Taller 1

Julian Builes
Santiago Bermudez
Daniel Reyes
Daniel Fierro

Septiembre 2020

1 Numero de Operaciones

1.1 Ejercicio 1

Evaluar el polinomio en x = 1.00000000001 con$P(x) = 1 + x + x^2 + ... + x^{50}$ y la primera derivada. Encuentre el error de calculo al compararlo con los resultados de la expresión equivalente $Q(x) = (x^{51} - 1)/(x - 1)$

Primer Polinomio:
x evaluado en $P(x)$ es igual a 51.00000001275

```
> D(polinomio1, "x")##Aca se deriva
50 * x^49 + 49 * x^48 + 48 * x^47 + 47 * x^46 + 46 * x^45 + 45 *
    x^44 + 44 * x^43 + 43 * x^42 + 42 * x^41 + 41 * x^40 + 40 *
    x^39 + 39 * x^38 + 38 * x^37 + 37 * x^36 + 36 * x^35 + 35 *
    x^34 + 34 * x^33 + 33 * x^32 + 32 * x^31 + 31 * x^30 + 30 *
    x^29 + 29 * x^28 + 28 * x^27 + 27 * x^26 + 26 * x^25 + 25 *
    x^24 + 24 * x^23 + 23 * x^22 + 22 * x^21 + 21 * x^20 + 20 *
    x^19 + 19 * x^18 + 18 * x^17 + 17 * x^16 + 16 * x^15 + 15 *
    x^14 + 14 * x^13 + 13 * x^12 + 12 * x^11 + 11 * x^10 + 10 *
    x^9 + 9 * x^8 + 9 * x^8 + 8 * x^7 + 7 * x^6 + 6 * x^5 + 5 *
    x^4 + 4 * x^3 + 3 * x^2 + 2 * x + 1
>
```

La anterior imagen muestra el polinomio derivado en R

x evaluado en $P'(x)$ es igual a 1275.0000004165

Segundo Polinomio:
x evaluado en $Q(x)$ es igual a 51

```
> polinomio2=expression((x^51-1)/(x-1))
> D(polinomio2, "x")
51 * x^50/(x - 1) - (x^51 - 1)/(x - 1)^2
>
```

La imagen muestra el segundo polinomio derivado en R

Error de calculo= $2.499999e^{-10}$

1.2 Ejercicio 2: Números Binarios

1. Encuentre los primeros 15 bits en la representación binaria de π

   11.0010010000111

2. Convertir los siguientes numeros binarios a base 10: 1010101; 1011.101; 10111.010101...; 111.1111...

   - $1010101_2 -> X_{10} = 85$
   - $1011.101_2 -> X_{10} = 11.625$
   - $10111.010101..._2 -> X_{10} = 23.328125$
   - $111.1111..._2 -> X_{10} = 7.9375$

3. Convierta los siguientes numeros de base 10 a binaria: 11.25; 2/3; 30.6; 99.9

   - $11.25_{10} -> X_2 = 1011.001$
   - $2/3_{10} -> X_2 = 0.1010$
   - $30.6_{10} -> X_2 = 11110.1001$
   - $99.9_{10} -> X_2 = 1100011.1110$

1.3 Ejercicio 3: Representación del punto flotante en números reales

1. ¿ Como se ajusta un numero binario infinito en un numero finito de bits?

   Para los computadores, que tienen una capacidad de memoria limitada, es necesario realizar distintos procedimientos con el fin de modificar ese dígito de una manera que pueda ser almacenada por el computador, en estos casos las computadoras binarias podrían usar aritmética de múltiple precisión para tener la suficiente significancia. Sin embargo es mas preciso utilizar aritmética de punto flotante. La cual utiliza una forma de separar el intervalo y la precisión expresando los números en notación científica:
   $n = fx10^e$
   Donde f es fracción/mantisa y e es el entero. Esta es la mejor manera de lograr la representación de los. Que sigue una representación que se puede ver en la siguiente imagen:

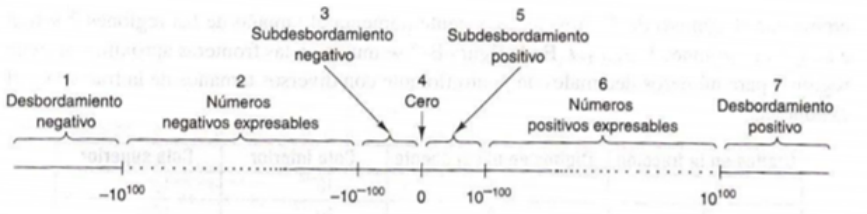


Figura B-1. La línea de los números reales se puede dividir en siete regiones.

La cual muestra que las formas de representación de los números reales. Sin embargo hay otra "técnica" que es utilizada para evitar problemas de desbordamiento, que es cuando un numero es demasiado grande para ser representado, o en su defecto subdesbordamiento que es cuando el numero es demasiado pequeño, en el ajuste de los bits en memoria, y es el redondeo, el cual es que si el resultado de un calculo no se puede expresar en la representación numérica usada se usa el numero mas cercano que si pueda ser usado.

Con lo anterior se ejemplifica como las maquinas logran ajustar los números infinitos, para lograr un almacenamiento en memoria que primero, no exceda sus limites y segundo que aun así signifique lo mismo y no altere el dígito usado. Todo lo anterior es para ajustar el numero a la memoria del computador, y ahora por ultimo toca mencionar que para representar un numero infinito binario en un numero finito de bits, es necesario colocar el '1' en todos los bits de exponente y el '0' en los bits de significado, esto dará la representación que el computador entenderá, este ultimo decifrara si es positivo o negativo según el primer bit de la secuencia.

Concepto	Precisión sencilla	Doble precisión
Bits del signo	1	1
Bits del exponente	8	11
Bits de la fracción	23	52
Total de bits	32	64
Sistema de exponente	Exceso en 127	Exceso en 1023
Intervalo del exponente	-126 a +127	-1022 a +1023
Número normalizado más pequeño	2^{-126}	2^{-1022}
Número normalizado más grande	aprox. 2^{128}	aprox. 2^{1024}
Intervalo decimal	aprox. 10^{-38} a 10^{38}	aprox. 10^{-308} a 10^{308}
Número desnormalizado más pequeño	aprox. 10^{-45}	aprox. 10^{-324}

Figura B-5. Características de los números de punto flotante IEEE

En la anterior imagen podemos ver las características de los números en representación de punto flotantes IEEE [1].

2. ¿Cual es la diferencia entre redondeo y recorte?

La diferencia entre redondeo y recorte consiste en que, en primer lugar el redondeo es el proceso de descartar cifras en la expresión decimal de un número. Se utiliza con el fin de facilitar los cálculos o evitar de dar la impresión de que se conoce un valor con mayor exactitud de la que realmente se tiene. Las aproximaciones en general se simbolizan con $\approx$. Por ejemplo: $\sqrt{2} \approx 1.414$. Cuando se redondean los valores intermedios en un cálculo típicamente se acumulan errores de redondeo que pueden hacer variar significativamente el resultado así obtenido del resultado del cálculo exacto. Para este proceso se deben de escoger las cifras que se conservan que serán las cifras significativas, y se observa si el numero siguientes es mayor o igual a 5, si este es el caso se aumenta 1 a la cifra escogida, en caso contrario se deja como este. Por ejemplo redondear 13.95 manteniendo las primeras dos cifras, el resultado final seria 14. Por otro lado, el recorte simplemente consiste en eliminar los bits que sobrepasen cierto umbral o longitud pero no se hace ninguna aproximación como en el redondeo. Además este método solo aplica para uso binario, mientras que el redondeo se aplica en notación decimal.

3. ¿ Como se ajusta un numero binario infinito en un numero finito de bits?
   Signo ——— Exponente ——————— Mantisa
   —0 ———— 11111111 ——— 00000000000000000000000
   —1 ———— 11111111 ——— 00000000000000000000000

4. Indique el numero de punto flotante (IEEE) de precisión doble asociado a x, el cual se denota como $fl(x)$;para $x(0.4)$
   $x(0.4) = 00111110110011001100110011001101$

5. Error de redondeo En el modelo de la aritmética de computadora IEEE, el error de redondeo relativo no es mas de la mitad del épsilon de maquina:

   $\frac{|fl(x)-x|}{|x|} \leq \frac{1}{2}\epsilon_{maq}$

   Teniendo en cuenta lo anterior, encuentre el error de redondeo para $x = 0.4$

6. Verificar si el tipo de datos básico de R y Python es de precisión doble IEEE y Revisar en R y Phython el format long
   Python y R utilizan, en sus datos básicos, una precisión doble IEEE (64 bits) [2]

7. Encuentre la representación en numero de maquina hexadecimal del numero real 9.4
   La representación hexadecimal es : 5E

8. Encuentre las dos raıces de la ecuación cuadrática $x^2 + 9^{12}x = 3$ Intente resolver el problema usando la aritmética de precisión doble, tenga en

cuenta la perdida de significancia y debe contra restarla.

De acuerdo a Symbolab, estas son las raíces de la ecuación:

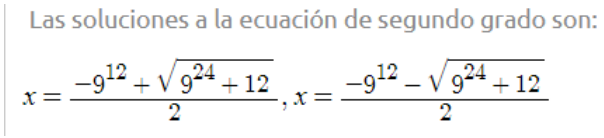


Figure 1: Ecuaciones de raíces

Valores de las raíces: $x = 0, x = -282429536481$

9. Explique como calcular con mayor exactitud las raıces de la ecuación: $x^2 + bx - 10^{-12} = 0$ Donde b es un numero mayor que 100

   Para lograr una mayor exactitud a la hora de calcular las raíces de esta ecuación, lo que consideramos mejor para resolver este inconveniente es utilizar algún método iterativo que tenga convergencia en la raíz, como lo podría ser el método de Muller o el de Newton, entre otros, ayudado con el hecho de que se tendría el apoyo de poder evaluar la tolerancia y de esta manera aumentar la precisión.

2 Raíces de una ecuación

- **2.1** Para comprobar veracidad del algoritmo se realizo tres pruebas haciendo uso de matrices cuadradas, donde se tiene n por m elementos, el número de filas es igual al número columnas, es decir, n = m y se dice, entonces, que la matriz es de orden n.

Matrices=An, Bn, Cn
m

An, con n = 2

	[,1]	[,2]
[1,]	1	3
[2,]	2	4

Bn, con n = 3

	[,1]	[,2]	[,3]
[1,]	1	4	7
[2,]	2	5	8
[3,]	3	6	9

Cn, con n = 4

	[,1]	[,2]	[,3]	[,4]
[1,]	1	5	9	13
[2,]	2	6	10	14
[3,]	3	7	11	15
[4,]	4	8	12	16

Mediante las funciones lower y upper implementadas en en la programación R - inferior, obtenemos una matriz con un triángulo inferior o superior como valores VERDADEROS. para devolver una matriz de valores lógicos con el triángulo inferior o superior como VERDADERO. Cabe mencionar que se valida

Los resultados obtenidos para la suma de la matriz triangular superior e inferior se evidencian a continuación, se presentan cada caso.

	n	SumaTraingularSup	SumaTriangularInf
An	2	2	3
Bn	3	11	19
Cn	4	36	66

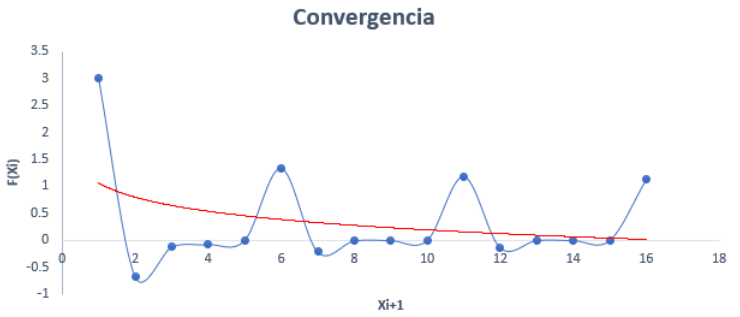
Figure 2: Valores obtenidos de la suma de la matriz triangular

Inicialmente se considera la representación lineal de la función para resolver problemas de convergencia, $A\mathbf{X} = B$, observamos que la solución se representa en $\mathbf{X}$ mediante la evaluación de la función en la iteración anterior.

La expresión se puede obtener una sucesión convergente, mediante el método de Jacobi podemos resolver es sistema $AX = B$ para obtener la solución de manera iterativa. entonces llegamos a la siguiente expresión

$$X = M^{-}1 * (B + NX)$$

Como resultado obtenemos una gráfica que representa la convergencia



Representa la grafica de convergencia $X = M^{-}1 * (B + NX)$

- **2.2** En primer lugar se sebe considerar la sucesión que se obtiene al realizar n^2 para cada uno de los numero naturales hasta n,

Sucesión para n = 5

```
> Sucesion
[1]  1  4  9 16 25
```

Sucesión para n = 10

```
> Sucesion
 [1]   1   4   9  16  25  36  49  64  81 100
```

Sucesión para n = 15

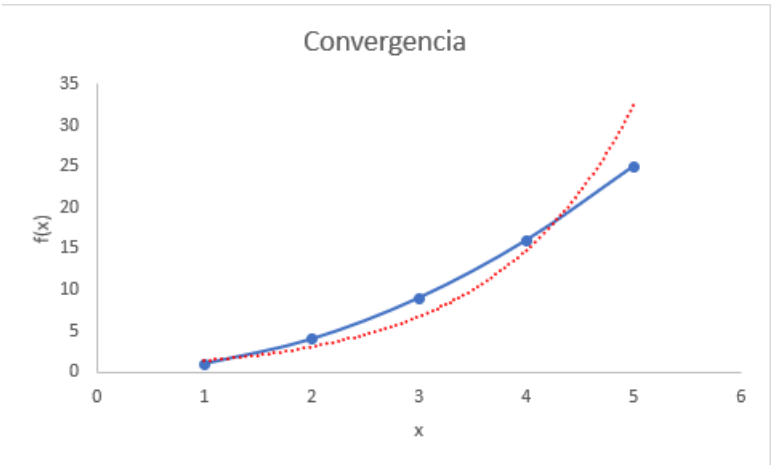
```
> Sucesion
 [1]   1   4   9  16  25  36  49  64  81 100 121 144 169 196 225
```

- Mediante la funcionalidad sum (), que nos permite calcular la suma de las sucesiones que están representada en el vector sucesión, obtenemos los siguientes resultados, tabulados respectivamente.

Resultados obtenidos

n	Resultado Suma
5	55
10	385
15	1240

- De esta manera obtenemos una gráfica donde se evidencia su convergencia



- **2.3** Para realizar el calculo de la altura máxima del cohete dada su expresión de trayectoria, primero se debe considera que dicha trayectoria se representa como una curva en un plano por lo tanto presentara un punto critico que llamaremos máximo, esto se debe a que la expresión es del tipo polinómico que nos permite realizar estas conjeturas.
  Para halar el máximo de la función $y(t) = 6 + 2.13t^2 - 0.0013t^4$ lo primero que se debe hacer es derivar dicha expresión, utilizando el método de Hornear con su deriva realizamos aproximaciones a su raíz, pero se consideran los máximos resultados al evaluar la función

en cada iteración, de este modo se obtiene el siguiente resultado.

Para la expresión de la trayectoria $y(t) = 6 + 2.13t^2 - 0.0013t^4$ El resultado para encontrar su altura máxima es el siguiente

```
El numero de multiplicaciones con horner es  4
La altura maxima del cohete es de 660.78
```

3 Convergencia de métodos iterativos

- Sean $f(x) = ln(x+2)$ y $g(x) = sin(x)$ dos funciones de valor real
  - Utilice la siguiente fórmula recursiva con E = 10^{-18} para el punto de intersección

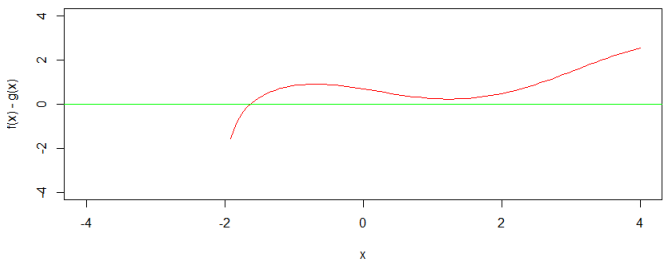
$$x_n = x_{n-1} - \frac{f(x_{n-1})(x_{n-1} - x_{n-2})}{f(x_{n-1}) - f(x_{n-2})}$$

  - Aplicar el método iterativo siguiente con E = 10^{-18} para encontrar el punto de intersección

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$$

3.1 Solución:

Primero es necesario graficar las funciones $f(x)$ y $g(x)$ (restadas) para saber el punto de intersección de las funciones



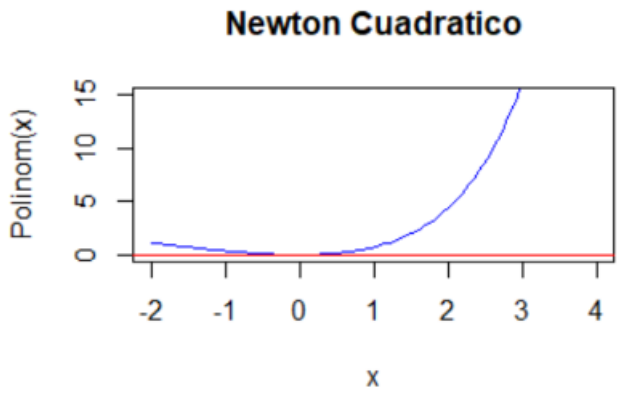
Para después hacer la comparativa entre el método iterativo y el método recursivo y así determinar cual es mejor que el otro en términos de optimización

	errorRec1	errorRecSig	errorIt	errorItsig
1	0.2685564	0.6314436	0.2685564	0.6314436
2	0.6314436	0.2868540	0.6314436	0.2868540
3	0.2868540	0.3703587	0.2868540	0.3703587
4	0.3703587	NA	0.3703587	NA

Con esta tabla en la cual el dato "errorRec1" se refiere el error del método recursivo, "errorRecSig" es el error de la siguiente posición del erroRec1 , "errorIt" es el error del método iterativo y de esa misma forma "errorItsig" es el continuo al error iterativo . Dados los datos que lanza la tabla , se puede concluir que los dos métodos son igual de ópitmos puesto que los datos arrojados son iguales .

– Con el método de newton demuestre que $f(x) = (e^x) - x - 1$. utilizando el método Newton con p0=1 verifique que converge a cero, pero no de forma cuadrática.

3.2 Solución:

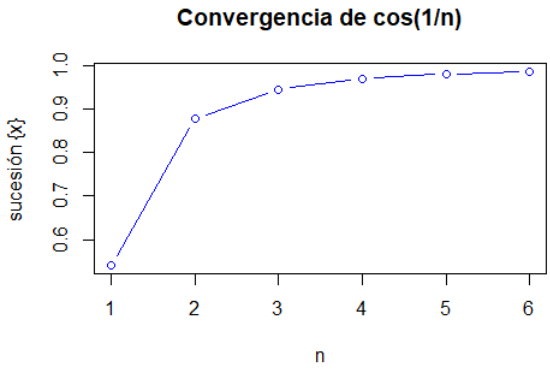


4 Convergencia acelerada

Dada la sucesión $\{x_n\}_n^\infty$ con $x_n = cos(\frac{1}{n})$

4.1 Ejercicio 1: Verifique el tipo de convergencia en x = 1

Para la ecuación dada, el proceso de evaluación para la convergencia consto en ir cambiando el valor de n de 1 hasta infinito, esto mostró que el valor de n, entre mas grande sea la sucesión va tender a converger el valor a 1, esto se puede ver en la siguientes imágenes.



Grafica de $x_n = cos(\frac{1}{n})$

```
Valor de la sucesion  0.5403023  con n =  1
Valor de la sucesion  0.8775826  con n =  2
Valor de la sucesion  0.9449569  con n =  3
Valor de la sucesion  0.9689124  con n =  4
Valor de la sucesion  0.9800666  con n =  5
Valor de la sucesion  0.9861432  con n =  6
```

Resultados de $x_n = cos(\frac{1}{n})$

4.2 Ejercicio 2: Comparar los primeros términos con la sucesión $\{A_n\}_n^\infty$

Como indica la ecuación de Δ^2 Aitken, esta permite acelerar la convergencia de una sucesión de una forma cuadrática, esta ecuación es la siguiente:

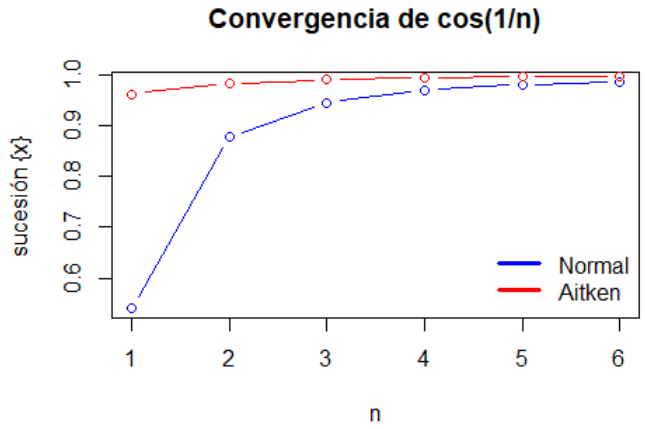
$$\{A_n\}_n^\infty = x_{n+2} - \frac{(x_{n+2} - x_{n+1})^2}{x_{n+2} - 2x_{n+1} + x_n} \tag{1}$$

Teniendo en cuenta como se evalúan las sucesiones y comparando ese método con el de Δ^2 Aiken, se pude ver a simple vista que la convergencia se vuelve cuadrática, permitiendo así evaluar de una forma mas rápida la convergencia de una sucesión cualquiera. Para ver esto de una forma mas precisa se realizo la comparación de convergencia entre los dos métodos para la ecuación del principio. El resultado es el siguiente:

```
Valor de la sucesion  0.5403023 Co Aitken 0.9617751  con n =  1
Valor de la sucesion  0.8775826 Co Aitken 0.9821294  con n =  2
Valor de la sucesion  0.9449569 Co Aitken 0.9897855  con n =  3
Valor de la sucesion  0.9689124 Co Aitken 0.9934156  con n =  4
Valor de la sucesion  0.9800666 Co Aitken 0.9954099  con n =  5
Valor de la sucesion  0.9861432 Co Aitken 0.99662  con n =  6
```

Resultados de $x_n = cos(\frac{1}{n})$

En la figura anterior se puede observar como desde la primera iteración, donde el valor de n es igual a 1, el resultado se aproxima más a la convergencia de la ecuación $x_n = cos(\frac{1}{n})$. Esto comprueba que el método Δ^2 Aitken permite una mayor rapidez al valor de convergencia de cualquier sucesión. Esto se puede ver de una forma más clara en la siguiente gráfica



Aitken Vs Sucesión normal

4.3 Ejercicio 3:encontrar coordenadas de coincidencia entre f(t) y g(t) para $t \geq 0$

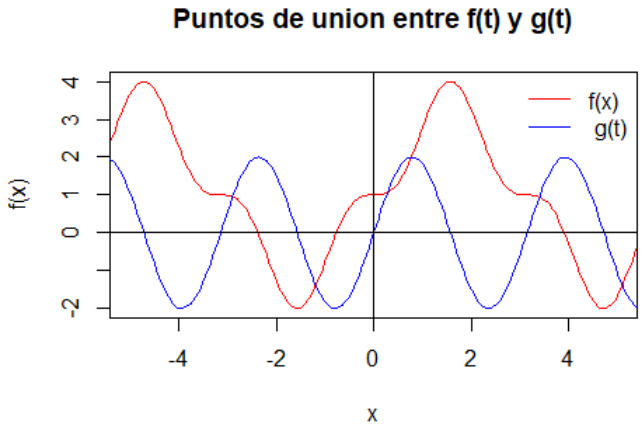
$$f(t) = 3 \cdot sin^3(t) - 1g(t) \tag{2}$$

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

Para poder realizar la búsqueda del punto donde las ecuaciones conicidad de una forma fácil lo que se hizo fue igualar las dos funciones $f(t) = g(t)$ y así poder definir una nueva función la cual se de denominara $p(t) = f(t) - g(t)$, luego de esto se pude utilizar un método como el Newton para buscar raíces de la nueva función, esto es posible ya que $p(x)$ contiene la igualación de las dos funciones, entonces el valor de las raíces va a ser el valor en el eje x donde las funciones $f(t)$ y $g(t)$ pueden coincidir. El procedimiento hecho en Rstudio mostró el siguiente resultado:

```
El valor de x es  0.2803467
 el valor de  f(x): 1.063547
 el valor de  g(x): 1.063547
```

Resultados de puntos de corte



Gráficas $f(x)$ y $g(x)$

4.4 Ejercicio 4: comparar Aitken con Steffensen

El metodo de Steffensen, es un método el cual busca encontrar la raíz de una función, este se basa en el método de Aitken con la combinación de el punto fijo, es un algoritmo muy simple, pero lo que hace es mejorar de forma cuadrática para encontrar la raíz de la función, este método fue implementado con la ayuda del libro de Numerical Analisis, donde se explicaba de forma detalla el algoritmo e indicaba como es la base matemática de lo que realiza el método, esto es :

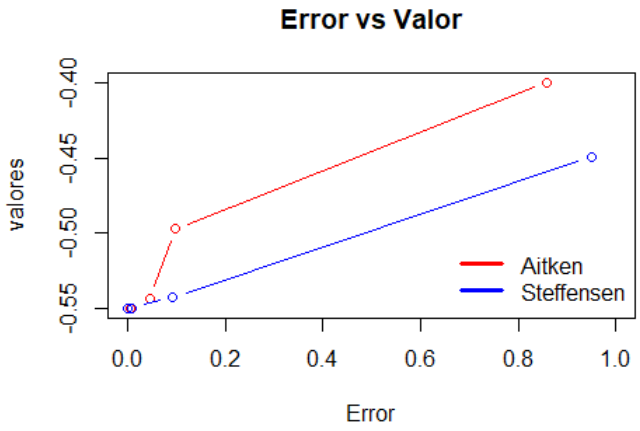
$$p_1 = f(p_0), p_2 = f(p_1), p0 = \Delta^2(p_0) \tag{4}$$

Los métodos de Steffensen y Aitken son completamente diferentes, ya que uno busca es la raíz de una función y el otro busca la convergencia de una sucesión. Pero para poder comparar estos dos métodos se realizo una adaptación del código original de Aitken para que este busque la raíz, ya que si se probaba la función $f(x) = x^2 - cos(x)$(función dada para el ejercicio) esta se quedaba iterado infinitamente porque la función es cuadrática y el valor de convergencia es ∞, mientras que Steffensen mostraba un valor real
Estos algoritmos se probaron con la ayuda del método de Aitken que se encuentra implementado en la librería Pracma en Rstudio, lo cual se hizo con el fin de poder mostrar y comparar el valor correcto o deseado. Para el resultado den ambos casos el valor fue muy similar sin importar si se manejara una toleracina de 10^-8 o $10 - 16$, los valores obtenidos fueron.

```
Tolerancia de  10-8
Aitken
 Iteraciones :  7
> print(a$v)
1 'mpfr' number of precision  120   bits
[1] -0.55000934992726156666495361946834445848
> cat('Steffensem \nIteraciones : ',s$i,'\n')
Steffensem
Iteraciones :  5
> print(s$v)
1 'mpfr' number of precision  120   bits
[1] -0.55000934992726156029573235223326886386
>
> cat('Tolerancia de  10-16\nAitken\n Iteraciones : ',c$i,'\n')
Tolerancia de  10-16
Aitken
 Iteraciones :  8
> print(c$v)
1 'mpfr' number of precision  120   bits
[1] -0.55000934992726156666495361947172922608
> cat('Steffensem \nIteraciones : ',z$i,'\n')
Steffensem
Iteraciones :  6
> print(z$v)
1 'mpfr' number of precision  120   bits
[1] -0.55000934992726156666495361947172922544
```

Resultados de Aitken y Steffensen

Con la imagen anterior se pude observar que el método de Steffensen requiere menos iteraciones para llegar a una valor muy aproximando al que llega Aitken, este resultado se comparo con el método de Pracma mostrando un valor muy similar lo cual indicia que están bien implementados, adicionalmente se pudo obtener la variación del error en cada algoritmo y así poder hacer una comparación, como se puede ver en las siguiente imagen el método de requiere de una iteración mas para poder llegar a un valor mas aproximado en comparación con el de Steffensen, lo cual indica que el segundo muestra una mejora en la convergencia de la raíz.



Resultados de Aitken y Steffensen

References

[1] Universidad de Buenos Aires. Numero de Punto Flotante, 2011.Recuperado de http://materias.fi.uba.ar/6601/comaflotante.pdf

[2] Andres Marzal, Isabel Gracia. Introduccion a la Programacion con Python, Recuperado de http://repositori.uji.es/xmlui/bitstream/handle/10234/24305/s23.pdf.