Métodos Numéricos en Optimización y Resolución de Ecuaciones

Jorge Mauricio Oviedo [1]

Resumen: El presente trabajo tiene por objetivo brindar una exposición clara y exhaustiva de los principales Métodos Numéricos en materia de resolución de Ecuaciones y Optimización. Para llevar a cabo dicha tarea se presenta una revisión teórica de tales tópicos complementada con ejemplos gráficos y algebraicos diseñados para una fácil y didáctica asimilación. Se brindan además rutinas de programación en Mathematica que automatizan la tarea de resolución de dichos problemas y permiten visualizar la evolución de los algoritmos. De esta manera, se logra cumplir el fin de fomentar el uso de tales métodos cuantitativos minimizando el esfuerzo de aprendizaje y resolución.

Palabras clave: Función, Ecuación algebraica, Máximo Global, Máximo Local, Derivada, Matriz Jacobiana

[1] joviedo@eco.unc.edu.ar

1.- Método de Newton para la resolución de Ecuaciones

No siempre es posible resolver exactamente una ecuación; por ejemplo esto ocurre con las ecuaciones $x^5 + x - 3 = 0$ ó $x + sen\, x = 0$.Los métodos numéricos para resolver ecuaciones surgen para salvar los inconvenientes que se presentan cuando no existe una vía algebraica para hallar los valores de una variable que satisfacen una ecuación. También existen situaciones en que si bien existen formulas o vías algebraicas exactas de hallar raíces de una ecuación la complejidad y la extensión de estos exigen una gran labor de cálculos a tal punto que los métodos numéricos son preferidos aun cuando algebraicamente es viable obtener las soluciones. Tal es el caso de polinomios de grado tres o cuatro completos no factorizables y con raíces irracionales (de modo que impidan hallar raíces por el método de Descartes) cuya complejidad y extensión de sus fórmulas[2] exactas para todas sus raíces hacen de los métodos numéricos una alternativa de solución preferible en cuanto a su economía en términos de esfuerzo[3].

Dentro de dichos métodos se encuentra el Método de Newton para hallar los ceros de una ecuación. A continuación se pasa a describir su funcionamiento.

Sea f(x) una función. Una solución de la ecuación f(x)=0 es un número r tal que f(r)=0. La gráfica de y=f(x) pasa por el punto (r,0)como se muestra en figura de abajo.

Si no se conoce r, se hace una conjetura x_1. (la manera de plantear esta conjetura inicial depende de la función particular f). Luego para hacer una conjetura mejor se usa el hecho de un pedazo de una curva diferenciable se parece a una recta; tal recta es su recta tangente en cualquier punto. Se traza entonces la tangente en *(x_1 , $f(x_1)$),* como se muestra en la figura. Si la tangente no es paralela al eje de las abscisas, ésta intercepta el eje x en un punto que se llamará x_2.(Sin duda, si la grafica de f fuera una recta, entonces x_2 sería precisamente r). Por lo tanto su ecuación mediante la fórmula de punto pendiente es:

$$y - f(x_1) = f'(x_1)(x - x_1)$$

La intersección de ésta recta con el eje x, que se llamará x_2, se obtiene estableciendo $y = 0$;

$$0 - f(x_1) = f'(x_1)(x - x_1)$$

al resolver se tiene:

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

Este proceso puede repetirse usando este valor como una nueva conjetura arribándose a este resultado:

$$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)}$$

Este mecanismo se puede repetir tantas veces como se desee obteniéndose una sucesión de estimativos x_1,x_2, x_3,... la cual a menudo se aproxima a r.

La i-ésima iteración será:

[2] Tales expresiones son conocidas como las fórmulas de Cardano y Tartaglia.

[3] Vale destacar que la complejidad de dichas formulas no quitan mérito a su existencia ya que las misma proveen los resultados exactos de las raíces (reales y complejas) del polinomio no en cambio así un método numérico que solo brinda una aproximación. A su vez dichas formulas también permiten obtener soluciones genéricas para el caso de que los coeficientes sean parámetros simbólicos.

$$x_i = x_{i-1} - \frac{f(x_{i-1})}{f'(x_{i-1})} \quad \text{siempre que} \quad f'(x_{i-1}) \neq 0$$

Una manera distinta y equivalente de explicar el desarrollo de la fórmula del algoritmo de Newton consiste en decir que en primer lugar se parte de un punto arbitrario llamado x_1; en ese punto se aproxima la función por medio de un desarrollo en serie de primer orden de la función (que será una recta tangente en x_1); se halla el cero de esa recta y se toma dicho valor como una nueva aproximación. En este nuevo punto se calcula el nuevo desarrollo en serie y el valor que lo anula para repetir nuevamente todo el proceso sucesivamente.

$$f(x_0) \cong f(x_0) + f'(x_0)(x - x_0)$$

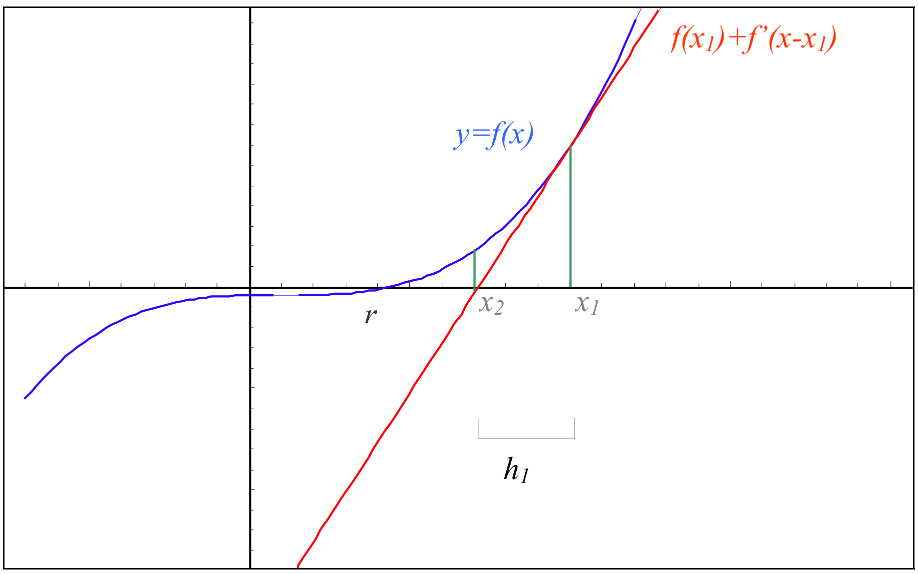
Luego el valor de x que lo anula es

$$f(x_0) + f'(x_0)(x - x_0) = 0 \rightarrow x_2 = x_1 - \frac{f(x_0)}{f'(x_0)}$$

que también puede expresarse como:

$$\boxed{x_2 = x_1 + h_1} \qquad \text{donde}: h_1 = -\frac{f(x_1)}{f'(x_1)}$$

El gráfico a continuación expresa geométricamente lo expuesto mas arriba:



Este proceso iterativo continuará indefinidamente a menos que se imponga un limite a su retroalimentación como podrían ser un numero delimitado de antemano de iteraciones, continuar hasta que la mejora en cada nueva iteración sea menor a un cierto $\varepsilon(|h_i| < \varepsilon)$, o bien que se generen iteraciones hasta que se obtenga una aproximación con un error menor un cierto δ.

A continuación se pasará aplicar el método para resolver la ecuación del siguiente ejercicio:

Calcular los ceros de las siguientes funciones utilizando el algoritmo de ***Newton****.*

x^2 - 40 = 0

$x^2 - 40 = 0$ que no es otra cosa que encontrar los ceros para $f(x) = x^2 - 40$

$x_{i+1} = x_i + h_i$ donde $h_i = -\dfrac{f(x_i)}{f'(x_i)} = -\dfrac{(x_i^2 - 55)}{2x_i}$ que será el algoritmo para este problema

Partiendo del valor arbitrario $x_0 = 10$ tomando como máximo 12 iteraciones, y con un ε =
=10^{-150} y mostrando los resultados con una precisión de 80 dígitos[4] se tiene:

X_1	10.0000000000000000000000000000000000000000000000000000000000000000000000000000000
H_1	-3.0000000000000000000000000000000000000000000000000000000000000000000000000000000
X_2	7.0000000000000000000000000000000000000000000000000000000000000000000000000000000
H_2	-0.64285714285714285714285714285714285714285714285714285714285714285714285714285714
X_3	6.3571428571428571428571428571428571428571428571428571428571428571428571428571429
H_3	-0.032504012841091492776886035313001605136436597110754414125200642054574638844301766
X_4	6.3246388443017656500802568218298555377207062600321027287319422150882825040128411
H_4	-0.000083523413493065507103856635376978935855953440960951321113547744636144462482978333
X_5	6.3245553208882725845731529651944785587848503065911417774108286673436463595503581
H_5	-5.5151392055131931580771687281277792475144170990697943938438391161513754580059969´ 10^{-10}
X_6	6.3245553203367586640218336493867616859720723818397000675038492279592624479352206
H_6	-2.4046560521324618532916389534818140358966177883653039903886809939790972018150477´ 10^{-20}
X_7	6.3245553203367586639977870888654370674391559923048819271448830500756094080313338
H_7	-4.5713654448273429873344490420530752857314565528455965007375866702855870496647405´ 10^{-41}
X_8	6.3245553203367586639977870888654370674391102786504336537150097055851888772784764
H_8	-1.6520831087495954435675413891322179186997801873763535206036666032351326370208797´ 10^{-82}
X_9	6.3245553203367586639977870888654370674391102786504336537150097055851888772784764
H_9	-2.1577632418198205526626200411062550054508138257996570752721460036732494624156111´ 10^{-165}

Ahora se buscarán si existen otros ceros partiendo de un x_0 distinto.

Partiendo del valor arbitrario $x_0 = 15$ tomando como máximo 10 iteraciones, y un ε ==10^{-200} y mostrando los resultados con una precisión de 80 dígitos[5] se tiene:

X_1	-8.0000000000000000000000000000000000000000000000000000000000000000000000000000000
H_1	1.5000000000000000000000000000000000000000000000000000000000000000000000000000000
X_2	-6.5000000000000000000000000000000000000000000000000000000000000000000000000000000
H_2	0.17307692307692307692307692307692307692307692307692307692307692307692307692307692
X_3	-6.3269230769230769230769230769230769230769230769230769230769230769230769230769231
H_3	0.0023673135375263034837502922609305588028992284311433247603460369417816226326864625
X_4	-6.3245557633855506195931727846621463642740238484919335983165770399812953004442366
H_4	4.4304877643733296561423855050433053228690602844397508696362076026522851846249199´ 10^{-7}
X_5	-6.3245553203367741822602071704235958599434915615859051543414900763605350352157182
H_5	1.5518262420081539120579037881603343964386489853456387561837059759392792262229903´ 10^{-14}

[4] Vale la misma aclaración hecha en la nota 5

[5] Idem Nota 6

X_6	- 6.32455532033675866399778708888447528090560995824194076785163661997297319815559588
H_6	1.90382134664996795915071141365982599169450405544526743912243213205725513109099978´ 10^{-29}
X_7	- 6.3245553203367586639977870888654370674391102786504336537150383600560281576015061
H_7	2.8654470839280323029642085402755693631211597952870187069157961949812304584038551´ 10^{-59}
X_8	- 6.3245553203367586639977870888654370674391102786504336537150097055851888772784764
H_8	6.4911970683454077450317098606081127268718498498540050379311714262532551407082728´ 10^{-119}
X_9	- 6.3245553203367586639977870888654370674391102786504336537150097055851888772784764
H_9	3.3311147777147162801174591516926951459311776614889959932680683437324501667442319´ 10^{-238}

En los dos casos, h_9 es menor en valor absoluto que el tolerable (e), por ende el algoritmo se detiene

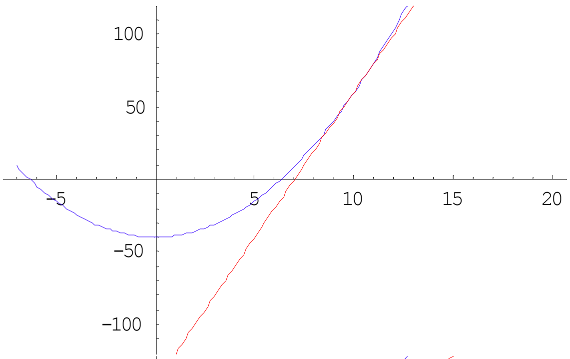
.f(x_9)=
2.7293785982157276048652981011696031314545521228138240106459027872418453223077557´ 10^{-164}
f(x_9)=
1.6350492611402659518404583309289544723673072727332556435263462505763572913508921´ 10^{-98}

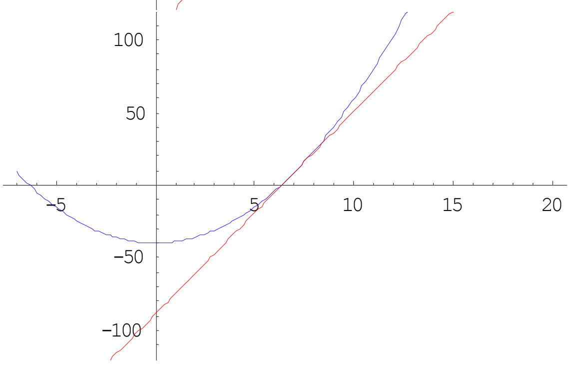
como se observa las evaluaciones en los puntos hallados son casi iguales a cero por lo que los valores hallados son muy buenas aproximaciones para las raíces de esta ecuación.

Sabiendo de antemano que, dada la característica de esta función polinómica, tendrá como máximo tantas raíces reales como el grado de la misma (2), se ha terminado con el procedimiento.

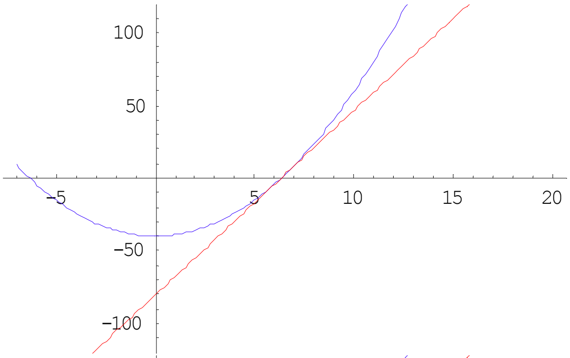
Geométricamente para los primeros cuatro pasos partiendo con x_1 =10, el método luce como sigue:

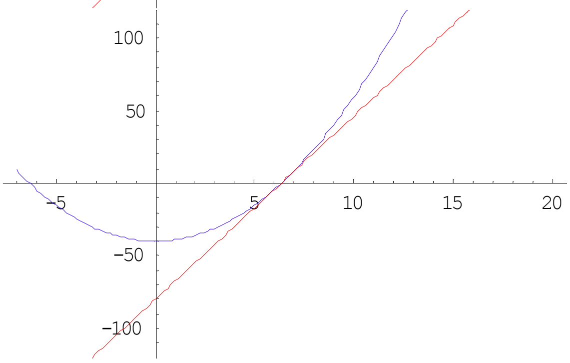
1 2





3 4





Para el caso de sistemas de ecuaciones el algoritmo de Newton toma la siguiente forma:

$\mathbf{x_i} = \mathbf{x_{i-1}} - \mathbf{J}^{-1}(\mathbf{x_{i-1}})\mathbf{F}(\mathbf{x_{i-1}})$ Donde las variables ahora representan vectores (puntos N dimensionales) y J y F representan al Jacobino del Sistema de Ecuaciones y al conjunto de ecuaciones respectivamente.

El método de Newton sirve también para detectar las raices complejas siempre que se definan adecuadamente los valores.

Así como para dar lugar a este método se utilizó un desarrollo en serie de orden uno el lector curioso podría preguntarse por que no aproximar la función por medio de un desarrollo de orden, dos tres, cuatro, etc que sin lugar a dudas será una mejor aproximación. Sin embargo se puede demostrar que dicho esfuerzo algebraico no redunda en un incremento la velocidad de convergencia del método (es decir la rapidez con que se aproxima al valor buscado) ya que cualquiera sea el orden del polinomio utilizado en su aproximación el orden de convergencia del algoritmo es igual a dos (aproximadamente, esto puede interpretarse como que en cada paso la exactitud se duplica)[6].

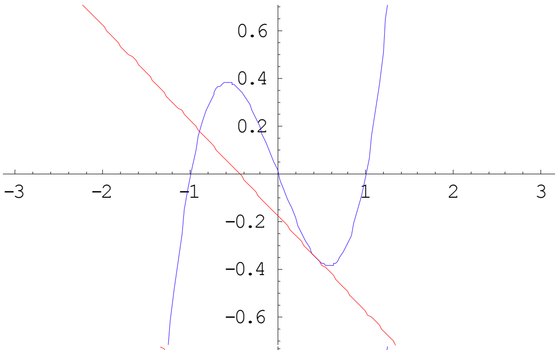
En comparación a otros métodos este es uno de los mas veloces, pero al igual que otros no esta exento de fallas.

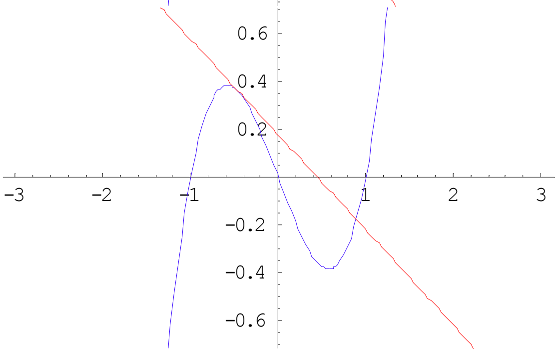
Entre estas cabe mencionar las siguientes:

- podría darse un caso en que el valor de partida o al que se arriba luego de algunas sucesiones, la derivada sea nula es decir es decir la función en ese punto presenta una tangente horizontal y el algoritmo explota al no estar definido en una derivada primera nula.
- En determinadas situaciones cuando se producen cambios bruscos y simétricos de convexidad el algoritmo puede entrar en un circulo vicioso para siempre. Tal es el caso de las funciones cúbicas como $f(x) = x^3 - x$ cuya comportamiento geométrico se da a lucir a continuación

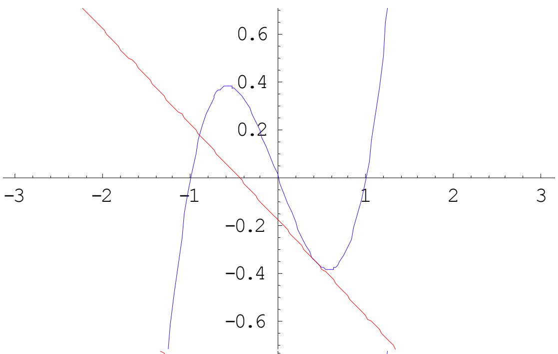
[6] Para una demostración de lo expuesto puede verse "Luemberguer, Programación lineal y no lineal" citado en la bibliografía al final

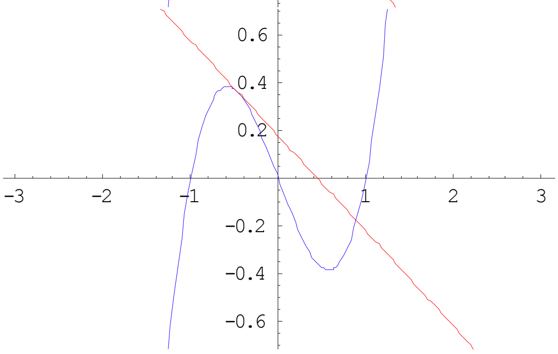
1 2





3 4





.

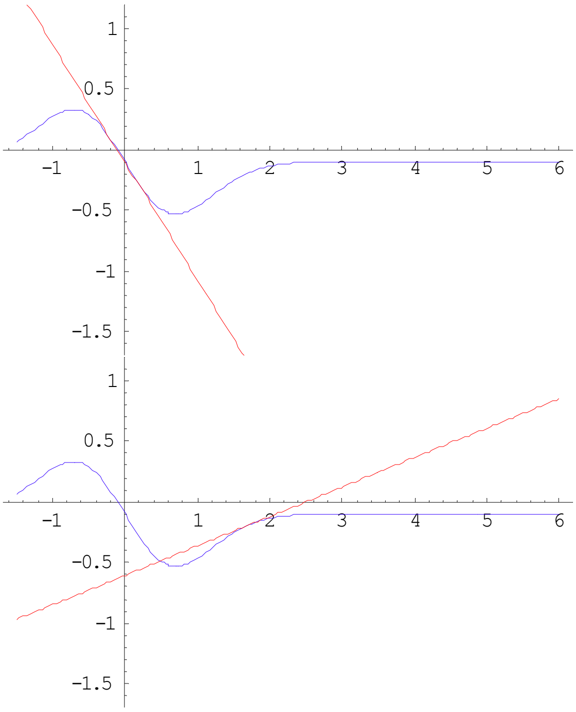
como se aprecia partiendo de un valor menor que uno, en este caso $x1=5^{-0.5}$, el algoritmo no converge a la raíz si no que oscila permanentemente en un circulo vicioso para toda la eternidad sin poder escaparse de él. De esta manera resulta imposible que detecte cualquiera de las tres raíces ubicadas en -1, 0 , 1.

- como todo método numerico los valores que se obtienen estan condicionados por el valor de arranque y en ese sentido cuando se hala una raiz no se puede estar seguro que sea la única mas cuando no se conoce mucho como se comporta la función. En estos casos es aconsejable tomar varios puntos de arrranque distintos y observar lo que sucede.

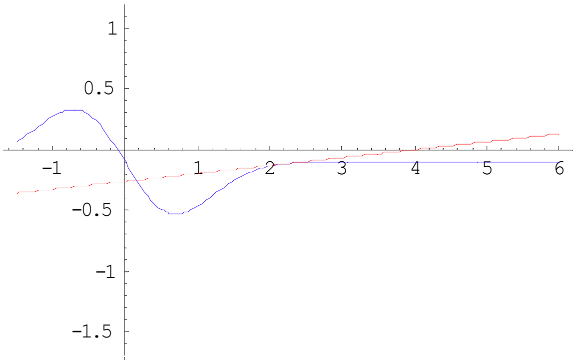
- Por ultimo puede llegar a suceder que debido a la lejanía del valor inicial con respecto a la raiz buscada conduzcan a un alejamiento continuo de dicho valor ya que el método no puede en estos casos adaptarse con total fidelidad a la curvatura de la función. Un ejemplo de este ultimo problema se puede observar en la siguient función:

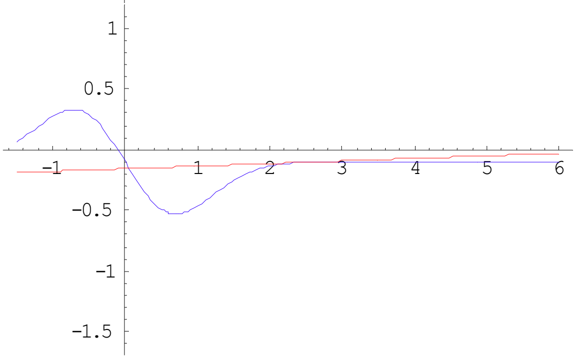
$f(x) = xe^{-x^2} - 0.1$ cuyo comportamiento cuando se aplica el método de Newton para hallar la raiz ubicada en el punto -0.1 aproximadamente partiendo de 0.1, se muestra en las siguientes graficas:

1 2



3 4





obsérvese la similitud del método de Newton con el de Newton-Raphson para minimizar funciones, en cuanto a su formula y a los problemas esto se debe a que básicamente resuelven el mismo tipo de problemas solo que desfasados en una derivada de orden uno.

Un método alternativo que no reviste de este ultimo problema es el método de la secante o también que en vez usar rectas tangentes utiliza rectas secantes en donde se deben dar dos valores iniciales para arrancar. Este método es mas inteligente ante los cambios bruscos de curvaturas como en la función anterior que el método de Newton, pero tiene la desventaja de no converge tan rápidamente como aquel.

Otro método también usado es el método de la bisección que consiste en tomar un rango de valores en donde se produce un cambio de signo de la función y mediante evaluaciones sucesivas se van eliminando la mitad del intervalo original. Es ultimo método tiene la enorme ventaja de no presentar los inconvenientes del método de Newton pero resulta también ser el de menor velocidad de convergencia.

2.- Metodo de Newton-Raphson para Optimización

Los métodos numéricos para optimizar funciones surgen para dar respuesta a los problemas con que usualmente se topan los mecamismos tradicionales de optimización simbólica, como suelen ser los casos en que la función objetivo no pueda ser derivable por una vía simbólica, o cuando la ecuación resultante de la condición de primer orden no puede resolverse por una vía algebraica. Ejemplos de este tipo de este ultimo serian cuando la función objetivo es

Dentro de los métodos numéricos para resolver este tipo de problemas se encuentra el desarrollado por Newton-Raphson. El algoritmo consiste en buscar los valores de la variable independiente que hacen mínima a la función a través de sucesivas iteraciones.

La idea subyacente que sigue el método es la de, partir de un punto arbitrario x_0, que se toma como una primera aproximación, y mejorarlo llevándo esta estimación al valor que hace mínima la parábola que mejor se ajuste a la función en el punto x_0. Luego se toma como segunda aproximación este ultimo valor así hallado, x_1 , y se lo sustituye por el punto que hace mínima la nueva parábola que mejor se ajuste a la función en el punto x_1, y así sucesivamente. La parábola que mejor se ajusta a una función en un punto dado es la que se obtiene por medio de un desarrollo en serie de segundo orden de dicha función, es decir por medio de un polinomio de Taylor de grado dos, ya que por definición el polinomio de Taylor de una funcion es aquel polinomio con la propiedad de que en el punto en donde se desarrolla los valores de sus derivadas sucesivas hasta el orden n (dos en este caso) coinciden con los de la función original. Es este pues la idea intuitiva del método de Newton-Raphson.

A continuación se traducirá lo antes mencionado a un lenguaje mas formal:
Sea x_0 un valor inicial y sea $T_2(f,x,x_0)$ el desarrollo en serie de Taylor de orden dos de $f(x)$ expandido en x_0 , se tiene:

$$f(x) \cong f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2}(x - x_0)^2 = T_2(f,x,x_0)$$

luego x_1, el valor de x en donde $T_2(f,x,x_0)$ se minimiza será:

$$\frac{\partial T_2(f,x,x_0)}{\partial x} = f'(x_0) + f''(x_0)(x - x_0) = 0 \rightarrow x_1 = x_0 - \frac{f'(x_0)}{f''(x_0)}$$

que se puede expresar como

$$x_1 = x_0 + h_0 \text{ donde } h_0 = -\frac{f'(x_0)}{f''(x_0)}$$

que es formula del algoritmo de Newton-Raphson siempre que se cumpla además la condición de segundo orden:

$$\frac{\partial^2 T(f,x,x_0)}{\partial x} = f''(x_0) > 0$$

así se obtiene x_1 el nuevo valor de la estimación el cual servirá para iniciar nuevamente el proceso obteniéndose en este segundo paso:

$$x_2 = x_1 + h_1 \qquad \text{donde } h_1 = -\frac{f''(x_1)}{f'(x_1)}$$

y en general en el i-ésimo paso se tendrá:

$$x_i = x_{i-1} + h_{i-1} \qquad \text{donde } h_{i-1} = -\frac{f''(x_{i-1})}{f'(x_{i-1})}$$

siendo esta la expresión del proceso iterativo que como puede observarse consta de infinitos pasos a menos que se impongan ciertas condiciones para su detención. Estas básicamente pueden ser de dos tipos: por un lado se puede fijar de antemano un numero predeterminado de pasos, por otro lado puede establecerse que este proceso iterativo concluya cuando la mejora de un paso con respecto a otro sea menor que un cierto número ε (es decir $|h_i| < \varepsilon$)o bien se imponen ambas y el proceso concluye cuando sucede cualquiera de las dos.

A continuación se verá la aplicación del método a siguiente ejercicio:

2) Encontrar el mínimo de la siguiente función: $f(x) = x^2 + x^4$

Para resolver este problema en primer lugar obtenemos f' y f'' en forma genérica;

$$f'(x) = 2x + 4x^3$$
$$f''(x) = 2 + 12x^2$$

Con lo cual $h_i = -\dfrac{2x_i + 4x_i^{\,3}}{2 + 12x_i^{\,2}}$ y por ende el algoritmo es

$$x_i = x_{i-1} - \frac{2x_{i-1} + 4x^3_{\,i-1}}{2 + 12x_{i-1}^{\,2}}$$

Partiendo del valor arbitrario $x_0 = 15$ tomando como máximo 12 iteraciones, y un ε = $=10^{-50}$ y mostrando los resultados con una precisión de 80 dígitos[7] se tiene:

X_1	10
H_1	-3.3444259567387687188019966722129783693843594009983361064891846921797004991680532
X_2	6.6555740432612312811980033277870216306156405990016638935108153078202995008319468
H_2	-2.2351565479591666878283937794547070057761162521008143577656954467623290003060702
X_3	4.4204174953020645933696095483323146248395243469008495357451198610579705005258765
H_3	-1.4983958020009316656887167648883680507010509654546487075521051759339652884153298
X_4	2.9220216933011329276808927834439465741384733814462008281930146851240052121105467
H_4	-1.0113046059534156191562822119042580487845678296380645075442815545517707707042004
X_5	1.9107170873477173085246105715396885253539055518081363206487331305722344414063463
H_5	-0.69251841081575773302824699870127423107518252347233164513185603908683460354855320
X_6	1.2181986765319595754963635728384142942787230283358046755168770914853998378577931
H_6	-0.48806627674154650153913342460965883900155201998247649705279027200751257907903599
X_7	0.73013239979041307395723014822875545527717100835332817846408681947788725877875712
H_7	-0.35931124904057008118718047084433992581342700482644856955183947152066709556702817
X_8	0.37082115074984299277004967738441552946374400352687960891224734795722016321172894
H_8	-0.25906312355599756609930709752309080578327052667731071877987308146944797332833470
X_9	0.11175802719384542667074257986132472368047347684956889013237426648777218988339424
H_9	-0.10656390340944918099020834749206622112804747599847606755986506135596055476460302
X_{10}	0.0051941237843962456805342323692585025524260008510928225725092051318116351187912213
H_{10}	-0.0051935633476767009119847833998909895159464073696191528315789367781021792224442514
X_{11}	5.6043671954476854944896936751303647959348147366974093026835370945589634696983093´ 10^{-7}
H_{11}	-5.6043671954406444070398163343989344115839742149374821722839450946397976467260393´ 10^{-7}
X_{12}	7.0410874498773407314303843508405217599271303995919999191658229722700391480773801´ 10^{-19}
H_{12}	-7.0410874498773407314303843508405217459641153498237594945963842481084075086798773´ 10^{-19}

[7] Nótese que se dijo 'mostrando' y no 'trabajando' con una precisión de 80 dígitos ya que los cálculos internamente fueron efectuados en forma exacta usando operaciones simbólicas. De este modo, en cada iteración no existen errores de arrastre por redondeos en pasos anteriores evitando así posibles caídas en situaciones caóticas. Se uso para ello el software Mathemática 4.0 cuyas rutinas de programación se exponen al final del trabajo.

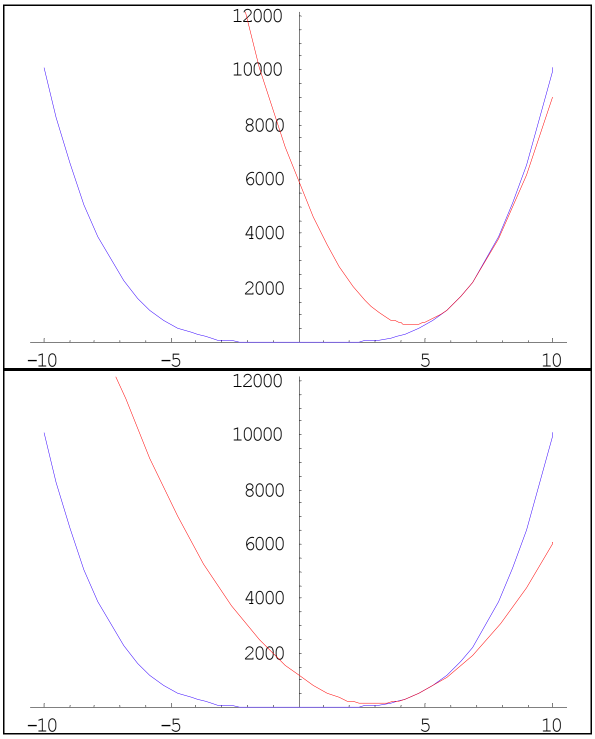
X_{13}	1.3963015049768240424569438724161631639397502869478954113276168758990795713483516´ 10^{-54}
H_{13}	-1.3963015049768240424569438724161631639397502869478954113276168758990795713483516´ 10^{-54}

así el valor critico x_{13} es un valor candidato a ser un mínimo. Para comprobar que efectivamente evaluamos dicho valor en la condición de segundo orden:

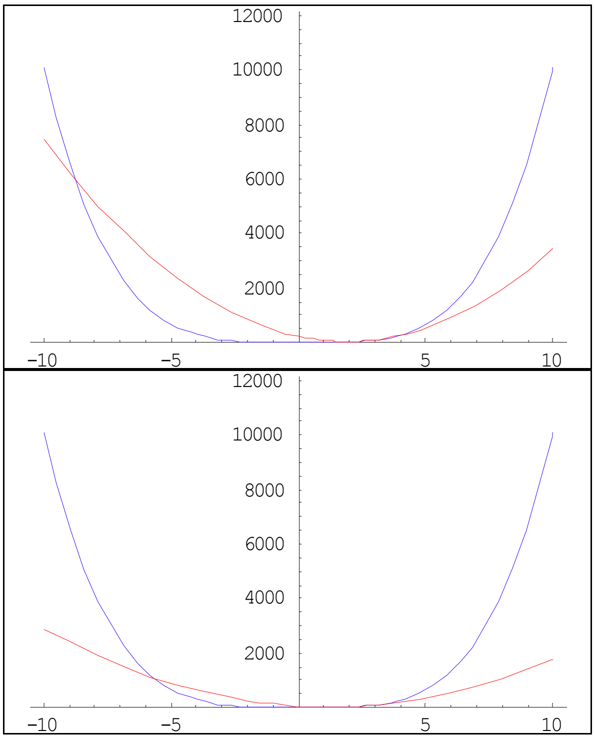
$f''(x_{13}) = 2 + 12x_{13}^2 = 2 > 0$ ya que x_{13} es positivo

por lo tanto se cumple la CSO, y el valor encontrado hace mínima la función ($f(x_{13}) = 0$).

Geométricamente para los cuatro primeros pasos, esto luce como sigue[8]:



[8] El formato de lectura de los cuatro pasos es $\begin{matrix} 1 & 2 \\ 3 & 4 \end{matrix}$



En donde la grafica en color azul muestra la función del ejercicio y las graficas rojas las parábolas de aproximación que surgen de los desarrollos en serie de orden dos expandidas en el punto de inicio la primera, en el punto donde se minimiza la primer parábola la segunda, en

la minimización de la segunda parábola lo hace la tercera y la cuarta en el punto mínimo de esta ultima.

Así como para dar lugar a este método se utilizó un desarrollo en serie de orden dos uno se podría preguntar por que no aproximar la función por medio de un desarrollo de orden tres, cuatro, etc que sin lugar a dudas será una mejor aproximación. Sin embargo se puede demostrar que dicho esfuerzo algebraico no redunda en un incremento la velocidad de convergencia del método (es decir la rapidez con que se aproxima al valor buscado) ya que cualquiera sea el orden del polinomio utilizado en su aproximación el orden de convergencia del algoritmo es igual a dos (aproximadamente, esto puede interpretarse como que en cada paso la exactitud se duplica).

En comparación a otros métodos este es uno de los más veloces, pero al igual que otros no está exento de fallas.

Entre estas cabe mencionar las siguientes:

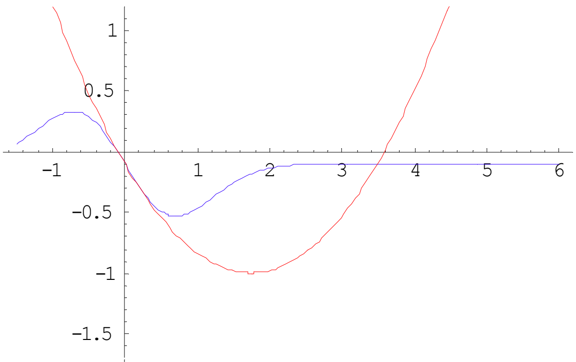
- Podría darse un caso en que el valor de partida o en un valor arribado la derivada segunda en ese punto sea igual a cero. En dicho caso el método explota pues en dicho punto la parábola se transforma en una recta y el método no sabe en que dirección continuar para minimizar a la función. En dicho punto la función experimentará un punto de inflexión (siempre que la primer derivada impar, precedida de todas sus inmediatas pares anteriores igual a cero, sea no nula). Se puede decir que en estos casos el método enloquece y no puede continuar.

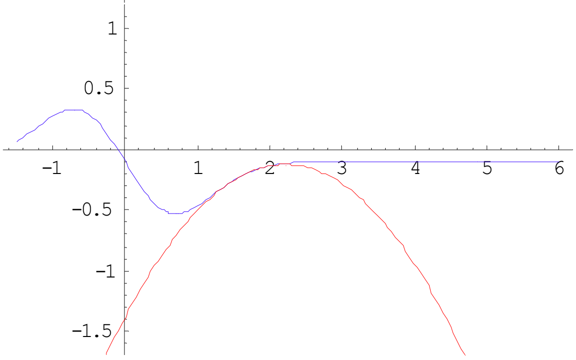
- Al igual que todos los métodos numéricos de optimización están condicionados por el valor inicial de arranque pudiéndose normalmente presentar situaciones en donde se arriben a diferentes resultados según de donde se parta. Además en estos casos no se puede distinguir un optimo global de un uno local. En consecuencia es aconsejable usar varios puntos de partidas y luego evaluar el menor de todos los óptimos encontrados.

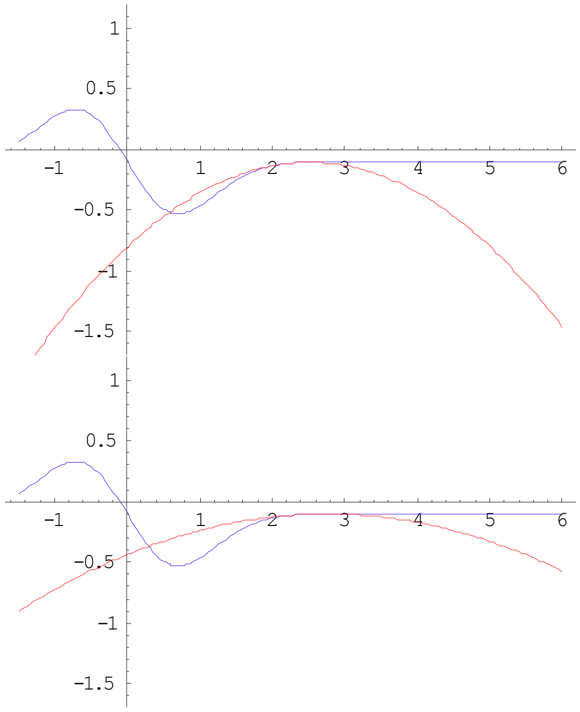
- La condicionalidad del método con respecto a la semilla de arranque puede además hacer que en determinadas situaciones una alejada elección del valor inicial con respecto al optimo buscado hagan confundir al método ya que al no ser lo suficientemente sensible a la curvatura de la función pueden obtenerse valores que en el proceso de aproximación se pasen del objetivo alejándose cada vez siendo atraídos por otro punto critico. En algunos casos puede que en el proceso de acercarse a un mínimo se aproximen inclusive a un máximo. Un ejemplo de este tipo de problema se encuentra en la siguiente función:

  $f(x) = -xe^{-x^2}$ partiendo del valor $x_0 = 0.1$

  Geométricamente este problema se ilustraría como:







En donde se ve claramente como partiendo del valor inicial 0.1 con el animo de aproximarnos al mínimo que se encuentra en $x = \frac{1}{\sqrt{2}} \cong 0.707172...$ el algoritmo se pasa y se desvía atraído en busca de un máximo. Los resultados arrojados por Mathemática son:

```
f x_ [illegible] E^ x^2 [illegible] g1 [illegible], N g2 [illegible] f' [illegible] / f'' [illegible]
g i [illegible] i-1 [illegible] f [illegible]
```

```
Catch Do [illegible] Abs [illegible] 000000001, Throw [illegible] 10
0.1000000000000000000000000000000000000000000000000000000000000000000000000
1.7442953020134228187919463087248322147651006711409395973154362416107382550335570
2.2167698982006083654887181556845620007706250976538424212560712614735604626480791
2.5083891683420673837557364645377416440054016213642790538630784670920140709586994
2.7493167811987578777058731800045996750504770955976911367638444568421784153557807
2.9611968293320005807629234592456188089969385664848973605865538834281743510092983
3.1532773462354514619115288515771538134188261142074642392559129538971441490855054
```

Para esto, se debe ir observando el comportamiento de la función en sucesivos puntos. Una estrategia para evitar el desplazamiento en la dirección errónea es no tomar todo el valor *h* sino elegir en valor λ y tomar un valor λh para encontrar el mínimo[9].

Este método puede extenderse para tratar el caso de optimización de funciones de varias variables en donde con n desarrollo similar se arriba a: $\mathbf{x_i} = \mathbf{x_{i-1}} + \mathbf{h_{i-1}}$ en donde $\mathbf{h_{i-1}} = -\mathbf{H}(\mathbf{x_{i-1}})^{-1} \nabla \mathbf{f}(\mathbf{x_{i-1}})$ donde ahora las variables en negritas indican vectores siendo además *H y* ∇f la matriz hessiana y el vector gradiente

[9] Esto se lo conoce como Método de Newton-Raphson generalizado.

de f respectivamente presentando todas las fallas descriptas con anterioridad. La intuición que existen detrás de estas nuevas fórmulas para el caso multivariado se puede expresar diciendo que partiendo de un punto arbitrario lo que se busca es hallar el valor que minimiza al paraboloide elíptico que mejor se adapta a la hipersuperficie en ese punto n-dimensional del cual se partió. Evidentemente el paraboloide elíptico que mejor se adapte será aquel correspondiente al desarrollo de Taylor de orden dos multivariado.

Un método alternativo también famoso para optimizar funciones de varias variables es el método del gradiente también conocido como método de descenso por la menor pendiente. Como su nombre lo indica la idea del método se basa en el concepto de que la derivada direccional de una función en un punto es máxima cuando esta se calcula en torno a la dirección del gradiente. Este método no posee las falencias del anterior pero su orden de corvengencia es mucho menor. También suelen utilizarse combinaciones de ambos métodos para diversificar las falencias de cada uno[10].

[10] Existen una gran diversidad de métodos de optimización. Para una visión mas amplia de éstos y otros métodos adicionales puede verse "Luemberguer, Programación lineal y no lineal".

BIBLIOGRAFÍA:

- **APÓSTOL, Tom:** *Calculus.* Ed. Reverté. 1976. Segunda Edición
- **CHIANG, Alpha:** *Economía Matemática.* Ed. McGraw-Hill. 1998
- **LUEMBERGER, M.:** *Programcicion lineal y no lineal*
- **STEWARD, James:** *Cálculo.* Ed. Iberoamerica.1981
- **STEIN, S.:** *Cálculo con geometría analítica. Ed Mc. Graw-Hill. 1995*