# Metodos Numéricos

Andrés Giraldo Gil
a_giraldo@javeriana.edu.co

Erika Alejandra González
gonzalez_erika@javeriana.edu.co

Leonel Steven Londoño
leonel-londono@javeiana.edu.co

15 de febrero de 2020

## 1. Numero de Operaciones

### 1.1. Teorema de Horner

#### 1.1.1. Marco teórico

El algoritmo de Horner es un algoritmo para evaluar de forma eficiente funciones polínomicas de grado $n$ de una forma monomial. [1]

Sea $P(x) = a_0x^n + a_1x^{n-1} + ... + a_n$ un polinomio, entonces

$$b_0 = a_0$$
$$b_k = a_{k-1}x_0(k = 1, ..., n-1)$$
$$b_n = P(x_0)$$

Así el cociente $P(x)/(x - x_0$ es $Q(x) = b_0x^{n-1} + b_1x^{n-1} + ... + b_{n-1}$

**Ejemplo:** Sea $P(x) = 2x^4 - 3x^2 + 3x - 4$ un polinomio incompleto de grado 4 para cualquier $x_0$ el polinomio puede ser evaluado de diferentes maneras:
Metodo 1: $P(x_0) = 2 * x_0 * x_0 * x_0 * x_0 - 3 * x_0 * x_0 + 3 * x - 4$
Metodo 2: $P(x_0) = -3 * x * (x) + 0 * x * (x^2) + 2 * x * (x^3) + 3 * x - 4$
Metodo 3: $P(x_0) = -4 + x * (3 - x * (-3 + x * (-3 + x * (x * 2))))$

| Método | Multiplicaciones |
|---|---|
| Método 1 | $(n(n+1))/2$ |
| Método 2 | $2n - 1$ |
| Método 3 | $n$ |

Tabla 1: Teorema de Horner

### 1.1.2. Ejercicios

**1. Utilice el método de inducción matemática para demostrar el resultado del método 3.**

- Sea $P_0(x) = a_0 x^0 = a_0$. El número de multiplicaciones necesarias para hallar $P_0(x_0)$ es cero. Por lo que el caso $k = 0$ se cumple.
- Se asume que $P_k(x) = a_0 + a_1 x + ... + a_k x_0^k$ y que $P_k(x_0)) = a_0 + a_1 x_0 + ... + a_k x^k$ tiene un total de $k$ multiplicaciones.
- Se asume que el método Horner se expresa de la manera:

$$\begin{gathered} b_k = a_k \\ b_k = a_{k-1} + b_k * x_0 \\ b_0 = a_0 + b_1 * x_0 \end{gathered}$$

- Se debe llegar a la forma $k + 1$ del polinomio

$$P_{k+1}(x_0) = a_0 + x(a_1 + x(a_2 + ... + x(a_k + x a_{k+1})))$$

- Se reemplaza $a_k$ por $b_k$. Esta es la primera instrucción del método de Horner:

$$P_{k+1}(x_0) = a_0 + x_0(a_1 + x(a_2 + ... + x(a_k + b_k)))$$

- También se añade la iteración en el método de Horner para $b_{k+1}$

$$\begin{gathered} b_k = a_k \\ b_k = a_{k-1} + b_{k+1} * x_0 \\ b_0 = a_0 + b_1 * x_0 \end{gathered}$$

- Se reemplaza $b_k$ en $P_k(x_0)$:

$$P_k(x_0) = a_0 + x_0(a_{k-1} + x_0(a_k + b_{k+1} * x_0)$$

  Lo anterior es equivalente a $P_{k+1}(x_0)$, donde se demuestra que el número de múltiplicaciones es $k$ el cual es el grado del polinomio

**2. Implemente en R o en Python para verificar los resultados de método de Horner.**

**Análisis del Problema:** El algoritmo de Horner es bastante eficiente debido a que no realiza más operaciones de las necesarias para evaluar un polinomio dado en cierto punto. Se comprueba que el número de multiplicaciones es $k$ el cual es el mismo grado del polinomio (4).

**Valores utilizados**

- Entradas:
  Vector [2,0,-3,3,-4]
  $x_i = -2$ -> valor que se va a evaluar en el polinomio
- Salidas:
  10 -> Multiplicación de $x_i$ por cada uno de los valores del vector, al que se le suma $k$
  Número mínimo de operaciones: 8
  Número de multiplicaciones = 4
  Número de sumas = 4

**3. Evaluar x = 1.0001 con $P(x) = 1 + x + x^2 + ... + x^{50}$. Encuentre el error de cálculo al compararlo con la expresion equivalente $Q(x) = (x^{51} - 1)/(x - 1)$**

**Análisis del Problema:** Se evaluó x en ambas funciones, se imprime el resultado de cada una y en el último decimal se puede evidenciar el error que presenta la función P(x) con respecto a Q(x). Realmente el error de cálculo entre funciones es mínimo debido a que las expresiones son equivalentes. Se obtiene un error relativo de $1.64995128671e^{-10}$, este error relativo se halló haciendo una resta del valor absoluto del valor real - valor absoluto del valor aproximado.

**Valores utilizados**

- Entradas:
  1.0001. Valor evaluado
- Salidas:
  $P(x) = 1 + x + x^2 + ... + x^{50} = 51.1277085003$
  $Q(x) = (x^{51} - 1)/(x - 1) = 51.1277085001$

## 1.2. Numeros Binarios

### 1.2.1. Marco teórico

En informática el sistema binario es un sistema de numeracion en el cual los números se representan con el 1 y el 0. Este sistema lo utilizan las computadoras debido al trabajo interno de dos niveles de voltaje. [2]

Los números binarios se expresan como:

$$...b_2b_1b_0b_{-1}2^{-1} + b_{-2}2^{-2}$$

### 1.2.2. Ejercicios

**a. Encuentre los primeros 15 bits en la representacion binaria de $\pi$**

Para realizar la representación del numero $\pi$ en forma, es necesario separar dicho numero en dos partes,parte entera y parte fraccionaria. Para la parte entera se debe dividir entre 2 y tomar el residuo hasta que el cociente sea mayor quel dividendo. despues de esto se toma la parte fraccionaria, se multiplica por 2 hasta conseguir los primeros 15 bits de la representación binaria.

**Entrada:** -> $\pi$

**Salida:** -> 11.00100100 00111

**b. Convertir los siguientes números binarios a base 10: 1010101, 1011.101, 10111.010101..., 111.1111...**

**Entradas:**

- 1010101
- 1011.101

- 10111.010101
- 111.1111

**Salidas:**

- 1010101 = 341
- 1011.101 = 11.625
- 10111.010101 = 23.666503
- 111.1111 = 7.9960938

**c. Convertir los siguientes números de base 10 a binaria: 11.25; $2/3$; 30.6; 99.9**

**Entradas:**

- 11.25
- 2/3
- 30.6
- 99.9

**Salidas:**

- 1011.01000000000000
- 0.10101010101010101
- 11110.1001100110011
- 1100011.11100110011

## 1.3. Punto Flotante

### 1.3.1. Marco teórico

Debido a que las máquinas trabajan con un sistema binario, algunos números reales se pueden representar exactamente como un número binario, sin embargo, otros no. Por ejemplo está el número $\pi$ que no tiene representación finita ni en el sistema decimal ni en el sistema binario.

La representación del punto flotante es una notación científica usada en las computadoras, permite la representación de números reales extremadamente grandes o pequeños de una manera eficiente y compacta. El estándar actual para la represetnación del punto flotante es el IEE754 [3]

Un número flotante consta de tres partes:

1. Signo (+,-)
2. Mantisa, contiene lacadena de bits significativos.

3. Exponente

Se debe considerar que cuando hay un número irracional por ejemplo $\sqrt{3}$ donde no se puede representar con exactitud, los errores de redondeo se tienen en cuenta.

**Épsilon de una máquina** Se llama épsilon de una máquina al menor valor de una determinada máquina que cumple:

$$1,0 + \epsilon - mach > 1,0$$

Para el punto flotante de precisión doble se tiene que:

$$\epsilon_{maq} = 2^{52}$$

### 1.3.2. Ejercicios

**1. ¿Cómo se ajusta un número binario infinito en un número finito de bits?**
Existen dos tipos de infinitos, positivos y negativos. La convención que se usa para representar estos números dice que si un número se considera infinito si cada uno de los bits del exponente es 1 y los de la mantisa son 0. [4]
La norma IEEE754, tiene símbolos especiales para representar los números infinitos positivo y negativos, se denomina NaN (not a number) también se tratan números no normalizados que permiten extender el rando de representación. [5]

| Signo | Exponente | Mantisa |
|---|---|---|
| 0 | 10000100 | 01110010101101000011111 |

Tabla 2: Número con infinitos decimales representado en binario

En maquina de 64 bits, la mantisa, el signo de la mantisa, el exponente y el signo del exponente, pueden ocupar MÁXIMO 64 bits. Es decir, si tomamos un número con infinitos decimales, inevitablemente se perderán decimales al representarlo en punto flotante. Este error es el *error de redondeo* ya que el número decimal se redondea al ignorar los decimales que se encuentran más a la derecha, en otras palabras, los décimales con menor peso. [6] De la matinsa de un número binario infinito se cogen los 23 bits más significativos. Como el resto de bits no pueden representarse, ya que no caben en la mantinsa, estos se descartan. Un número se podria representar como en la tabla 2.

**2. ¿Cuál es la diferencia entre redondeo y recorte?**
**Redondeo:** Es el proceso de descartar cifras en la expresión decimal de un número. Se utiliza con el objetivo de facilitar los cálculos o tratar de dar la impresión de que se conoce un valor con mayor exactitud a la que realmente se tiene. [7]

| Numero Original | Resultado | Notas |
|---|---|---|
| 13,9 | 14 | 14 es más cercano que 13 |
| 13,95 | 14,0 | 13.95 es igual de cercano a 14.0 y 13.9 , regla 2 |
| 22,805 | 22,800 | 22,805 está más cerca de 22,800 que de 22,900 |

Tabla 3: Ejemplos de redondedo según el método del NIST

El método recomentado por el NIST e ISO tiene 2 reglas:

1. Se escoge el número m'as cercano que tenga la cantidad de dígitos significativos escogida.
2. Si ambos números más cercanos con la cantidad de dígitos significativos escogidos son igual de cercanos se escoge el que tiene como último dígito significativo un número par.

**Recorte:** Consiste en eliminar los bits que se encuentran más allá de cierto umbral, esto puede significar una pérdida de información considerable, debido a que el número se recortaïgnorando estas cifras finales, permitiendo así una menor precisión y exactitud en cuanto al valor real de las operaciones.

**4. Indique el número de punto flotante (IEEE) de precisión doble asociado a $x$, el cual se denota como $fl(x)$ para $x(0.4)$**

El formato en punto flotante de doble precisión es un formato de número informático que ocupa 64 bits en su memoria, representa un amplio rango de valores mediante el uso del punto flotante. Se especifíca en el estándar IEEE 754 [8] Para poder realizar este ejercicio se debe pasar de entero a binario el número a evaluar:

$$(0.4)_{10} = (100110011001100\ldots)_b \text{ ó } (1,100110011001100\ldots)_b X10^{-1}$$

Posteriormente se puede definir que el exponente es '-1', la característica es 1023 - 1 = 1022, lo cual es equivalente a:

$$(1022)_{10} \text{ ó } (1111111110) \text{ en binario}$$

La mantisa son los 52 primeros dígitos después de la (,) dado que hay infinitos dígitos se redondea, estos son los datos obtenidos:

$$Mantisa = 1001100110011001100110011001100110011001100110011010$$

Se puede observar en los dos últimos dígitos que han sido redondeados hacia arriba. El número 0.4 en el estándar 754-IEEE corresponde a:

$$fl(x) = (0111111111010011001100110011001100110011001100110011001100110011010)_b$$

En base decimal seria:

$$fl(0.4) = (0.40000000000000002220446049250313080847263336181640625)_{10}$$

**5. Encuentre el error de redondeo para $x = 4$ en el modelo de la artmética de la computadora IEEE**

Según IEEE-754 un número infinito se representa de tal manera que los bits que corresponden al exponente mínimo se colocan todos en1 y el bit del signo indicará el signo del infinito. Durante el proceso de redondeo se usa el valor del bit siguiente para obtener un resultado más aproximado del valor real. En el truncamiento, se corta el número de cifras decimales ignorando las siguientes para el valor de la cifra que va de últimas.

1. Se convierte el número decial a su representación en binario de la siguiente manera:

$$0,4*2=0,8=0 \; 0,8*2=1,6=1 \; 0,6*2=1,2=1 \; 0,2*2=0,4=0 \; 0,8*2=1,6=1$$
$$=0,0110011001100110011001100110011*2^{-2}$$

El carácter de este número es infinito, por lo que para que pueda ser representado hacemos uso de la norma IEEE 754 (para representaciones de números en coma flotante). Se usa un tamaño de número de 64 bits, se dividen en 1 bit para el signo del número, para el exponente se utilizan 11 bits mínimo y 52 bits para la precisión. Se encuentra que:

$Signo = 0$
$Exponentes = (1021)_{10} = (01111111101)_2$
$Fraccion = 1001100110011001100110011001100110011001100110011001$

Por lo tanto, el valor de $fl(0,4)$ en el estándar IEEE 754 es igual a: $(0,399999999999999966693309261245)_{10}$

2. Para encontrar el error de conversión al estándar IEEE 754, se utiliza la fórmula:

$$\|(fl(x) - x)\|/\|x\| \leq e_{mach}/2$$
$$8,32667268468875 * 10^{-17} \leq 1,110223 * 10^{-16}$$

El error de redondeo es $\approx 8,327 * 10^{-17}$ o $8,327 * 10^{-15}$ %

**6. Verificar si el tipo de datos básico de R y Python es de precisión doble IEEE y revisar en R y Python el formato long.**

En R existen cuatro tipos o modos de datos los cuales son: complex, character logical y numeric. Este último es un número real con doble precisión IEEE754, se puede escribir como números enteros (3,-4), fracción decimal (3.27) o con notación científica (3.12e-14). En R no existe el formato long. [9]

En Python los datos de tipo long están implementados a bajo nivel mediante un tipo long de C. El tipo long de Python permite almacenar números de cualquier precisión, limitado por la memoria disponible en la máquina. Para poder obtener datos de precisión doble IEEE754 en Python, se utiliza el tipo de dato "float". Sin embargo, implementa su tipo float a bajo nivel mediante una variable de tipo double de C, utilizando 64 bits, lo que quiere decir que en Python siempre se utiliza doble precisión, y en concreto se sigue el estándar IEEE 754. [10]

**7. Encuentre la representación en número de máquina hexadecimal del núemero real 9.4**

1. Tomamos la parte entera del numero real (9) y se divide entre 16:

$$9/16 = 0.5625$$

2. Se toma la parte fraccionaria del valor inicial (0.4) y se multiplica por la base hexadecimal (16):

$$0.4 * 16 = 6.4$$

3. Nuevamente se toma la parte fraccionaria del anterior resultado (0.4) y se multiplica por la base hexadecimal (16):

$$0.4 * 16 = 6.4$$

4. El paso anterior se repite de manera iterativa cuantas veces sea necesario, con la finalidad de que tras la multiplicación quede un número entero, en este caso no se logra debido a que es un número periódico.

5. Se toma el residuo de la primera división el cual es 9.

   El resultado es 9.6666666...

**8.Encuentre las dos raíces de la ecuacion cuadrática $x^2+912x = 3$ Intente resolver el problema usando la arimética de precisión doble, tenga en cuenta la pérdida de significancia y debe contrarestarla.**

La respuesta se obtuvo gracias al uso de la herramienta *Wolfram Alpha*, la cual arrojo dos raíces:

$x = 6/(282429536481 + \sqrt{79766443076872509863373})$

$x = (-28242953648/2) - (\sqrt{79766443076872509863373/2})$

Al comparar ambos resultados obtenidos con precisión doble (debido al uso de la herramienta) los resultados al restarlos son:

$-2.82429536481000000000000000000000000000000000010^{11}$

Lo que indica la gran diferencia entre ambos puntos y comprueba su lugar como respuestas independientes.

# 2. Raices de una Ecuación

## 2.1. Marco Teórico

Para la solución de ecuaciones no lineales existen métodos iterativos y métodos directos para solucionar el problema $f(x) = 0$.

**Métodos Iterativos** Consite en acercarse a la solución mediante aproximaciones sucesivas, a partir de un valor inicial estimado. En cada iteración se puede usar el resultado anterior para obtener una mejor aproximación.

**Métodos directos** La aproximación a la respuesta se produce a través de una serie finita de operaciones arimeticas y la eficiencia del metodo depende de la cantidad de operaciones el cual se puede asociar al tamaño del problema, en notación O()
*[Información sacada del enunciado del taller]*

## 2.2. Ejercicios

**1. Implemente en R o Python un algoritmo que le permita sumar únicamente los elementos de la sub matriz triangular superior o triangular inferior, dada la matriz cuadrada $A_n$. Imprima varias pruebas, para diferentes valores de $n$ y exprese $f(n)$ en notación $O()$ con una gráfica que muestre su orden de convergencia.**

**Gráfica**

**Análisis del Problema** Para este ejercicio se enviaron distintos tamaños para realizar la matriz cuadrada con dichos valores, estas matrices contenian varios número 1 en su submatriz triangular superior para posteriormente todos estos ser sumados. Este proceso se ralizó con cada matriz y aumentando en cada iteración el tamaño de la misma. Se puede observar que entre más aumenta el tamaño de la matriz mayor es su orden de convergencia. Este algoritmo tiene una complejidad de $O(n)^2$

**Valores Utilizados**

**Entradas:** 10 20 30 35 40 50 60 70 80 90

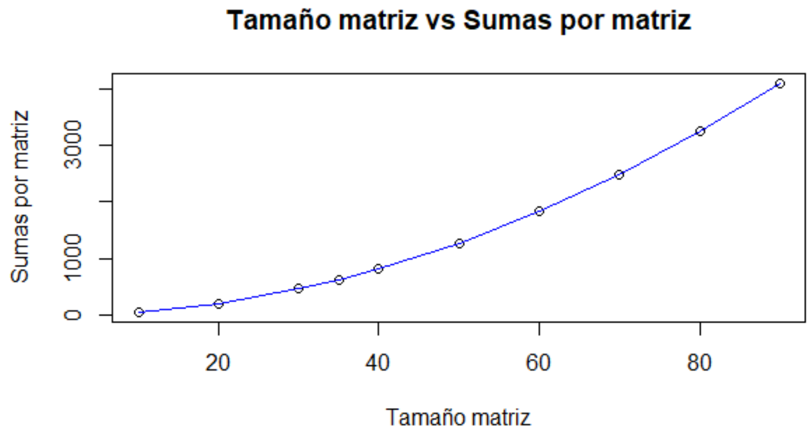


Figura 1: Tamaño de matriz vs Sumas por matriz

**Salidas:** 55 210 465 630 820 1275 1830 2485 3240 4095

**2. Implemente en R o Python un algoritmo que le permita sumar los $n^2$ primeros números naturales al cuadrado. Imprima varias pruebas, para diferentes valores de $n$ exprese $f(n)$ y en notación $O()$ con una gráfica que muestre su orden de convergencia**

**Gráfica**

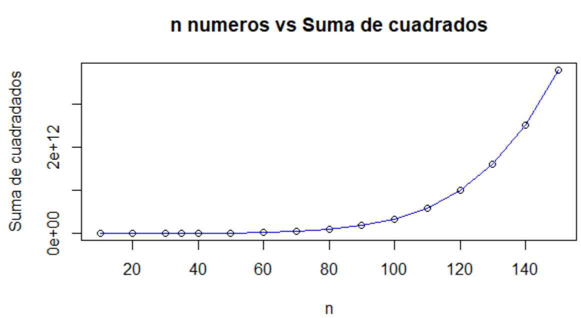


Figura 2: n numeros vs Suma de cuadrados

**Analisis del problema**

En este problema se recibe como entrada una serie de valores que se usarán para sumar los primeros n números naturales al cuadrado, estos valores van aumentando exponencialmente debido a que cada vez es mayor la cantidad de sumas de números al cuadrado. Este algoritmo tiene una complejidad de $O()^2$.

**Valores Utilizados**

**Entradas:** 10 20 30 35 40 50 60 70 80 90 10 110 120 130 140 150

**Salidas:** 338350 21413400 243405150 613505725 1366613600 5211458750 15558480600 39228339150 87401814400 177179806350 333383335000 590593540350 995431682400 1.609079e+12 2.510037e+12 3.797128e+12

**3. Para describir la trayectoria de un cohete se tiene el modelo:** $y(t) = 6 + 2,13t^2 - 0.013t^4$**. Donde y es la altura en [m] y tiempo en [S]. El cohete está colocado verticamente sobre la tierra. Utilizando dos métodos de solución de ecuación no lineal, encuentre la altura máximca que alcanza el cohete.**

**Analisis del problema**

En este problema se fue aumentando el valor de t y realizando constantes comparaciones entre los valores obtenidos hasta encontrar el punto en el que el valor de y era menor al anterior, cuando esto suceda, significa que el cohete ha alcanzado la altura máxima, esta trayectoria forma una parábola concava hacía abajo. Esta es una función cuadrática con concavidad hacía abajo.

**Valores Utilizados**

**Salidas:** Altura final: 877.8647 m

**Gráfica**

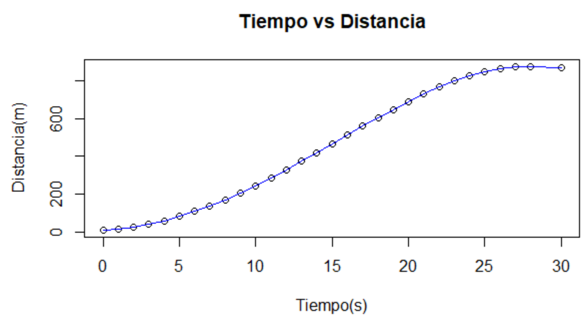


Figura 3: Tiempo vs Distancia

# 3. Convergencia de Metodos Iterativos

## 3.1. Marco teórico

Los metodos iterativos son aquellos que progresivamente van calculando aproximaciones a la solucion de un problema. Es decir que sobre un resultado se realizan operaciones repetitivas y en cada iteracion se espera que el resultado sea la solucion mas aproximada. Por ende los metodos deben cumplir ciertos requisitos, para hallar la solucion o parar de ejecutarse. Entre mas iteraciones el resultado sera mas aproximado, pero eso no quiere decir que no se puedan parar en la primera iteracion, ya que en esta ya se puede obtener un resultado cercano al esperado.[12]

### 3.1.1. Metodo de Newton

1. **Marco Teorico**

   Este metodo se clasifica dentro de los metodos abiertos, aquellos que no garantizan encontrar una convergencia global. La manera en la que esta se puede alcanzar es seleccionando un valor cercano a la raiz esperada, este valor depende de la naturaleza de la funcion.

2. **Analisis de los ejercicios**

Sea:

$$f(x) = e^x - x - 1 \tag{1}$$

**Ejercicio 1:**
Para demostrar que una funcion tiene un cero de multiplicidad 2, entiendase por multiplicidad a la cantidad de raices que tiene una funcion, se usa el metodo de Newton para hallar dichos ceros en un punto determinado.

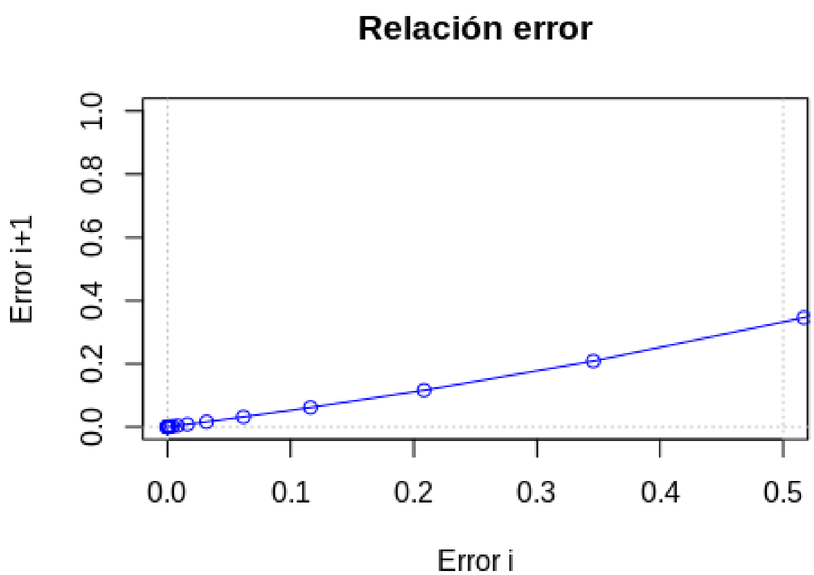


Figura 4: Relacion entre el error

**Ejercicio 2:**
Verificar que la funcion converge a cero pero no de forma cuadratica usando P0 = 1. Se puede observar que al usar P0 = 1 en el metodo de Newton con la funcion dada, una de las raices cambian mientras que la otra permanece constante, eso quiere decir que la convergencia no se realiza de la misma manera.

## 3. Valores Utilizados

**Ejercicio 1:**
Entradas:
La funcion inicial y la derivada de la misma.
x0 = 0
Toleracia = 1*e-8
Cantidad maxima de iteraciones = 30

Salidas:

Iteraciones hasta encontrar las raices = 29.
Raices: 1.848537046e-08 y 1.63852842

**Ejercicio 2:**
Entradas:
La funcion inicial y la derivada de la misma.
x0 = 0
Toleracia = 1*e-8

Cantidad maxima de iteraciones = 30

Salidas:

Iteraciones hasta encontrar las raices = 29.
Raices: 9.982938826e-09 y 1.63852842

# 4. Convergencia Acelerada

## 4.1. Marco teórico

La aceleracion de la convergencia de ciertos problemas consiste en metodologia que permiten obtener las raices de una funcion mas rapido, en terminos de tiempo, recursos y pasos. La idea es encontrar la convergencia de una funcion mas rapido que por otros metodos. El enfoque se basa en considerar una sucesion nueva basada en la original, que tome valores de determinada forma, pero converja mas rapido y al mismo limite que la original.

### 4.1.1. Metodo de Aitken

1. **Marco Teorico**

   Generalmente usada para sucesiones que convergen linealmente, particularmente usada con el metodo de bisección. La idea detras del algoritmo es que si existe una sucesion que converge ha de existir otra que converge en el mismo punto pero mas rapido, por ende el metodo usa dicha sucesion para hallar el resultado de la primera.[11]

2. **Analisis del problema**

   Al aplicar el metodo de Aitken se puede observar que:

   **Ejercicio 1:** Para este ejercicio el tipo de convergencia es cuadratica en x = 1 independientemente del origen y esto sucede a dos cosas, primero, como indica la teoria la sucesion ha de ser cuadratica asi como su convergencia de esa forma esta puede ser mucho mas rapida que el metodo de Biseccion por si solo y segundo, dado que el valor inicial es muy cercano a la raiz la convergencia es mas rapida y precisa.

   **Ejercicio 2:** Al observar los 10 primeros terminos de la sucesion se observa como el valor se va acercando a la raiz cada vez mas, en pocas palabras la curva es es hacia abajo, ver tabla 4.

| Iteracion | valor |
|---|---|
| 3 | 2 |
| 4 | 2 |
| 5 | 1.666666667 |
| 6 | 1.5 |
| 7 | 1.583333333 |
| 8 | 1.583333333 |
| 9 | 1.5625 |
| 10 | 1.572916667 |
| 11 | 1.572916667 |
| 12 | 1.5703125 |

Tabla 4: Primeras 10 iteraciones de Aitken

3. **Valores Utilizados**

   Entradas:

   Funciones:

$$f(x) = cos(1/x) \tag{2}$$

$$g(t) = 4sin(t)cos(t) \tag{3}$$

$$f(t) = 3sin3t1 \tag{4}$$

   Salidas:

   Ejercicio 1:

#### 4.1.2. Metodo de Steffensen

1. **Marco Teorico**

   Al igual que el metodo anterior se usa para obtener los ceros de una funcion o las raices, la diferencia radica en que es una combinación entre el metodo de punto fijo y el Aitken.

2. **Analisis del problema**

   Al aplicar el metodo de Steffensen presenta una convergencia mas rapida que el metodo del Punto Fijo y el Aitken, ademas a diferencia de Newton no necesita derivada alguna de la funcion a evaluar. Tambien se observa que solo necesita un punto inicial, la desventaja de este metodo es que se pudo apreciar que escoger el valor inicial decide el rendimiento del algoritmo, la idea detras de esto es que entre mas cerca este el valor inciial de la raiz el metodo se comportara de manera optima y rapida, pero un numero muy alejado afectara de manera negativa el comportamiento del arlgoritmo.

3. **Valores Utilizados**

   Entradas:

$$f(x) = 2x - sin(x) \tag{5}$$

   Salidas:

   Resultado sin aceleracion: 0.5500068665
   Resultado con aceleracion: 0.550008138
   Valor real de la solucion: 0.55382701
   Resultado sin aceleracion: 1.63848877
   Resultado con aceleracion: 1.638487454
   Valor real de la solucion: 1.63853

# Referencias


[1] A new method of solving numerical equations of all orders, by continuous approximation. Horner, W.G. 1819.

[2] Cálculo infinitesimal y geometría analítica. Aguilar, T. Madrid.

[3] IEEE Std 1003.1. 2004. The Open Group. https://pubs.opengroup.org/onlinepubs/009695399/frontmatter/refdocs.html Consultado el 11 de Febrero de 2020

[4] Representación de números. http://users.df.uba.ar/dmitnik/metodosnumericos/algoritmos/RepresentacionNumerica.html Consultado el 11 de Febrero de 2020

[5] Capitulo 3. Punto Flotante. https://medium.com/@matematicasdiscretaslibro/cap%C3%ADtulo-3-punto-flotante-c689043db98b

[6] Representación numérica en un ordenador. Universidad Politecnica de Madrid. 2015.

[7] ISO 80000-1:2009 apéndice B. Rounding of numbers.

[8] IEEE Standard for Floating-Point Arithmetic. IEEE. 2008.

[9] Computación y programación en R. Conesa, D. Depto. de Estadística e Investigación Operativa. Universidad de Valencia.

[10] Python/Generalidades/Tipos de datos fundamentales https://es.wikibooks.org/wiki/Python/Generalidades/Tipos_de_datos_fundamentales Consultado el 12 de Febrero de 2020

[11] http://files.tutoriametodosnumericos.webnode.es/200000371-eba66eca00/Aitken.pdf

[12] CCIR/ITESM (2009). Metodos iterativos para resolver sistemas lineales. Departamento de Matematicas.