# Búsqueda de raíces

## Método de Newton

Anabel Estévez Carrillo y Rebeka Luis Hernández

La Laguna, 14 de Mayo de 2014

Técnicas Experimentales
Lenguajes y Sistemas Informácicos
Facultad de Matemáticas
Universidad de La Laguna

# Índice

# Índice

# Índice

# Índice

# Índice

# Motivación y objetivos

El objetivo propuesto es obtener los conocimientos necesarios para desarrollar un informe tecnico-científico usando latex, así como:

- La implementación con Python del método de Newton.

# Motivación y objetivos

El objetivo propuesto es obtener los conocimientos necesarios para desarrollar un informe tecnico-científico usando latex, así como:

- La implementación con Python del método de Newton.
- Como se comporta el método de Newton aplicado a la función f(x)=ln x.

# Motivación y objetivos

El objetivo propuesto es obtener los conocimientos necesarios para desarrollar un informe tecnico-científico usando latex, así como:

- La implementación con Python del método de Newton.
- Como se comporta el método de Newton aplicado a la función f(x)=ln x.

El procedimiento será descrito en un artículo realizado con el procesador de texto *LaTeX* , el cual permite crear documentos con un aspecto profesional y Beamer, una clase de *LaTeX* para la creación de presentaciones.

# Fundamentos teóricos

**Definición**

*El método de Newton-Raphson es un método iterativo con el que se pueden encontrar aproximaciones de soluciones de ecuaciones no lineales.*

# Fundamentos teóricos

**Definición**

*El método de Newton-Raphson es un método iterativo con el que se pueden encontrar aproximaciones de soluciones de ecuaciones no lineales.*

El método parte de un valor inicial que se introuce en la ecuación:

$$x_n + 1 = x_n - \frac{f(x_n)}{f'(x_n)} \ ,$$

obteniándose así un resultado.Ese resultado se introduce en la misma expresión, obteniendo un nuevo resultado, y así sucesivamente.Si la elección del valor inicial es buena el método proporciona una aproximación a la solución real mejor que la que se haya tenido anteriormente.

Para conseguir una buena aproximación se debe tener en cuenta que:

- La función debe ser continua y dos veces derivable en un intervalo [a,b].

Para conseguir una buena aproximación se debe tener en cuenta que:

- La función debe ser continua y dos veces derivable en un intervalo [a,b].
- Se debe escoger un intervalo en el que $f(x) = 0$ cumpla el teorema de Bolzano.

Para conseguir una buena aproximación se debe tener en cuenta que:

- La función debe ser continua y dos veces derivable en un intervalo [a,b].
- Se debe escoger un intervalo en el que $f(x) = 0$ cumpla el teorema de Bolzano.
- $f(a)$ y $f(b)$ tengan signos distintos.

En conclusión, eligiendo como valor inicial el extremo del intervalo que cumpla:

En conclusión, eligiendo como valor inicial el extremo del intervalo que cumpla:

$$f(a) \times f'(a) > 0, \text{ o que } f(b) \times f'(b) > 0$$

está asegurado que el método converge a la única solución de la ecuación en [ a,b ].

# Descripción de los experimentos

Se ha redactado un código en el lenguaje de programación *Python*, este cuenta con una función llamada newton que calcula el resultado de aplicar la fórmula de Newton evaluando el error cometido. Además, el proceso se ejecutará tantas veces como sea necesario para que el error sea menor que un valor prefijado.

Ahora bien, el programa se puede ejecutar de dos formas:

- Utilizando los parámetros introducidos por el usuario, siempre que estos sean mayores que cero.

Ahora bien, el programa se puede ejecutar de dos formas:

- Utilizando los parámetros introducidos por el usuario, siempre que estos sean mayores que cero.
- Tomando los valores por defecto.

# Descripción del material

El ordenador empleado cuenta con:

- Un procesador Intel(R) Pentium(R) CPU B960 @ 2.20GHz.

# Descripción del material

El ordenador empleado cuenta con:

- Un procesador Intel(R) Pentium(R) CPU B960 @ 2.20GHz.
- 400Gb de disco duro y 6Gb de memoria RAM.

# Descripción del material

El ordenador empleado cuenta con:

- Un procesador Intel(R) Pentium(R) CPU B960 @ 2.20GHz.
- 400Gb de disco duro y 6Gb de memoria RAM.
- Una velocidad de CPU de 800.000Hz y un caché de CPU de 2048 KB.

# Descripción del material

El ordenador empleado cuenta con:

- Un procesador Intel(R) Pentium(R) CPU B960 @ 2.20GHz.
- 400Gb de disco duro y 6Gb de memoria RAM.
- Una velocidad de CPU de 800.000Hz y un caché de CPU de 2048 KB.
- El sistema operativo Linux *kubuntu*.

# Descripción del material

El ordenador empleado cuenta con:

- Un procesador Intel(R) Pentium(R) CPU B960 @ 2.20GHz.
- 400Gb de disco duro y 6Gb de memoria RAM.
- Una velocidad de CPU de 800.000Hz y un caché de CPU de 2048 KB.
- El sistema operativo Linux *kubuntu*.
  Además, se ha interpretado el código descrito con el intérprete de Python, la versión utilizada ha sido la 2.7.3

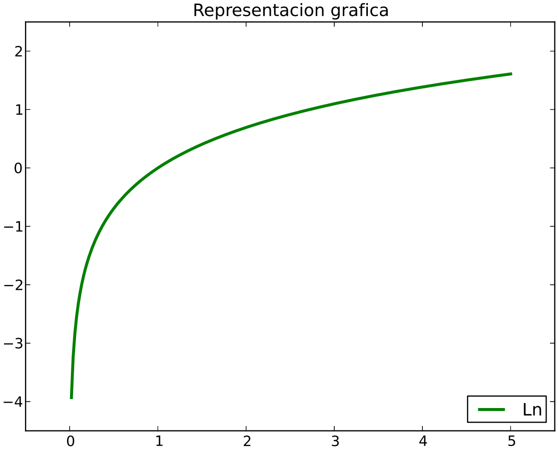
# Resultados obtenidos

Se ha comprobado su precisión dando valores que serán las aproximaciones iniciales de la raíz y se han obtenido los resultados plasmados en la siguiente tabla.

| Valor inicial | Raiz | f(raiz) | Iteraciones necesarias |
|---|---|---|---|
| 1 | 1.000 | 0.000 | 1 |
| 2 | 1.000 | 0.000 | 6 |
| 3 | 1.000 | 0.000 | 7 |
| 5 | 1.000 | 0.000 | 8 |
| 6 | 1.000 | 0.000 | 10 |
| 7 | 1.000 | 0.000 | 11 |

Table: Resultados experimentales

Al ejecutar el programa con distintos valores como aproximación inicial de la raíz se observa que en todos los casos se calcula la raíz situada en el punto x=1 con precisión, y por tanto el valor de la función en ese punto es cero.



Cabe destacar que el número de veces que se aplica la fórmula del método de newton, crece cuando la aproximación inicial que introducimos se aleja de la raíz. Si se traza una función que represente el número de iteraciones necesarias para calcular las raíces mediante el método de newton para la función $f(x) = log(x)$, se observa que ésta tiene una mayor pendiente cuando nos aproximamos cero

El tiempo de CPU aumenta a medida que se incrementa el valor inicial. Sin embargo, para cualquier valor inicial que se introduzca el tiempo de CPU registrado será cero. Así que, para obtener datos al respecto, se ha ejecutado el programa 100000 para cada valor empleando el módulo timeit.

| Valor inicial | Tiempo de CPU al ejecutarse 100000 |
|---|---|
| 1 | 0.1154410839 |
| 2 | 0.5907168388 |
| 3 | 0.6773560047 |
| 4 | 0.5563027859 |
| 5 | 0.7719118595 |
| 6 | 0.9161560535 |
| 6.5 | 0.8663120270 |
| 7 | 0.9962458611 |
| 7.35 | 1.3717629910 |

Table: Resultados tiempo CPU

Estos resultados se han representado en la siguiente gráfica:

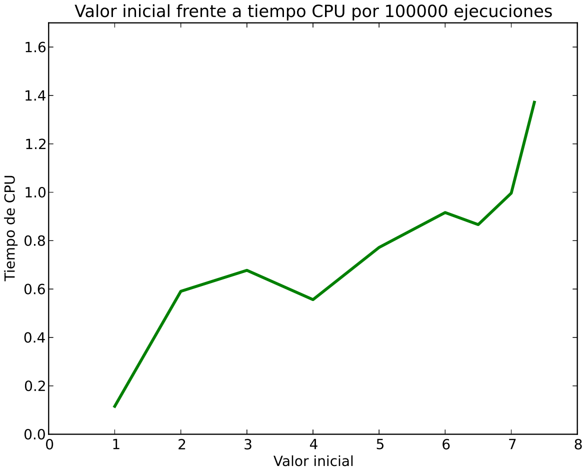



Figure: Tiempo de CPU

# Conclusiones

- Se ha implementado un código en el lenguaje de programacion Python que consigue resolver de manera precisa la raiz de la función tratada aplicando el método de Newton, si bien el algoritmo diseñado puede ser extrapolado a cualquier otra función.

# Conclusiones

- Se ha implementado un código en el lenguaje de programacion Python que consigue resolver de manera precisa la raiz de la función tratada aplicando el método de Newton, si bien el algoritmo diseñado puede ser extrapolado a cualquier otra función.
- Si el valor de partida que tomamos como aproximación inicial de la raiz diste de éste, se necesitará aplicar el método de Newton una mayor cantidad de veces para obtener resultados precisos.

# Conclusiones

- Se ha implementado un código en el lenguaje de programacion Python que consigue resolver de manera precisa la raiz de la función tratada aplicando el método de Newton, si bien el algoritmo diseñado puede ser extrapolado a cualquier otra función.
- Si el valor de partida que tomamos como aproximación inicial de la raiz diste de éste, se necesitará aplicar el método de Newton una mayor cantidad de veces para obtener resultados precisos.
- El programa diseñado actúa con eficiencia, sin embargo, el tiempo de CPU necesarió será mayor cuando la aproximación inicial de la raíz se aleje de la misma.

# Conclusiones

- Se ha implementado un código en el lenguaje de programacion Python que consigue resolver de manera precisa la raiz de la función tratada aplicando el método de Newton, si bien el algoritmo diseñado puede ser extrapolado a cualquier otra función.
- Si el valor de partida que tomamos como aproximación inicial de la raiz diste de éste, se necesitará aplicar el método de Newton una mayor cantidad de veces para obtener resultados precisos.
- El programa diseñado actúa con eficiencia, sin embargo, el tiempo de CPU necesarió será mayor cuando la aproximación inicial de la raíz se aleje de la misma.

- La elección de un buen valor como aproximación inicial de la raíz es importante para que el programa funcione correctamente. Se han desarrollado los puntos a tener en cuenta para elegir de forma correcta el valor inicial.

- La elección de un buen valor como aproximación inicial de la raíz es importante para que el programa funcione correctamente. Se han desarrollado los puntos a tener en cuenta para elegir de forma correcta el valor inicial.
- A partir del valor 7.389, las aproximaciones en vez de acercarse al 1.00000 se alejan, por lo que llega un momento en el que el valor de x es tan grande, que se produce una división por cero al evaluar la expresión $\frac{log(x)}{(1/x)}$. Por lo tanto, como x es muy grande el cociente $\frac{1}{x}$ tiende a 0 y la división principal provoca el error.

# Bibliografía


- Tutorial de Python. *http : //es.tldp.org/Tutoriales/Python/tut.pdf*
- Tutorial de Latex. *http : //www.fing.edu.uy/ canale/latex.pdf*
- Método de Newton.