



Presentación de un algoritmo para la obtención de ceros de funciones reales no lineales de varias variables reales

| Presentation of an algorithm for obtaining zeros of non-linear real functions of several real variables |

 Manuel Vázquez Mourazos
manuelvazquezmourazos@gmail.com
IES Arcebispo Xelmírez I
Melide, Coruña, España

Recibido: 15 enero 2022

Aceptado: 15 junio 2022

Resumen: El método de Newton-Raphson permite calcular raíces de una función real de variable real. Además, existe una extensión de este método para calcular raíces de una función $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$. En este artículo, partiendo de la interpretación geométrica del método de Newton-Raphson, se presentará una ampliación del mismo que permite calcular raíces de una función real de varias variables reales, $f : \mathbb{R}^n \rightarrow \mathbb{R}$. Así, el objetivo de este artículo es presentar una exposición didáctica que permita deducir la ampliación del método de Newton-Raphson de forma sencilla.

Además, una vez finalizada la deducción del algoritmo, se expondrá una posible programación del mismo en MatLab. Con el código que se expondrá, se hará una serie de ejecuciones del mismo para comprobar su funcionamiento y se analizarán los resultados obtenidos. Finalmente, se concluirá exponiendo los aspectos negativos del nuevo método desarrollado y algunas líneas de investigación que contribuirían a mejorar y completar el estudio realizado en este artículo.

Palabras Clave: método de Newton-Raphson, cero de una función, función real de varias variables reales, función no lineal.

Abstract: The Newton-Raphson method allows to calculate roots of a real function of real variable. Also, there is an extension of this method to compute roots of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$. In this article, starting from the geometric interpretation of the Newton-Raphson method, an extension of it will be presented that allows calculating roots of a real function of several real variables, $f : \mathbb{R}^n \rightarrow \mathbb{R}$. Thus, the objective of this article is to present a didactic exposition that allows to deduce the extension of the Newton-Raphson method in a simple way.

Furthermore, once the algorithm has been deduced, a possible programming of the algorithm will be exposed in MatLab. With the code that will be exposed, a series of executions will be made to verify its operation and the results obtained will be analyzed. Finally, it will conclude by exposing the negative aspects of the new method developed and some lines of research that would contribute to improve and complete the study carried out in this article.

Keywords: Newton-Raphson method, root of a function, real function of several real variables, non-linear function.

1. Introducción

El método de Newton-Raphson fue presentado por Isaac Newton en su obra *De analysi per aequationes numero terminorum infinitas* (publicada en 1711 por William Jones) y por Joseph Raphson en su obra *Aequationum Universalis* (publicada en 1691). Este método permite aproximar raíces de funciones reales de variable real mediante un proceso iterativo que genera una sucesión que converge al valor de una raíz.

Para tener una visión histórica amplia acerca de la construcción del algoritmo que hoy en día se conoce, se puede consultar [9], ya que la visión de Newton y Raphson mostraban ciertas diferencias que se unificaron posteriormente. Hasta el momento de Newton-Raphson es estudiado en los grados relacionados con las matemáticas, como se puede apreciar en manuales universitarios como [3] o [8]. Este hecho es debido a que su deducción e interpretación se pueden obtener empleando diferentes conceptos matemáticos como la fórmula de Taylor o la interpretación geométrica del concepto de derivada.

Este artículo pretende exponer de forma didáctica una extensión del método de Newton-Raphson al caso de funciones $f : \mathbb{R}^n \rightarrow \mathbb{R}$. Para ello se utilizará la interpretación geométrica del método, extendiéndola al caso particular en el que el dominio de la función tenga dos dimensiones. En este caso, la superficie generada por la función se podrá aproximar por el plano tangente a la superficie en un punto dado, lo que permitirá aproximar el cero de la función por uno alguno de los puntos de corte del plano tangente con el plano nulo. Así, una vez que se haya planteado el método para el caso particular en el que $f : \mathbb{R}^2 \rightarrow \mathbb{R}$, se podrá generalizar el método deducido al caso de funciones $f : \mathbb{R}^n \rightarrow \mathbb{R}$.

Una vez expuesto el método con su correspondiente algoritmo, se tratarán los aspectos computacionales del mismo, planteando su programación en MatLab. Luego, empleando el código que se expondrá, se realizarán varias ejecuciones de este con diferentes funciones que permitirán testear el algoritmo y su funcionamiento. De este modo se podrá constatar el funcionamiento del método desarrollado.

Notación 1

Se harán una serie de aclaraciones con el objetivo de exponer la notación que se utilizará en lo sucesivo:

1. A lo largo del artículo se empleará de forma recurrente escalares y vectores. Para diferenciarlos, los vectores se representarán con una letra minúscula en negrita (por ejemplo, \mathbf{v} sería un vector), mientras que los escalares se representarán por una letra cursiva (por ejemplo, v sería un escalar).
2. Los vectores se considerarán vectores columna en todo caso.
3. El empleo de los vectores será muy reiterado. Por ello, para denotar la componente i -ésima de un vector \mathbf{v} , se empleará \mathbf{v}_i .
4. En una sucesión de vectores, se denotará al k -ésimo vector como $\mathbf{v}^{(k)}$.
5. Dada una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$, se denotará por $\nabla f(\mathbf{x})$ al valor del gradiente de la función f evaluado en \mathbf{x} .
6. Dada una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$, se denotará por $f_{x_i}(\mathbf{x})$ a la derivada parcial de f respecto de la componente i -ésima o, lo que es lo mismo, a la componente i -ésima del gradiente de la función.

2. Revisión del método de Newton-Raphson

La naturaleza de este método iterativo radica en aproximar, en cada k -iteración, la raíz de la ecuación por el punto en el que corta la recta tangente a $f(x)$, en $(x_k, f(x_k))$, al eje OX. Por consiguiente, teniendo en cuenta que la ecuación de la recta tangente a la gráfica de $f(x)$ en el punto $(x_k, f(x_k))$ es

$$y = f(x_k) + f'(x_k)(x - x_k), \quad (1)$$

tomando $y = 0$, y despejando el valor de x , se obtiene que el nuevo iterante viene dado por

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \quad (2)$$

La deducción de este método también se podría obtener utilizando el polinomio de Taylor de grado 1.

Teorema 1 Teorema de Taylor

Sea $k \geq 1$ un entero y sea $f : \mathbb{R} \rightarrow \mathbb{R}$ una función diferenciable k veces en un punto $a \in \mathbb{R}$. Entonces existe una función $h_k : \mathbb{R} \rightarrow \mathbb{R}$ tal que

$$f(x) = f(a) + f'(a)(x - a) + \frac{f''(a)}{2!}(x - a)^2 + \dots + \frac{f^{(k)}(a)}{k!}(x - a)^k + h_k(x)(x - a)^k,$$

$$\text{con } \lim_{x \rightarrow a} h_k(x) = 0.$$

Demostración. Se puede consultar en [1, Th. 6.4.1]. □

Aplicando el Teorema 1, dada una función $f : \mathbb{R} \rightarrow \mathbb{R}$, se puede aproximar la función en el entorno de un punto cualquiera x_k por la expresión

$$f(x) \approx f(x_k) + f'(x_k)(x - x_k). \quad (3)$$

De esta forma, teniendo en cuenta que el objetivo es la búsqueda de una raíz de la ecuación, se tomaría como nuevo iterante x_{k+1} , aquel que cumpla que $f(x_{k+1}) = 0$. De este modo, tomando la aproximación (3) como una igualdad y despejando, se obtiene que

$$0 = f(x_k) + f'(x_k)(x_{k+1} - x_k) \Rightarrow x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \quad (4)$$

Así, se vuelve a obtener la misma expresión empleado el Teorema de Taylor.

Finalmente, para visualizar la interpretación geométrica basada en el concepto de la tangente, se puede observar la Figura 1. En ella se presenta esta interpretación en una k -iteración cualquiera del método para el caso particular de una función.

Con estas premisas, a continuación se introduce el algoritmo 1 que resulta de aplicar el método de Newton-Raphson.

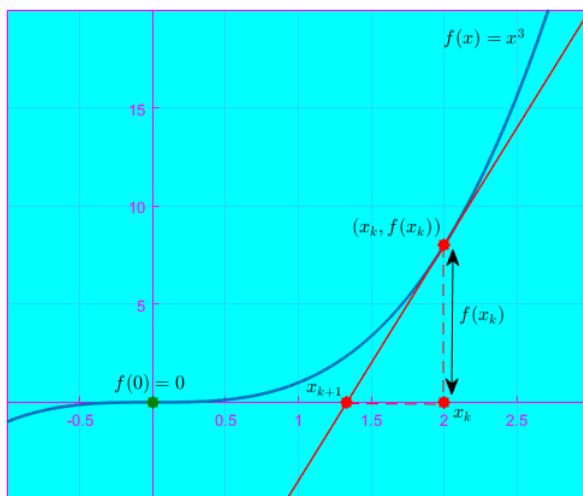


Figura 1: Representación gráfica de una iteración del método de Newton-Raphson

Algoritmo 1 Método de Newton-Raphson

Se toma un iterante inicial x_0 y un valor de tolerancia, tol , para proceder con el siguiente proceso iterativo:

- Se calcula $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$.
- Si $|x_{k+1} - x_k| < tol$ o $\left| \frac{x_{k+1} - x_k}{x_k} \right| < tol$, entonces se finaliza y se toma x_{k+1} como solución.
En caso contrario se repite el proceso.

Para comprobar el funcionamiento del Algoritmo 1 se puede consultar, entre otras muchas referencias, [7, secc. 6.2], donde se puede observar alguna ejecución del método y el error que se va cometiendo en cada iteración. Además, en dicha referencia se podrán observar algunas de las problemáticas generales que presenta el algoritmo. En términos generales es importante resaltar su gran empleo debido a su velocidad de convergencia, pero esta puede no ser tal en algunas funciones particulares o también puede no estar definido el método si en alguna iteración la derivada de la función es nula en el iterante considerado.

Por último, se resalta que el método de Newton-Raphson es extensible a la resolución de un sistema de ecuaciones, esto es, a la obtención de raíces de una función $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$. Como este hecho no es el objeto de estudio de este artículo, se señala la referencia [6] como recurso para conocer los detalles acerca de esta cuestión.

3. Deducción del método de Newton-Raphson para aproximar ceros de una función escalar.

El objetivo de estudio de este artículo es el de realizar una analogía del método de Newton-Raphson con funciones reales de varias variables reales. Para ello se comenzará introduciendo la Definición 1 que explica el tipo de funciones de estudio.

Definición 1

Se denomina *función real de varias variables reales* [4, Def. 1.2] a cualquier función $f : D \subset \mathbb{R}^n \rightarrow \mathbb{R}$, con $n \geq 2$. El conjunto D se llama *dominio*, y el conjunto de todos los valores que toma la función se denomina *imagen o recorrido*.

En este caso, el objetivo del método es el de aproximar un vector $\mathbf{u} \in D$ tal que $f(\mathbf{u}) = 0$. Para ello se estudiará la superficie que genera la función $f(\mathbf{x})$ en el caso de que $D \subset \mathbb{R}^2$. Así pues, se podrá realizar después una generalización del método para el caso en el que $D \subset \mathbb{R}^n$ con $n \geq 3$.

Empezando de forma análoga al método de Newton-Raphson, y considerando una función con dominio en \mathbb{R}^2 , dado un iterante cualquiera $\mathbf{x}^{(k)}$, se obtiene un punto $(\mathbf{x}_1^{(k)}, \mathbf{x}_2^{(k)}, f(\mathbf{x}^{(k)})) \in \mathbb{R}^3$ que se corresponde con la superficie que genera la representación de la función $f(\mathbf{x})$. En este punto, si $f(\mathbf{x}^{(k)}) > 0$, el objetivo sería encontrar un iterante $\mathbf{x}^{(k+1)}$ que redujese el valor del funcional y lo aproximase a cero; por el contrario, si $f(\mathbf{x}^{(k)}) < 0$, el objetivo sería encontrar un iterante $\mathbf{x}^{(k+1)}$ que aumentase el valor del funcional y lo aproximase a cero. Para ello, dado el iterante de partida, $\mathbf{x}^{(k)}$, este se modificará del siguiente modo: $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \cdot \mathbf{v}^{(k)}$, siendo $\mathbf{v}^{(k)}$ la dirección en la que se desplaza el iterante de partida y α_k la distancia que lo hace en dicha dirección (a la cuál se denominará como *paso*). En este punto, se introducirá el concepto de dirección de descenso que será de ayuda.

Definición 2

Dada una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$ y un punto $\mathbf{u} \in \mathbb{R}^n$, se denominará a un vector $\mathbf{d} \in \mathbb{R}^n$ como dirección de descenso [10, Def. 1.4.3.] si existe un intervalo $(0, \xi)$ tal que,

$$f(\mathbf{u} + \lambda \mathbf{d}) < f(\mathbf{u}); \quad \forall \lambda \in (0, \xi).$$

Teorema 2 Condición suficiente de dirección de descenso

Dada una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$ diferenciable, si una dirección $\mathbf{d} \in \mathbb{R}^n$ tal que $\nabla f(\mathbf{u})^t \mathbf{d} < 0$, entonces \mathbf{d} es una dirección de descenso de la función f en el punto \mathbf{u} .

Demostración. Obtenida de [10, pp. 58-59]. Haciendo un desarrollo de Taylor centrado en \mathbf{u} , se obtiene que,

$$f(\mathbf{u} + \lambda \mathbf{d}) = f(\mathbf{u}) + \lambda \nabla f(\mathbf{u})^t \mathbf{d} + o(\lambda) \Rightarrow \frac{f(\mathbf{u} + \lambda \mathbf{d}) - f(\mathbf{u})}{\lambda} = \nabla f(\mathbf{u})^t \mathbf{d} + \frac{o(\lambda)}{\lambda}. \quad (5)$$

Así, teniendo en cuenta que $\nabla f(\mathbf{u})^t \mathbf{d} < 0$, para un λ suficientemente pequeño se obtiene que $f(\mathbf{u} + \lambda \mathbf{d}) - f(\mathbf{u}) < 0$, de lo que concluye que $f(\mathbf{u} + \lambda \mathbf{d}) < f(\mathbf{u})$. Por consiguiente, se concluye que existe un valor suficientemente pequeño de λ para el que se cumple la definición de dirección de descenso. \square

Observación 1

De la Definición 2 se puede deducir la definición de dirección de ascenso de forma análoga. Así, dada una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$ y un punto \mathbf{u} , se denominará dirección de ascenso a aquella dirección \mathbf{d} que cumpla que

$$f(\mathbf{u} + \lambda \mathbf{d}) > f(\mathbf{u}); \quad \forall \lambda \in (0, \xi). \quad (6)$$

Por otra parte, de forma análoga al Teorema 2, dada una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$ y un punto \mathbf{u} , una condición suficiente para que una dirección \mathbf{d} sea de ascenso en dicho punto, sería que $\nabla f(\mathbf{u})^t \mathbf{d} > 0$. En este caso, la demostración se sigue de la mostrada en dicho Teorema.

Teniendo en cuenta estas premisas, una buena dirección $\mathbf{v}^{(k)}$ sería $\pm \nabla f(\mathbf{x}^{(k)})$ dado que se corresponde con la dirección de máximo descenso o ascenso (dependiendo del signo). Así, si $f(\mathbf{x}^{(k)}) > 0$, se tomaría $\mathbf{v}^{(k)} = -\nabla f(\mathbf{x}^{(k)})$; y si $f(\mathbf{x}^{(k)}) < 0$, se tomaría $\mathbf{v}^{(k)} = \nabla f(\mathbf{x}^{(k)})$.

Ahora bien, una vez que se tiene elegida la dirección que determinará el nuevo iterante, es crucial conocer la distancia α_k que se debe recorrer en dicha dirección. En este punto se utilizará la inclinación del plano tangente respecto del plano nulo. Pues bien, dado que $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{v}^{(k)}$ es un punto del plano $z = 0$, la idea es tomar como α_k el valor que hace que $\mathbf{x}^{(k+1)}$ sea uno de los puntos en los que se interseca el plano $z = 0$ con el plano tangente a la superficie que define $f(\mathbf{x})$ en el punto $(\mathbf{x}_1^{(k)}, \mathbf{x}_2^{(k)}, f(\mathbf{x}^{(k)}))$. Para ello, dado el punto $P = (\mathbf{x}_1^{(k)}, \mathbf{x}_2^{(k)}, f(\mathbf{x}^{(k)}))$, se tiene que el plano tangente a dicho punto en la superficie que define $f(\mathbf{x})$, viene dado por la expresión:

$$\Pi \equiv P + \xi (1, 0, f_{\mathbf{x}_1}(\mathbf{x})) + \lambda (0, 1, f_{\mathbf{x}_2}(\mathbf{x})); \quad \lambda, \xi \in \mathbb{R}. \quad (7)$$

Así pues, el vector normal a la superficie en el punto P , al cual se denotará por \mathbf{n} , vendrá dado por la expresión:

$$\mathbf{n} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ 1 & 0 & f_{\mathbf{x}_1}(\mathbf{x}) \\ 0 & 1 & f_{\mathbf{x}_2}(\mathbf{x}) \end{vmatrix} = (-f_{\mathbf{x}_1}(\mathbf{x}), -f_{\mathbf{x}_2}(\mathbf{x}), 1). \quad (8)$$

Este vector nos resulta de interés dado que, conociendo el ángulo con el que corta al plano $z = 0$, se puede obtener el ángulo con el que corta el plano tangente a este mismo plano. Pues bien, dado que el vector \mathbf{n} es el vector normal al plano Π , el ángulo que forman es $\pi/2$. Ahora, denotando por β el ángulo formado por el vector \mathbf{n} con el plano $z = 0$ y por γ el ángulo formado por el plano Π con el plano $z = 0$, se tiene que $\pi = \pi/2 + \gamma + \beta$, lo que implica que $\gamma = \pi/2 - \beta$. Así pues, teniendo en cuenta que

$$\sin(\beta) = \frac{|\mathbf{n} \cdot (0, 0, 1)|}{|\mathbf{n}| \cdot |(0, 0, 1)|} = \frac{1}{|\mathbf{n}|} \Rightarrow \beta = \arcsin\left(\frac{1}{|\mathbf{n}|}\right), \quad (9)$$

es fácil conocer el valor de γ aplicando trigonometría básica. En la Figura 2 se observa la representación de los ángulos β y γ para una superficie dada.

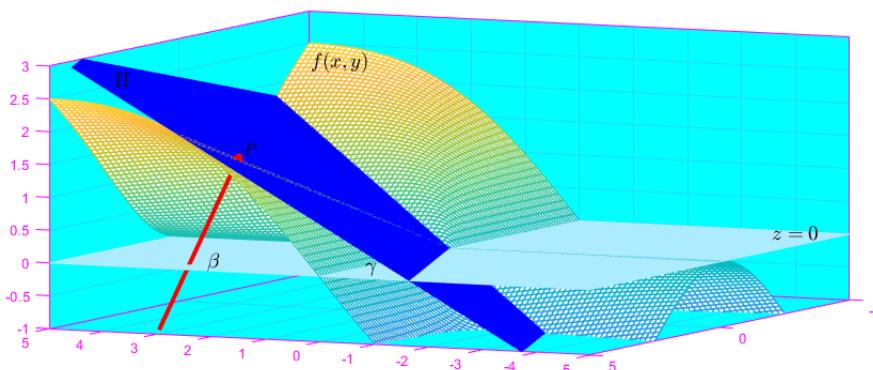
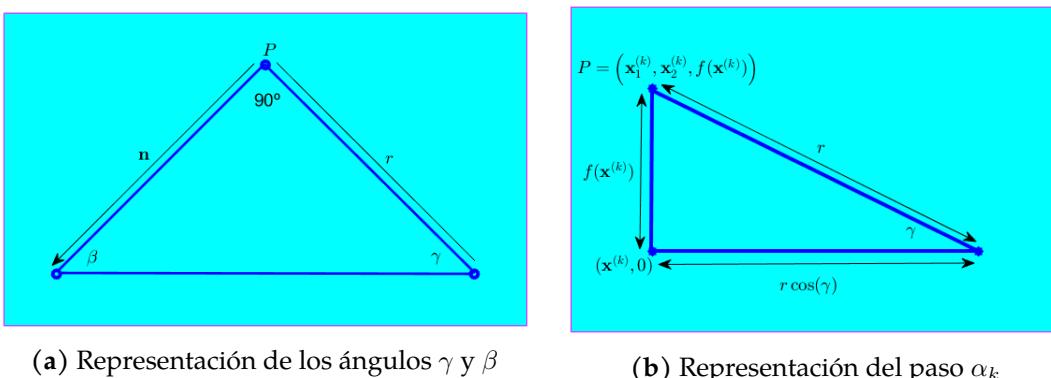


Figura 2: Representación gráfica de los ángulos β y γ

De este modo, de la Figura 2, se puede extraer el triángulo que se observa en la Figura 3a. En dicha imagen se representa el punto P y, a partir de él, el triángulo rectángulo que se obtiene al considerar el vector normal al plano tangente y la recta que representa al plano tangente. De este modo, se puede observar claramente que $\gamma = \pi/2 - \beta$.

Una vez conocido el valor del ángulo γ , se puede hacer una estimación de la distancia α_k que se debe recorrer en la dirección de $\mathbf{v}^{(k)}$, que es el objetivo inicial. Pues bien, dado el iterante de partida $\mathbf{x}^{(k)}$ y el

**Figura 3:** Representaciones de los triángulos que relacionan los ángulos γ y β

punto P de la superficie, se puede establecer el triángulo de la Figura 3b. Para ello, se calcula primero el valor de r utilizando el ángulo γ calculado previamente, pues

$$r = \frac{f(\mathbf{x}^{(k)})}{\sin(\gamma)}. \quad (10)$$

Por consiguiente, una vez calculado el valor de r ya se puede calcular fácilmente el paso α_k como

$$\alpha_k = r \cdot \cos(\gamma). \quad (11)$$

Finalmente, con estos datos quedaría definido el nuevo iterante, $\mathbf{x}^{(k)}$, salvo por un detalle. El vector $\mathbf{v}^{(k)}$ debe ser unitario, pues en caso contrario se estaría multiplicando el valor del paso α_k por la norma del vector, lo que alteraría el valor del nuevo iterante que se desea tomar. Por ello, el nuevo iterante estará definido como

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \frac{\mathbf{v}^{(k)}}{|\mathbf{v}^{(k)}|}. \quad (12)$$

Con estas premisas quedaría definido el proceso iterativo que se desea implementar. En el método de Newton-Raphson se toma como aproximación de la curva que define una función en un punto a la recta tangente y, en este caso, al plano tangente. Ahora bien, una recta corta al eje en un único punto (salvo que sea paralela al eje), pero un plano lo hace en infinitos puntos. Por consiguiente, se toma el punto intersección del plano tangente con el plano nulo en la dirección de máximo descenso (o ascenso) según sea necesario disminuir (o incrementar) el valor de la función en el siguiente iterante. De este modo se puede apreciar una cierta analogía entre el método planteado y el método de Newton-Raphson. A continuación se expondrá el algoritmo 2 que resume los pasos a seguir por el método desarrollado.

Algoritmo 2

Dada una función $f : \mathbb{R}^2 \rightarrow \mathbb{R}$, un iterante inicial $\mathbf{x}^{(0)}$ y una tolerancia $tol > 0$, se procede con el siguiente método iterativo:

PASO 1: Se calcula $\mathbf{n} = (-f_{\mathbf{x}_1}(\mathbf{x}^{(k)}), -f_{\mathbf{x}_2}(\mathbf{x}^{(k)}), 1)$, para luego calcular $\beta = \arcsin\left(\frac{1}{|\mathbf{n}|}\right)$ y, finalmente, $\gamma = \pi/2 - \beta$.

PASO 2: Se calcula $r = \frac{f(\mathbf{x}^{(k)})}{\sin(\gamma)}$ y, con ello, el paso, $\alpha_n = r \cdot \cos(\gamma)$.

PASO 3: Se actualiza el iterante. Si $f(\mathbf{x}^{(k)}) > 0$ se toma $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \cdot \frac{-\nabla f(\mathbf{x}^{(k)})}{|\nabla f(\mathbf{x}^{(k)})|}$ y, en caso

contrario, $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \cdot \frac{\nabla f(\mathbf{x}^{(k)})}{|\nabla f(\mathbf{x}^{(k)})|}$.

PASO 4: Se comprueba el test de tolerancia, si $|f(\mathbf{x}^{(k+1)})| < tol$ se finaliza y se toma como solución $\mathbf{x}^{(k+1)}$, en caso contrario, se vuelve al PASO 1.

Observación 2

El test de parada implementado en el Algoritmo 2 se podría cambiar por otro. Por ejemplo, otro posible test sería el error relativo entre los dos últimos iterantes, esto es:

$$\left| \frac{\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}}{\mathbf{x}^{(k+1)}} \right| < tol$$

Hasta aquí se concluiría la deducción del algoritmo para la búsqueda de raíces de una función real de varias variables reales con dominio en un espacio de dos dimensiones. Ahora bien, una cuestión interesante sería la generalización de este método para una función real de varias variables reales con dominio en un espacio de más de dos dimensiones. Para ello, se hará una generalización del método que se acaba de plantear. Por consiguiente, observando el Algoritmo 2, se puede deducir que la única diferencia radica en que, en lugar de considerar el plano tangente a un punto de la superficie en cada iteración, se considerará el hiperplano tangente. De este modo, modificando el cálculo del vector normal, \mathbf{n} , en el paso 1 del algoritmo, se obtendría la generalización del mismo para el caso de más de dos dimensiones (ver algoritmo 3).

Algoritmo 3

Dada una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$, un iterante inicial $\mathbf{x}^{(0)}$ y una tolerancia $tol > 0$, se procede con el siguiente método iterativo:

PASO 1: Se calcula \mathbf{n} tomando $\mathbf{n}_i = -f_{x_i}(\mathbf{x}^{(k)})$ para $i = 1, \dots, n$ y $\mathbf{n}_{k+1} = 1$, para luego calcular $\beta = \arcsin\left(\frac{1}{|\mathbf{n}|}\right)$ y, finalmente, $\gamma = \pi/2 - \beta$.

PASO 2: Se calcula $r = \frac{f(\mathbf{x}^{(k)})}{\sin(\gamma)}$ y, con ello, el paso, $\alpha_k = r \cdot \cos(\gamma)$.

PASO 3: Se actualiza el iterante. Si $f(\mathbf{x}^{(k)}) > 0$ se toma $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \cdot \frac{-\nabla f(\mathbf{x}^{(k)})}{|\nabla f(\mathbf{x}^{(k)})|}$ y, en caso contrario, $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \cdot \frac{\nabla f(\mathbf{x}^{(k)})}{|\nabla f(\mathbf{x}^{(k)})|}$.

PASO 4: Se comprueba el test de tolerancia, si $|f(\mathbf{x}^{(k+1)})| < tol$ se finaliza y se toma como solución $\mathbf{x}^{(k+1)}$, en caso contrario, se vuelve al PASO 1.

3.1. Relación del Algoritmo 3 con el método de Newton-Raphson

Lo que se ha visto hasta el momento es la deducción y la interpretación geométrica del Algoritmo 3 que permite obtener ceros de una función real de varias variables reales. En dicha deducción, se ha comentado que se trata de una extensión o de una analogía del método de Newton-Raphson y, en esta sección, se mostrará el motivo de dicha afirmación. Para ello se considerará una función real de una variable real y se comprobará que el Algoritmo 3 es, en realidad, el método de Newton-Raphson para el caso de una función $f : \mathbb{R} \rightarrow \mathbb{R}$.

En el caso de una función real de variable real, dado un iterante x_k , el método de Newton-Raphson toma como nuevo iterante x_{k+1} el punto intersección entre el eje OX y la recta tangente a la curva en el punto $(x_k, f(x_k))$. Esta situación aparece representada en la Figura 4.

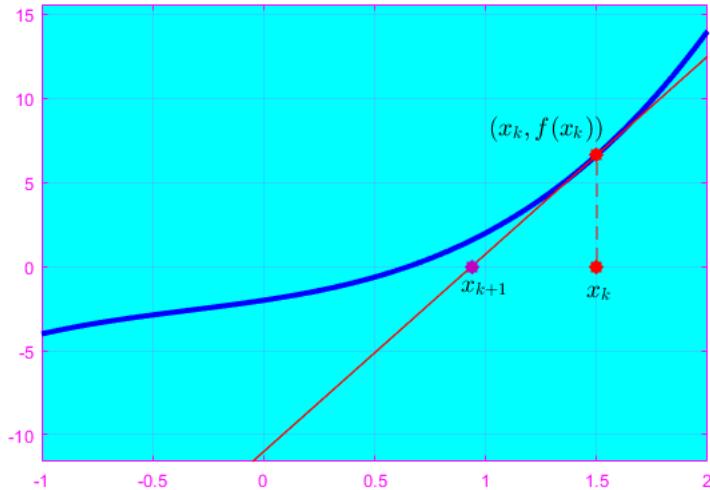


Figura 4: Representación geométrica del método de Newton-Raphson

Entonces, aprovechando la Figura 4, se representará sobre ella los ángulos γ y β si se emplease el método planteado en esta sección. El resultado se puede apreciar en la Figura 5 (a) y, seguidamente, en la Figura 5 (b) se puede observar la representación del valor del paso $\alpha_k = r \cos(\gamma)$.

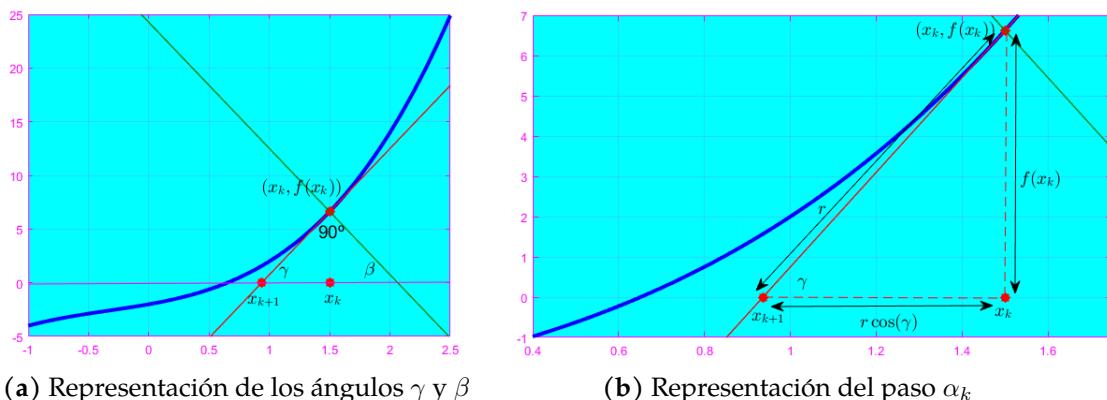


Figura 5: Representación del método desarrollado frente al método de Newton-Raphson

Ahora bien, dado que se trata de un problema unidimensional, ya que $x_k \in \mathbb{R}$ en cada k -iteración, sólo existen dos posibles direcciones en las que moverse, a la derecha o a la izquierda del iterante x_k . En este caso, dado que el gradiente de la función se corresponde con el valor de la derivada, $f'(x_k)$, y teniendo en cuenta que $f(x_k) > 0$, se obtiene que $v_k = -f'(x_k)$, de lo que se concluye que

$$x_{k+1} = x_k + r \cos(\gamma) \frac{-f'(x_k)}{|f'(x_k)|} = x_k + r \cos(\gamma)(-1) = x_k - r \cos(\gamma). \quad (13)$$

De este modo, se deduce que x_{k+1} se corresponde con el resultado de desplazarse una distancia de $r \cos(\gamma)$ unidades a la izquierda del iterante x_k , lo cuál hace coincidir el nuevo iterante con el mismo iterante que se obtendría aplicando el método de Newton-Raphson. Así pues, para el caso en el que se considere una función $f : \mathbb{R} \rightarrow \mathbb{R}$ el método desarrollado es el mismo que el método de Newton-Raphson y, para una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$, se podría considerar como una ampliación o desarrollo del mismo.

Por otra parte, en lo referente a los aspectos computacionales, no existe una gran diferencia entre los

algoritmos 2 y 3. Por ello, a continuación se mostrará una implementación del algoritmo genérico (algoritmo 3) en MatLab.

Código 1: Función que implementa el Algoritmo 3

```

1 function [x,fx,k]=metodo_algoritmo_1.3(x0,f,f1,tol)
2 % Parámetros de entrada:
3 %      x0: iterante inicial aportado (debe ser un vector columna).
4 %      f: función de estudio (debe estar en formato simbólico).
5 %      f1: derivada de la función de estudio (debe estar en formato simbólico).
6 %      tol: nivel de tolerancia del algoritmo.
7 % Parámetros de salida:
8 %      x: solución numérica obtenida.
9 %      fx: valor de la función en la solución numérica obtenida.
10 %      k: número de iteraciones empleadas.
11
12 k=0;
13 while (norm(f(x0),2) > tol)
14     % Paso 1:
15     n=[-f1(x0);1]; beta=asin(1/norm(n,2)); gamma=pi/2-beta;
16     % Paso 2:
17     r=f(x0)/sin(gamma);
18     alpha=r*cos(gamma);
19     % Paso 3:
20     if (f(x0) > 0)
21         v=-f1(x0);
22     else
23         v=f1(x0);
24     end
25     x0=x0+alpha*(v/norm(v,2)); % Actualización del iterante.
26     k=k+1;                      % Actualización de la iteración.
27 end
28
29 x=x0;
30 fx=f(x0);
31
32 end

```

En el Código 1 se muestra la programación de una función con argumentos de entrada: $x0$ (el iterante inicial), f (la función a considerar), $f1$ (una función que devuelve el gradiente de la función) y tol (un valor de tolerancia). Con base en estos argumentos, se sigue el proceso detallado en el Algoritmo 3, y se devuelven como argumentos de salida: x (la raíz numérica encontrada por el algoritmo), fx (el valor de la función en la solución aportada) y k (las iteraciones empleadas por el método).

Observación 3

Es importante hacer notar que el método diverge en caso de que se obtenga un iterante en el que el gradiente de la función fuese el vector nulo. En un iterante con estas características se estaría ante un punto que se correspondería con un punto de ensilladura de la función, por lo que el hiperplano tangente a la superficie generada por la función en dicho punto sería paralelo al hiperplano nulo. De este modo se obtendría que $\gamma = 0$ y, por consiguiente, el valor de r no estaría definido. En consecuencia, se podría añadir al Código 1 una línea que mostrase un mensaje de error en caso de que se obtuviese un iterante con gradiente nulo.

4. Pruebas y ejecuciones

En esta sección se pondrá a prueba el Algoritmo 3 mediante la ejecución del Código 1 en diferentes casos. A continuación se expondrán los test empleados y los resultados obtenidos.

- **Test 1:** Se considera la función $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ definida por:

$$f(x, y) = x^2 + y^2 \quad (14)$$

cuya solución es conocida, pues $f(0, 0) = 0$ es su única solución. Para ello, se parte del iterante inicial $\mathbf{x}^{(0)} = (2 \ 5)^t$ con una tolerancia $tol = 10^{-5}$.

Tabla 1: Resultados obtenidos para la función (14).

Iteración (k)	Iterante ($\mathbf{x}^{(k)}$)	Valor de la función ($f(\mathbf{x}^{(k)})$)
0	$(2 \ 5)^t$	29
1	$(1 \ 2.5)^t$	7.25
\vdots	\vdots	\vdots
10	$(1.95 \cdot 10^{-3} \ 4.88 \cdot 10^{-3})^t$	$2.76 \cdot 10^{-5}$
11	$(9.76 \cdot 10^{-4} \ 2.44 \cdot 10^{-3})^t$	$6.91 \cdot 10^{-6}$

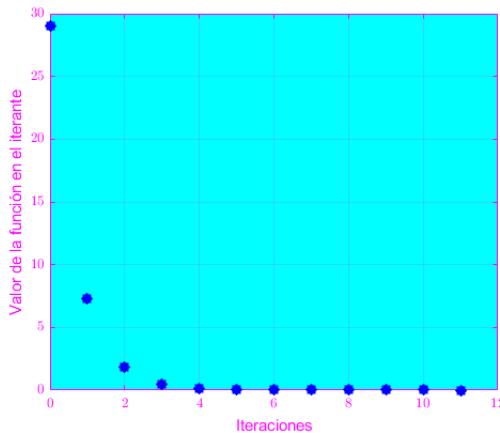


Figura 6: Representación de la evaluación del valor de la función (14) en las iteraciones

Ante los resultados obtenidos (ver tabla 1 y figura 6) se puede concluir que el algoritmo ha funcionado de una forma correcta. En pocas iteraciones se reduce notablemente el valor de la función y, finalmente, en las últimas iteraciones se afina el valor de los iterantes para reducir el valor de la función hasta la tolerancia requerida. Por lo que el algoritmo ha convergido.

- **Test 2:** Se considera la función $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ definida por:

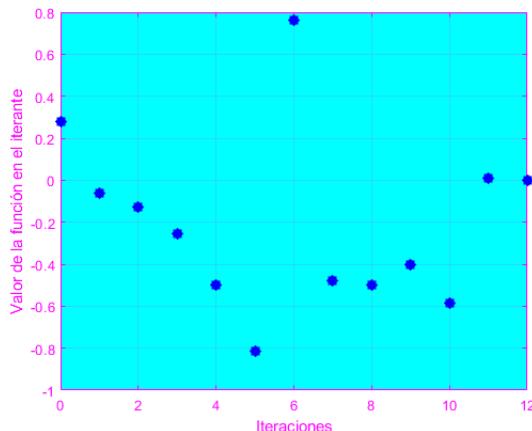
$$f(x, y) = \frac{xy - 3}{x^2 + y^2 - 4}, \quad (15)$$

la cual tiene infinitas soluciones que se corresponden con los elementos del conjunto $S = \{(x, y) \in \mathbb{R}^2 : xy = 3\}$. Para ello, se parte del iterante inicial $\mathbf{x}^{(0)} = (2 \ 5)^t$ con una tolerancia $tol = 10^{-5}$.

En este caso se obtiene una mayor variabilidad en el valor de la función en los diferentes iterantes (ver tabla 2 y figura 7). Estos valores se comprenden dentro de un rango pequeño, pero la forma de la superficie que genera la función produce la variabilidad que se muestra. Además, esta

Tabla 2: Resultados obtenidos para la función (15).

Iteración (k)	Iterante ($\mathbf{x}^{(k)}$)	Valor de la función ($f(\mathbf{x}^{(k)})$)
0	$(2 \ 5)^t$	0.28
1	$(0.27 \ 5.36)^t$	-0.06
\vdots	\vdots	\vdots
11	$(-0.36 \ -14.93)^t$	0.01
12	$(-0.2 \ -14.93)^t$	$-6.31 \cdot 10^{-6}$

**Figura 7:** Representación de la evaluación del valor de la función (15) en las iteraciones

prueba ejemplifica que el valor de la función no se aproxima siempre al cero en cada iteración. Este hecho radica en que la dirección de máximo descenso (o ascenso) asegura descender (o aumentar) el valor de la función, pero sólo durante un paso α_k concreto. En este sentido, el paso α_k tiene una interpretación geométrica mostrada en el apartado anterior, pero no asegura ese hecho de que se aproxime la función al cero en cada iteración. Sin embargo, como se muestra, el algoritmo termina convergiendo correctamente.

- **Test 3:** Se considera la función $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ definida por:

$$f(x, y, z) = x^2 + y + 8x^2y^5z^3. \quad (16)$$

Para esta función, el vector nulo es la solución trivial. Además, también existen infinitas soluciones que se corresponden con el conjunto $S = \left\{ (x, y, z) \in \mathbb{R}^3 : z = \sqrt[3]{\frac{-x^2 - y}{8x^2y^5}} \text{ con } x, y \in \mathbb{R} \setminus \{0\} \right\}$. Para ello, se parte del iterante inicial $\mathbf{x}^{(0)} = (2 \ 5 \ 3)^t$ con una tolerancia $tol = 10^{-5}$.

Tabla 3: Resultados obtenidos para la función (16).

Iteración (k)	Iterante ($\mathbf{x}^{(k)}$)	Valor de la función ($f(\mathbf{x}^{(k)})$)
0	$(2 \ 5 \ 3)^t$	2700009
1	$(1.66 \ 4.66 \ 2.66)^t$	932673.65
\vdots	\vdots	\vdots
71	$(0.45 \ -0.19 \ 0.69)^t$	0.0029
72	$(0.44 \ -0.2 \ 0.69)^t$	$2.04 \cdot 10^{-6}$

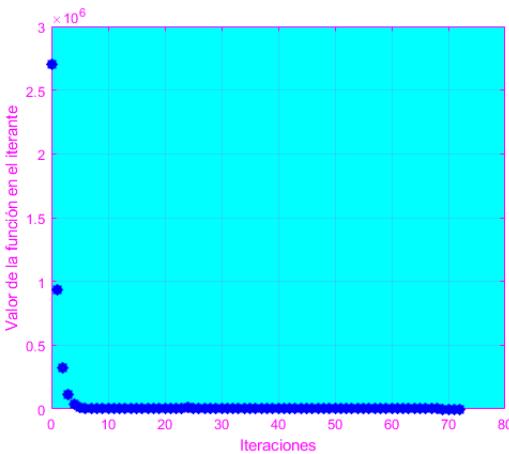


Figura 8: Representación de la evaluación del valor de la función (16) en las iteraciones

En este test se incrementa el valor de la dimensión de la función respecto de las dos pruebas anteriores (ver tabla 3 y figura 8). Como se puede apreciar, el algoritmo permite descender el valor de la función de forma correcta y no se aprecian grandes cambios en la secuencia de las iteraciones. En este caso, un hecho que resalta, es el enorme valor del funcional en el iterante de partida, motivo por el cual las iteraciones necesarias son mucho mayores que en las pruebas anteriores. Sin embargo, el algoritmo vuelve a converger de forma correcta alcanzando la tolerancia requerida.

- **Test 4:** Se considera la función $f : \mathbb{R}^4 \rightarrow \mathbb{R}$ definida por:

$$f(x, y, z, t) = x^2 + y^2 + z^2 + t^2 - 25. \quad (17)$$

En este caso, el conjunto de soluciones vuelve a ser infinito, ya que cualquier punto de la esfera \mathbb{S}^3 sería un cero de la función. Para ello, se parte del iterante inicial $\mathbf{x}^{(0)} = (1 \ 3 \ 1 \ 0)^t$ con una tolerancia $tol = 10^{-5}$.

Tabla 4: Resultados obtenidos para la función (17).

Iteración (k)	Iterante ($\mathbf{x}^{(k)}$)	Valor de la función ($f(\mathbf{x}^{(k)})$)
0	$(1 \ 3 \ 1 \ 0)^t$	-14
1	$(0.36 \ 1.09 \ 0.36 \ 0)^t$	-23.54
\vdots	\vdots	\vdots
6	$(-1.5 \ -4.52 \ -1.5 \ 0)^t$	$2 \cdot 10^{-3}$

En este caso se vuelve a incrementar la dimensión del dominio de la función de estudio. Al igual que en el test 2, los iterantes no se aproximan siempre al cero en cada iteración. Este hecho sucede a partir de la segunda iteración, por lo que el algoritmo converge (ver tabla 4 y figura 9). Sin embargo, este hecho permite hacer hincapié en el hecho de que el algoritmo no acerca el valor del funcional al cero en cada iteración, lo cuál también sucede en el método de Newton-Raphson, ya que se approxima la función por un hiperplano y se toma un punto de este como nuevo iterante. Por consiguiente, en una función que presenta varios cambios de tendencia en el valor que toma, se puede presentar este comportamiento.

Finalmente, una vez realizados estos tests, y como añadido, se hará un breve estudio acerca del número de iteraciones requerido por el método en los casos de los test 1 y 2, que son aquellos cuyas funciones están definidas en \mathbb{R}^2 . Para ello se presenta la Figura 10, en la que se observa un gráfico de

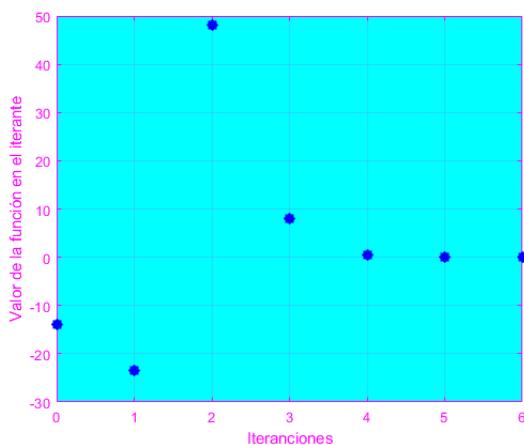


Figura 9: Representación de la evolución del valor de la función (17) en las iteraciones

colores que muestra el coste en iteraciones del Algoritmo 3 en un entorno del punto inicial tomado en la ejecución del método (el punto señalado en el centro de ambos gráficos).

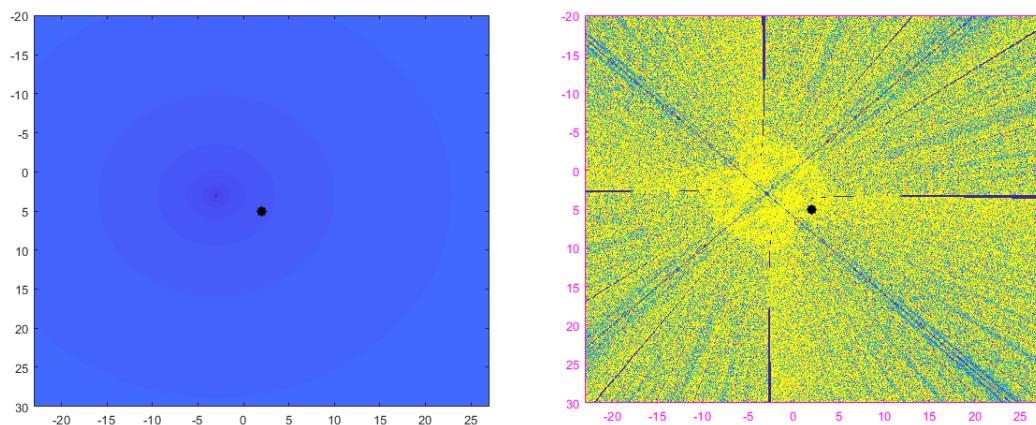


Figura 10: Representación del número de iteraciones del Algoritmo 3 en los Test 1 y 2.

Con base a la Figura 10, se puede observar que el número de iteraciones requerido en el test 1 es gradual a la distancia del punto inicial considerado respecto de la solución única del problema. Sin embargo, en el caso del test 2 no se obtiene un resultado homogéneo. En este caso, la infinidad de soluciones posibles hace que las iteraciones requeridas por el método sean muy dispares en un entorno pequeño.

5. Conclusiones

Con base en el estudio realizado, se pueden establecer las siguientes conclusiones generales acerca del método desarrollado:

1. El Algoritmo 3 permite aproximar raíces numéricas de funciones reales de varias variables reales, $f : \mathbb{R}^n \rightarrow \mathbb{R}$, aplicando una extensión de la interpretación geométrica del método de Newton-Raphson.
2. Para el caso particular en el que se considere una función real de una variable real, $f : \mathbb{R} \rightarrow \mathbb{R}$, el Algoritmo 3 se corresponde con el método de Newton-Raphson.

3. El Algoritmo no está definido en el caso de que alguno de los iterantes $\mathbf{x}^{(k)}$ generados se corresponda con un punto de ensilladura de la función de estudio, $f : \mathbb{R}^n \rightarrow \mathbb{R}$. En este caso el gradiente de la función en dicho punto sería nulo y, en consecuencia, el hiperplano tangente sería paralelo al hiperplano nulo, por lo que no estaría definido el procedimiento seguido por el Algoritmo 3.

Finalmente, como futuro trabajo, sería interesante estudiar una demostración de la convergencia del método, basado en ideas desarrolladas en la literatura. Para dicho fin, se podría basar en los métodos desarrollados en los trabajos [2, Teorema 3] y [5, Teorema 1].

6. Bibliografía

- [1] Bartle, Robert G; Sherbert, Donald R. *Introducción al análisis matemático de una variable*. Limusa-Wiley. 2010.
- [2] Candelario, Giro; Cordero, Alicia; Torregosa, Juan R.; Vassileva, María P. *An optimal and low computational cost fractional Newton-type method for solving nonlinear equations*. Applied Mathematics Letters 124. pp. 107-650 (2022). <https://www.sciencedirect.com/science/article/pii/S0893965921003487>
- [3] Ezquerro Fernández, José Antonio. *Iniciación a los métodos numéricos*. Universidad de La Rioja, Servicio de Publicaciones. 2012.
- [4] Freixas Bosch, Josep. *Las funciones de varias variables*. UOC. 2002.
- [5] Kansal, M.; Cordero, A.; Bhalla, S. et al. *New fourth- and sixth-order classes of iterative methods for solving systems of nonlinear equations and their stability analysis*. Numer Algor 87, 1017?1060 (2021). <https://doi.org/10.1007/s11075-020-00997-4>
- [6] Kelley, C.T. *Solving Nonlinear Equations with Newton's Method*. SIAM. 2003.
- [7] Steven, S.C; Canale, R.P. *Métodos numéricos para ingenieros* McGraw-Hill. 2007.
- [8] Viaño, Juan Manuel *Lecciones de métodos numéricos. Volumen 2: Resolución de ecuaciones numéricas*, Andavira. 1997.
- [9] Ypma, Tjalling J. *Historical development of the Newton-Raphson method* SIAM Review, Vol. 37, No 4., pp.531-551. 1995.
- [10] Sun, Wenyu and Yuan, Ya-Xiang. *Optimization theory and methods. Nonlinear programming*. Springer. 2006.