# Tarea 2- Optimización 1

Daniel Vallejo Aldana

18 de febrero de 2020

## Problemas de convexidad

**Problema 1.** El conjunto $S = \{a \in \mathbb{R}^k | p(0) = 1, |p(t)| \leq 1 \text{ para } t \in [\alpha, \beta]\}$, donde $a = [a_1, .....a_k]^t$ y

$$p(t) = a_1 + a_2 t + ..... + a_k t^{k-1}$$

es convexo? Argumenta su respuesta.

**Solución** Si, el conjunto $S$ es convexo. Para evitar la ambigüedad de la notación, llamemos $p_a(t)$ a $p(t) = a_1 + a_2 t + ..... + a_k t^{k-1}$ para $a \in S$.
Consideremos ahora un $\lambda \in (0, 1)$, $t \in [\alpha, \beta]$ y sea $x \in \mathbb{R}^k$ tal que $x = \lambda a + (1 - \lambda)b$ para $a, b \in S$. Probaremos que $x \in S$. Para esto consideremos

$$\begin{aligned} |p_x(t)| &= |x_1 + x_2 t + ....x_k t^{k-1}| \\ &= |(\lambda a_1 + (1-\lambda)b_1) + (\lambda a_2 + (1-\lambda)b_2)t + .... + (\lambda a_k + (1-\lambda)b_k)t^{k-1}| \\ &= |\lambda p_a(t) + (1-\lambda)p_b(t)| \\ &\leq |\lambda p_a(t)| + |(1-\lambda)p_b(t)| \end{aligned}$$

Por la desigualdad triangular

$$= \lambda |p_a(t)| + (1-\lambda)|p_b(t)|$$

ya que $\lambda \in (0, 1)$

$$\leq \lambda + (1 - \lambda)$$

Ya que $a, b \in S$

$$= 1$$

Por lo anterior $x \in S$, como $a$ y $b$ son elementos arbitrarios de $S$ entonces $S$ es convexo.

**Problema 2.** Suponga que $f$ es convexa y que $\lambda_1 > 0$ y $\lambda_2 \leq 0$ con $\lambda_1 + \lambda_2 = 1$, y sean $x_1, x_2 \in dom(f)$. Muestra que la desigualdad

$$f(\lambda_1 x_1 + \lambda_2 x_2) \geq \lambda_1 f(x_1) + \lambda_2 f(x_2)$$

Siempre se cumple.

**Demostración** Sabemos por hipótesis que la función $f$ es convexa por lo que cumple la desigualdad de Jensen. Aplicando la desigualdad de Jensen a $f(1(\lambda_1 x_1 + \lambda_2 x_2) + \lambda_1 x_1 - \lambda_2 x_2)$ se tiene lo siguiente.

$$f\left(\frac{1(\lambda_1 x_1 + \lambda_2 x_2) + \lambda_1 x_1 - \lambda_2 x_2}{1 + \lambda_1 - \lambda_2}\right) \leq \frac{f(\lambda_1 x_1 + \lambda_2 x_2) + \lambda_1 f(x_1) - \lambda_2 f(x_2)}{1 + \lambda_1 - \lambda 2})$$

Utilizando la propiedad de que $\lambda_1 + \lambda_2 = 1$ y agrupando términos semejantes obtenemos que la desigualdad anterior puede escribirse de la siguiente manera.

$$\begin{aligned} f\left(\frac{1(\lambda_1 x_1 + \lambda_2 x_2) + \lambda_1 x_1 - \lambda_2 x_2}{1 + \lambda_1 - \lambda_2}\right) &\leq \frac{f(\lambda_1 x_1 + \lambda_2 x_2) + \lambda_1 f(x_1) - \lambda_2 f(x_2)}{1 + \lambda_1 - \lambda 2}) \\ f\left(\frac{2\lambda_1 x_1}{2\lambda_1}\right) &\leq \frac{f(\lambda_1 x_1 + \lambda_2 x_2) + \lambda_1 f(x_1) - \lambda_2 f(x_2)}{2\lambda_1}) \\ f\left(x_1\right) &\leq \frac{f(\lambda_1 x_1 + \lambda_2 x_2) + \lambda_1 f(x_1) - \lambda_2 f(x_2)}{2\lambda_1}) \\ 2\lambda_1 x_1 - \lambda_1 x_1 + \lambda_2 x_2 &\leq f\left(\lambda_1 x_1 + \lambda_2 x_2\right) \end{aligned}$$

Al manipular los términos de la desigualdad

$$\lambda_1 f(x_1) + \lambda_2 f(x_2) \leq f\left(\lambda_1 x_1 + \lambda_2 x_2\right)$$

De lo anterior queda probado que la desigualdad $\lambda_1 f(x_1) + \lambda_2 f(x_2) \leq f\left(\lambda_1 x_1 + \lambda_2 x_2\right)$ siempre se cumple ■

**Problema 3.** Muestra que la función $f : \mathbb{R}^n \to \mathbb{R}$

$$f(x) = -exp(-g(x))$$

Es convexa donde $g : \mathbb{R}^n \to \mathbb{R}$ tiene un dominio convexo y cumple

$$\begin{bmatrix} \nabla^2 g(x) & \nabla g(x) \\ \nabla^t g(x) & 1 \end{bmatrix} \succeq 0 \tag{1}$$

para $x \in dom(g)$.

**Demostración** Supongamos $f \in C^2$, y calcularemos $\nabla^2 f(x)$. Para esto procedemos por la regla de la cadena y tenemos lo siguiente.

$$\nabla^2 f(x) = exp(-g(x)) \left[-\nabla g(x) \nabla^t g(x) + \nabla^2 f(x)\right]$$

Probaremos que $\nabla^2 f(x)$ es semi definido positivo. Como $exp(-g(x))$ es siempre positivo probaremos que $[-\nabla g(x)\nabla^t g(x) + \nabla^2 f(x)]$ también lo es.

Utilizando la propiedad de que la matriz (1) es semidefinida positiva se tiene que para cualquier vector $\mathbf{v}$ se tiene que $\mathbf{v}^t Q \mathbf{v} \geq 0$ para $Q$ la matriz indicada en (1).

De lo anterior se tiene que

$$\begin{bmatrix}\mathbf{v}^t c\end{bmatrix} \begin{bmatrix}\nabla^2 g(x) & \nabla g(x) \\ \nabla^t g(x) & 1\end{bmatrix} \begin{bmatrix}\mathbf{v} \\ c\end{bmatrix} \geq 0$$

Y por lo tanto

$$\begin{aligned}\mathbf{v}^t \nabla^2 g(x)\mathbf{v} + 2c(\mathbf{v}^t \nabla g(x)) + c^2 &\geq 0 \\ \mathbf{v}^t \nabla^2 g(x)\mathbf{v} &\geq -2c(\mathbf{v}^t \nabla g(x)) - c^2\end{aligned}$$

Luego, consideremos

$$\begin{aligned}\mathbf{v}^t \nabla^2 f(x)\mathbf{v} &= exp(-g(x)) \left[-\mathbf{v}^t \nabla g(x) \nabla^t g(x)\mathbf{v} + \mathbf{v}^t \nabla^2 f(x)\mathbf{v}\right] \\ &\geq exp(-g(x)) \left[-(c + \mathbf{v}^t \nabla g(x))^2\right]\end{aligned}$$

Como $c$ es cualquier constante, tomemos $c = -\mathbf{v}^t \nabla g(x))^2$ con lo que se tiene que

$$\mathbf{v}^t \nabla^2 f(x)\mathbf{v} \geq 0$$

De lo cual se sigue que $\nabla^2 f(x)$ es semi definido positivo y en consecuencia la función $f$ es convexa que es lo que se buscaba demostrar ■.

# Método de gradiente descendente

## Resumen

El propósito del presente trabajo es describir, implementar y analizar el método de gradiente descendente para encontrar el argumento que minimiza una función $f : \mathbb{R}^k \to \mathbb{R}$ dada. Para encontrar el tamaño de paso necesario para el funcionamiento del algoritmo se utilizaran las técnicas de tamaño de paso fijo, backtracking y la aproximación del tamaño de paso utilizando la matriz Hessiana.

**Palabras clave:** Suficiente descenso, backtracking, Condiciones de Wolfe.

# Introducción

Cada iteración de un método de búsqueda en linea calcula una dirección de búsqueda $p_k$ y luego decide que tanto se mueve en esa dirección. Dicha iteración está dada por

$$x_{k+1} = x_k + \alpha_k p_k$$

donde $\alpha_k$ es un escalar positivo que se denomina longitud de paso [1].

Los métodos típicos de búsqueda en linea porponen una secuencia de candidatos para la longitud de paso $\alpha$, aceptando alguno de los valores propuestos cuando ciertas condiciones fueron satisfechas. EL método de búsqueda en linea se realiza en dos etapas: Encontrar un intervalos con longitudes de paso deseables y la segunda etapa es encontrar una buena longitud de paso en este intervalo [1].

Una de la condiciones principales para escoger el tamaño de paso dada la dirección de descenso $p_k$ es la condición de Armijo, la cual se puede expresar de la siguiente manera

$$f\left(x_k + \alpha p_x\right) \leq f(x_k) + c_1 \alpha \nabla f_k^t p_k \tag{2}$$

Sin embargo, esta condición no siempre nos asegura que el algoritmo haga un proceso razonable, por lo que es necesario agregar otra condición que nos asegure un buen comportamiento del algoritmo, a esta condición se le conoce como condición de curvatura dada por

$$\nabla f(x_k + \alpha_k p_k)^t p_k \geq c_2 \nabla f_k^t p_k \tag{3}$$

A las condiciones (2) y (3) se les conoce como las condiciones de Wolfe.

Como se mencionó anteriormente, la condición de suficiente descenso no es suficiente para asegurar que el algoritmo hace un proceso razonable para la dirección de búsqueda dada. Sin embargo si el método de búsqueda en linea escoge los candidatos a tamaño de paso apropiados utilizando el método de *backtracking* el algoritmo solo necesitará la condición de suficiente descenso [1].

La aproximación por *backtracking* asegura que el valor $\alpha_k$ corresponde a un valor fijo o bien es lo suficientemente pequeño para satisfacer la condición de suficiente descenso pero no es demasiado pequeño.

Las condiciones de paro para el algoritmo de gradiente descendente pueden ser la diferencia relativa entre $f(x_{k+1})$ y $f(x_k)$ dada por $\frac{\|f(x_{k+1}) - f(x_k)\|}{max((1, \|f(x_k\|))}$ o el error realitivo entre las $x_k$ así como el número de iteraciones. Para esta tarea se utilizaron las cuatro condiciones anteriores, donde el algoritmo para cuando alguna de ellas se cumple.

# Método/Algoritmo

Se parte de un algoritmo base de suficiente descenso, cuyo pseudocódigo se ilustra a continuación.

**Input**: $x_0$
**Output**: $x^*$
1: $k = 0$, $g_0 = \nabla f(x_0)$
2: **while** $\|g_0\| \neq 0$ **do**
3: $\quad \alpha_k = argmin_{\alpha>0} f(x_k + \alpha_k g_k)$
4: $\quad x_{k+1} = x_k + \alpha_k g_k$
5: $\quad g_{k+1} = \nabla f(x_{k+1})$
6: $\quad$ k=k+1
7: **end while**

Así mismo, el algoritmo para backtracking está dado de la siguiente forma.
**Input**: Una aproximación $\hat{\alpha} > 0$, $p \in (0,1)$, $c_1 \in (0,1)$, $\alpha = \hat{\alpha}$
**Output**: El tamaño de paso $\alpha_k > 0$
1: **Repeat** until $f(x_k + \alpha p_k) \leq f(x_k) + c_1 \alpha \nabla f_k^t p_k$
2: **end(repeat)**
3: $\alpha = \alpha_k$

Finalmente para la aproximación por Hessiano se tiene que $\alpha_k = \frac{g_k^t g_k \hat{\alpha}^2}{2(\hat{f} - f_k + \hat{\alpha} g_k^t g_k)}$ y se utiliza el mismo algoritmo base descrito anteriormente.

# Análisis de resultados

Se dividirá por casos el análisis de los resultados obtenidos.

## Caso 1: Función Rosenbruck con n=2

El propósito del algoritmo es encontrar el argumento que minimice la función

$$f(x) = 100(x2 - x1^2)^2 + (1 - x1)^2$$

Cuyo gradiente $\nabla f(x)$ está dado de la siguiente manera

$$\nabla f(x) = \left[2(200x_1^3 - 200x_1x_2 + x_1 - 1, 200(x_2 - x_1^2)\right]^t$$

Las condiciones iniciales para este problema fueron $x_0 = [-1{,}2, 1]$, tolerancias relativas para f y x del orden de $1 \times 10^{-14}$ y un número de iteraciones máximas de 18000, para el algoritmo con tamaño de paso fijo 0.001 se tiene lo siguiente.

| Tamaño de paso fijo | | | |
|---|---|---|---|
| $k$ | $\|x_{k+1} - x_k\|$ | $\|\nabla f(x_k)\|$ | $f(x_k)$ |
| 1 | 0.2328676877542266 | 49.03058747211632 | 5.352911580008958 |
| 2 | 0.049030587472116305 | 1.826163174050757 | 4.117789857067972 |
| 3 | 0.0018261631740506975 | 1.7747378177931528 | 4.114552150338376 |
| 17998 | 2.770930e-07 | 0.000276982 | 9.60049826127145e-08 |
| 17999 | 2.7698230831e-07 | 0.0002768716 | 9.59282787e-08 |
| 18000 | 2.7687161031e-07 | 0.000276760 | 9.585163617e-08 |

Para el algoritmo utilizando la técnica de backtracking se obtuvieron los resultados que a continuación se presentan

| Backtracking | | | |
|---|---|---|---|
| $k$ | $\|x_{k+1} - x_k\|$ | $\|\nabla f(x_k)\|$ | $f(x_k)$ |
| 1 | 0.11370492566124345 | 78.66764541184953 | 6.804582697895967 |
| 2 | 0.038411936236254655 | 33.88616565458079 | 4.654327960200474 |
| 3 | 0.016545979323525865 | 15.7276567344156 | 4.24497169610671 |
| 1503 | 2.1725077169e-08 | 1.68909611e-06 | 3.2771351108e-12 |
| 1504 | 1.319606332e-08 | 3.68457568e-06 | 3.2620794e-12 |
| 1505 | 3.598218465e-09 | 1.613799505e-06 | 3.2540636e-12 |

Finalmente para el algoritmo de aproximación del Hessiano se obtuvieron los siguientes resultados

| Aproximación por Hessiano | | | |
|---|---|---|---|
| $k$ | $\|x_{k+1} - x_k\|$ | $\|\nabla f(x_k)\|$ | $f(x_k)$ |
| 1 | 0.16573671209894808 | 18.919710809780096 | 4.294928723331757 |
| 2 | 0.017695347486938958 | 1.7730718368647735 | 4.126997544944013 |
| 3 | 9.14312943724134 | 33582.01416023042 | 27143.830343361868 |
| 17998 | 2.556928916417568e-07 | 0.00010636586329261558 | 3.1473674130381884e-09 |
| 17999 | 1.3647887156115157e-07 | 5.677393266328426e-05 | 3.140109067380431e-09 |
| 18000 | 2.5510312799878855e-07 | 0.00010612055173286842 | 3.1328674651128993e-09 |

De los resultados anteriormente mostrados vemos que el método que utiliza backtracking para encontrar el mejor tamaño de paso logró hacer que el algoritmo terminara en un número menor de iteraciones comparado con los otros dos métodos. Así mismo vemos que la precicisión con la que encontró el $x_k$ es mayor a la arrojada por los otros dos métodos.

**Curvas de nivel de los algoritmos**

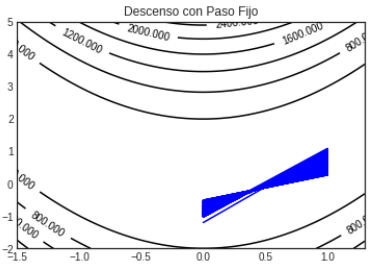


Figura 1: Curvas de nivel con tamaño de paso fijo

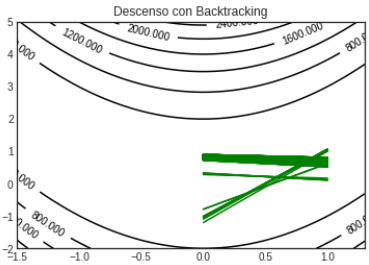


Figura 2: Curvas de nivel con tamaño de paso generado por backtracking

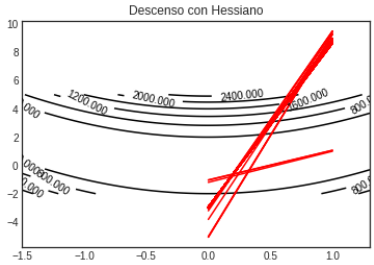


Figura 3: Curvas de nivel con tamaño de paso generado por aproximación de hessiano

## Caso 2: Rosenbruck con n=100

La función Rosenbruck para n=100 está dada de la siguiente forma.

$$f(x) = \sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$$

Las condiciones iniciales de la función son $x_0 = [-1{,}2, 1, ..., -1{,}2, 1]$ y las tolerancias son las mismas que en la función anterior.

Los resultados obtenidos con tamaño de paso fijo se muestran a continuación

| Tamaño de paso fijo | | | |
|---|---|---|---|
| $k$ | $\|x_{k+1} - x_k\|$ | $\|\nabla f(x_k)\|$ | $f(x_k)$ |
| 1 | 0.11252478482538858 | 897.946032794079 | 418.8738128328874 |
| 2 | 0.08979460327940793 | 738.4417002511012 | 345.5618157008769 |
| 3 | 0.0738441700251101 | 621.8359279582658 | 295.42772761822107 |
| 29998 | 1.0483951589659278e-05 | 0.10483301084690505 | 3.996138372080148 |
| 29999 | 1.048330108470184e-05 | 0.10482650628358152 | 3.9961372731182268 |
| 30000 | 1.0482650628412568e-05 | 0.10482000220633723 | 3.996136174292675 |

Para el algoritmo usando backtracking se tienen los siguientes resultados.

| Backtracking | | | |
|---|---|---|---|
| $k$ | $\|x_{k+1} - x_k\|$ | $\|\nabla f(x_k)\|$ | $f(x_k)$ |
| 1 | 0.63358239039187 | 269.28228011943963 | 165.50945061917548 |
| 2 | 0.41343697279214375 | 174.75889703636784 | 103.72373523602288 |
| 3 | 0.24866680665237462 | 153.95990175466116 | 82.1670748854852 |
| 19709 | 1.0202923511168132e-08 | 9.154348702680508e-06 | 3.9866238543473647 |
| 19710 | 1.0169684083269306e-08 | 9.157374678387275e-06 | 3.9866238543473127 |
| 19711 | 1.150128202561265e-08 | 1.0302888385863497e-05 | 3.9866238543472745 |

Finalmente para el algoritmo utilizando la aproximación del Hessiano se tiene la siguiente tabla

| Aproximación por Hessiano | | | |
|---|---|---|---|
| $k$ | $\|x_{k+1} - x_k\|$ | $\|\nabla f(x_k)\|$ | $f(x_k)$ |
| 1 | 0.632956629612631 | 268.92035140974536 | 161.33916490168434 |
| 2 | 0.4129660247226544 | 174.6364741918362 | 99.69820520822034 |
| 3 | 0.24862541464061205 | 153.9192461670668 | 78.15961995653743 |
| 21211 | 4.766596205714067e-09 | 4.291567130293515e-06 | 9.034456905463493e-12 |
| 21212 | 4.66194446959001e-09 | 4.197344736142935e-06 | 9.02445338172454e-12 |
| 21213 | 4.761318188584773e-09 | 4.28681503240123e-06 | 9.01446093421208e-12 |

Podemos notar que ninguno de los tres algortimos presentados en este trabajo convergió para el punto $x_0$ dado.

## Caso 3: Función Wood

La función que vamos a optimizar, conocida como Wood function se define de la siguiente manera.

$$f(x) = 100(x_1^2 - x - 2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10{,}1((x_2 - 1)^2 + (x_4 - 1)^2) + 19{,}8(x_2 - 1)(x_4 - 1)$$

Cuyo gradiente está dado de la siguiente forma

$$\nabla f(x) = \begin{bmatrix} 2(200x_1^3 - 200x_1x_2 + x_1 - 1) \\ -200x_1^2 + 220{,}2x_2 + 19{,}8x_4 - 40 \\ 2(180x_3^3 - 180x_4x_3 + x_3 - 1) \\ 19{,}8x_2 - 180x_3^2 + 200{,}2x_4 - 40 \end{bmatrix}$$

La condición inicial $x_0$ para esta función fue el vector $x_0 = [-3, -1, -3, -1]$ y los resultados obtenidos se muestran a continuación en la siguientes tablas.

A continuación se muestra la tabla con los resultados para la longitud de paso fijo del orden $1 \times 10^{-4}$

| Tamaño de paso fijo | | | |
|---|---|---|---|
| $k$ | $\|x_{k+1} - x_k\|$ | $\|\nabla f(x_k)\|$ | $f(x_k)$ |
| 1 | 1.6397125601763256 | 4435.165843654825 | 3588.344924791783 |
| 2 | 0.4435165843654824 | 2797.587580066072 | 2004.6626352085161 |
| 3 | 0.27975875800660727 | 2022.6896094224687 | 1334.7748274103033 |
| 29998 | 9.05136596911706e-08 | 0.0009051474208247855 | 7.876963739086229 |
| 29999 | 9.051474212001111e-08 | 0.0009051582446540985 | 7.876963739004301 |
| 30000 | 9.051582451288839e-08 | 0.0009051690686021669 | 7.876963738922369 |

A continuación se presenta la tabla de los resultados usando backtracking

| Backtracking | | | |
|---|---|---|---|
| $k$ | $\|x_{k+1} - x_k\|$ | $\|\nabla f(x_k)\|$ | $f(x_k)$ |
| 1 | 2.001602246308991 | 3063.3314611729174 | 2261.4021394065912 |
| 2 | 0.7478836575129193 | 1229.333812343345 | 734.8549252071826 |
| 3 | 0.6002606505582739 | 493.24864415615053 | 245.66521802585007 |
| 9831 | 9.437166176126246e-09 | 2.5568447432772085e-06 | 3.725667788348692e-12 |
| 9832 | 9.987674763229839e-09 | 3.5926484384889873e-06 | 3.7082899830601175e-12 |
| 9833 | 3.5084457338675198e-09 | 2.329057461883627e-06 | 3.6989588872746284e-12 |

Finalmente presentamos la tabla con los resultados utilizando el tamaño de paso dado por la aproximación del Hessiano.

| Aproximación por Hessiano | | | |
|---|---|---|---|
| $k$ | $\|x_{k+1} - x_k\|$ | $\|\nabla f(x_k)\|$ | $f(x_k)$ |
| 1 | 2.067147297814026 | 2853.09391459996 | 2071.5526465873386 |
| 2 | 0.9983126588160758 | 778.5193293205613 | 430.16260075998593 |
| 3 | 0.7403540051285061 | 217.6892332704479 | 99.77456985549541 |
| 14018 | 5.761970687306518e-09 | 2.835132755739948e-06 | 3.3637138078212708e-12 |
| 14019 | 7.0731451270774155e-09 | 3.4802859105296847e-06 | 3.3536871551679894e-12 |
| 14020 | 5.7447952356917495e-09 | 2.8266816273952357e-06 | 3.3436903900529295e-12 |

Resulta interesante observar de lo anterior que el algoritmo utilizando un tamaño de paso fijo encontró otro punto diferente al que se tenía como el argumento minimizador. Los otros dos algoritmos funcionaron correctamente siendo el algoritmo de Backtracking el que menos iteraciones tuvo que realizar.

# Conclusiones

De los resultados obtenidos en los experimentos realizados con el algoritmo de gradiente descendente se puede concluir que la implementación con método de paso fijo resultó ser la que peores resultados arrojó de los algoritmos anteriores, lo anterior debido a que dicho método solo cumple la condición de Armijo o condición de suficiente descenso la cual como se presentó en la introducción no garantiza un comportamiento razonable del algoritmo. Así mismo para el caso de la función Rosenbruck para n=100, ninguno de los dos algoritmos convergió para el punto inicial $x_0$ dado lo cual nos demuestra que la convergencia de los algoritmos de máximo descenso dependen del punto inicial que se le pase por parámetro. Para comprobar esto se tomó un punto inicial lejano al óptimo en el caso de la función Rosenbruck para n=2 y se vió que el algoritmo diverge y jamás se encuentra el argumento que minimiza dicha función. Por otro lado al tomar el punto $x_0 = [1, -1{,}2, 1, ....., 1, -1{,}2, 1]$ en el caso de la función Rosenbruck con n=100 se vió que los tres algoritmos convergen al punto $x^* = [1, 1, 1, ..., 1, 1]$. Así mismo de la función Wood, podemos notar que el tamaño de paso dado ocasionó que el algoritmo no convergiera al valor deseado por lo que la elección de dicho tamaño de paso también tiene una implicación en la convergencia del algoritmo.

# Referencias

[1] Nocedal Jorge, *Numerical Optimization*, Segunda edición, Springer, 2006