La razón es por un probable ajuste debido a que tendremos un término x^2, al hacer la derivadas parciales el radio espectral de la matriz debe ≤ 1.

Teniendo esta función definida por A y b, que es la información que obtenemos para resolver el sistema, podemos utilizarla como una función multivariable con dimensión equivalente al tamaño de la matriz. Dada esta definición, sabemos que la función tiene un mínimo y es único.

Por ejemplo, dada la matriz:

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

y

$$b = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

En este caso, dado que la matrix es $2x2$, la función multivariable asociada a esta matriz utilizando la ecuación anterior será de la siguiente forma:

$$f(u, v) = \frac{1}{2}(u^2 + 2v^2) - 3u - 4v$$

El gradiente de esta función quedará como:

$$\nabla F(u, v) = (3 - u, 2v - 4)$$

B. Funcionamiento del algoritmo

Para hacer que el algoritmo pueda funcionar en este ejemplo de Python, se debe definir únicamente la matriz A y un vector solución b, ya que se ha implementado una función que convierte cualquier matriz en una función multivariable. Así también, utilizando la librería Sympy el gradiente de esta función dinámica se encuentra para utilizarlo en el algoritmo.

Utilizando librerías de Python, es posible graficar los movimientos que dio el algoritmo a través de las curvas de nivel de la función.

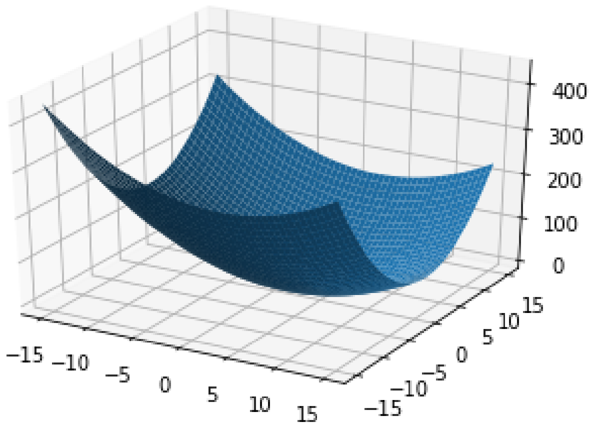


Fig. 3. Gráfica en 3d de la función multivariable

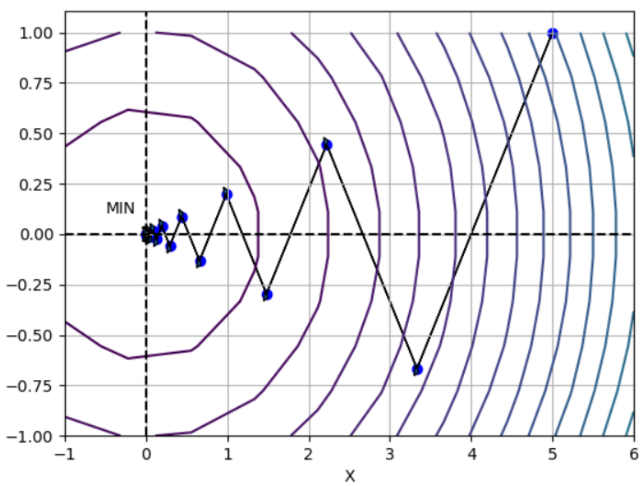


Fig. 4. Curvas de nivel utilizando el descenso más pronunciado

C. Código

```
#this is the method itself, receives xk first guess, the function, tolerance
def desc(xk,f,tol):
    #initial error
    error=1
    #this is the array with all the dots to graphic
    X=[]
    #array with all vectors
    S=[]
    #append to the dots array the initial guess
    X.append(xk)
    #make this algorithm while the error is still bigger than the tolerance
    #level
    while(error>tol):
        #calculates the first direction of the steepest descent using the
        #negative of the gradient
        sk=np.dot(-1.0,gradientef(xk))
        #optimizes alpha funciton using Golden Ration optimization of Python
        ak1=sp.optimize.golden(funcion,args=(xk,sk))
        #getting the direction using the step size
        sk_n=np.dot(ak1,sk)
        #append the direction
        S.append(sk_n)
        #calculating our new xk+1
        xk1=xk+np.dot(ak1,sk)
        #append the new dot
        X.append(xk1)
        #calculating the error
        error=np.linalg.norm(xk1-xk)
        #changing xk to the new value
        xk=xk1
    return xk,X,S
```

D. Resultado

El resultado de utilizar el algoritmo en el ejemplo expuesto en la explicación es el siguiente:

```
A:
[[1 0]
 [0 2]]
b:
[[3]
 [4]]
exact solution using numpy:
[[3.]
 [2.]]
the approx solution is: [3.00000004 2.        ]
iterations:
12
```

Referencias

[1] Stephane Canu, Rémi Flamary, David Mary. Introduction to optimization with applications in astronomy and astrophysics. 2016. ffhal-01346134f

[2] Bombal, R. Marín, Vera: Problemas de Análisis matemático: Cálculo Diferencial, 1988, ed. AC, ISBN 84-7288-101-6.

[3] Rauw, G., Manfroid, J., Gosset, E., Nazé, Y., Sana, H.; De Becker, M.; Foellmi, C.; Moffat, A. F. J. (2007): Early-type stars in the core of the young open cluster Westerlund 2. Astronomy and Astrophysics 463 (3): 981-991.

[4] https://arxiv.org/pdf/1805.09551.pdf

[5] N. M. Ball and R. J. Brunner. Data mining and machine learning in astronomy. International Journal of Modern Physics D, 19(07):1049–1106, 2010.

[6] N. Komodakis and J.-C. Pesquet. Playing with duality: An overview of recent primal? dual approaches for solving large-scale optimization problems. Signal Processing Magazine, IEEE, 32(6):31–54, 2015.

[7] M. R. Hestenes and E. Stiefel, “Methods of Conjugate Gradients for Solving Linear Systems,” J. Res. Natl. Bur. Standards 49 (1952) pp. 409-436.

[8] Clase de Scientific Computing con Silvia Ruiz en Universidad Francisco Marroquín, verano 2019

[9] Explicación de radio espectral con Mario Castillo, nobiembre 2019