Pontificia Universidad Javeriana
Departamento de Matemática
Análisis Numérico

Taller 1

Métodos: Ejercicio de raíces
por

Laura Mariana Jiménez Jiménez
la.jimenez@javeriana.edu.co

Paula Valentina Sanchez Peña
pa-sanchez@javeriana.edu.co

Sebastián Gutiérrez Zambrano
sebastian_gutierrez@javeriana.edu.co
Profesora: Eddy Herrera Daza

27 de septiembre de 2020

1. Método Newton-Raphson

1.1. Introducción al método

Método de Newton-Raphson es una eficiente técnica para encontrar las raíces de una función. Como se usa en los cálculos diferenciales, este se basa en una aproximación lineal la cual describe que en una función $f(x)$ con r una raíz de la ecuación $f(x) = 0$ se puede estimar un valor aproximado a r empezando desde un valor inicial x_0 y que con este se puede estimar x_1 tal que este es un valor aún más aproximado al valor de r y, a partir de este x_1 se puede estimar un x_2 todavía más cercano a r y se continua de esta manera hasta que se llegue a un valor estimado lo suficientemente cercano a r. Lo anterior es un método iterativo que usa el método de Newton-Raphson.

El método dice que se parte de un x_0 el cual es un valor estimado de r en donde $r = x_0 + h$. Dado que el valor verdadero es r y $h = r - x_0$, el número h mide qué tan lejano se encuentra el valor estimado x_0 del valor verdadero.

Dado que h es un valor muy "pequeño"se puede usar la aproximación lineal por tanto se concluye que

$$0 = f(r) = f(x_0 + h) \approx f(x_0) + hf'(x_0) \tag{1}$$

y a menos de que $f'(x_0)$ sea muy cercano a 0

$$h = -\frac{f(x_0)}{f'(x_0)} \tag{2}$$

Con esto se puede reemplazar h en la expresión que se mencionó anteriormente:

$$r = x_0 + h \approx x_0 - \frac{f(x_0)}{f'(x_0)} \tag{3}$$

De esta manera si hay un nuevo valor estimado x_1 que se acerca aún más a r, este es expresado de esta manera:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{4}$$

Y de esta manera se puede decir que el valor estimado x_{n+1} es calculado a partir del valor estimado actual x_n lo cual finalmente el método de Newton-Raphson se expresa de la siguiente manera:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{5}$$

1.2. Preguntas

- **¿Cuáles son condiciones para aplicar el método?**

  Para aplicar el método la función debe cumplir las siguientes condiciones:

  Ser derivable y su derivada debe ser diferente y lejana de cero. Si se quiere aplicar el algoritmo de Aitken la función debe ser derivable dos veces. (dentro del intervalo [a,b] que se escoge).

Debe haber raíz, por lo cual teniendo un intervalo [a, b], f(a)*f(b) <0. Esto nos indica que f(a) y f(b) tiene signos diferentes.

La función debe ser cóncava hacía arriba o hacía abajo. En el siguiente ejemplo se puede ver que cuando una función no es cóncava, no converge el método.

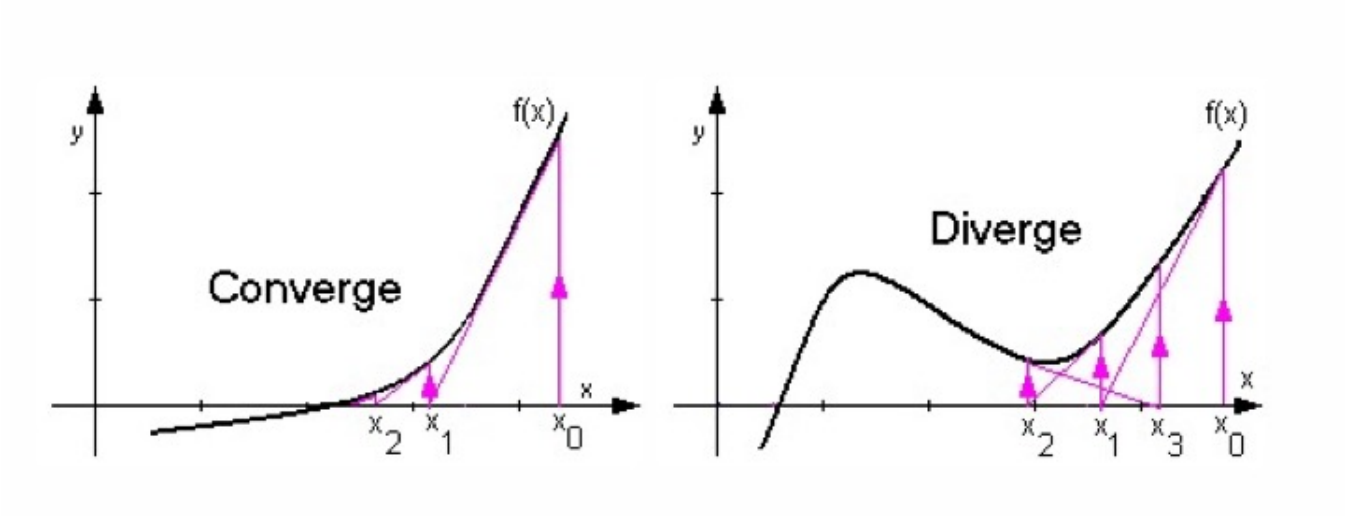

Fig 1. gráficas convergencia

Para verificar dicha concavidad se puede hacer de la siguiente manera.

$$f''(x) <= 0 \vee f''(x) >= 0, \forall x \in [a, b] \tag{6}$$

Para que el método converja, la tangente en x (cualquier número entre a y b) debe intersecarse con el eje x, Como se muestra en la imagen.

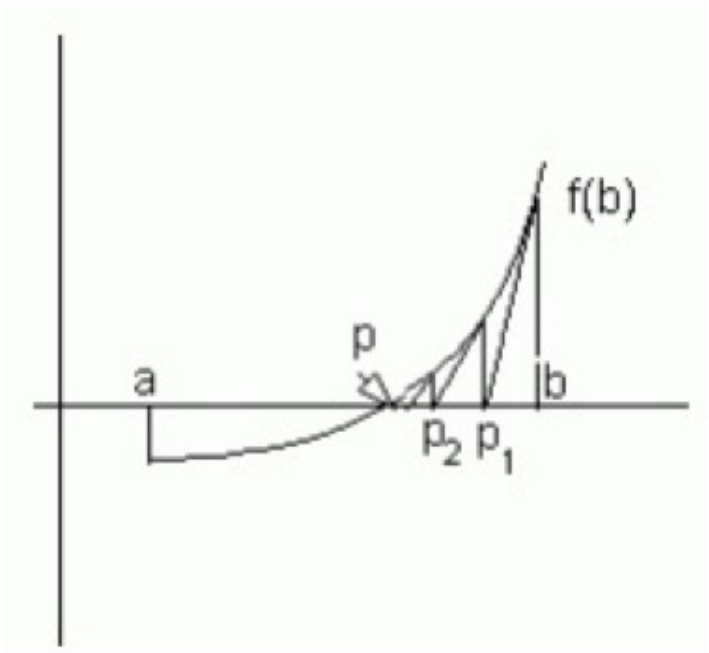

Fig 2. gráfica de convergencia

Esto se puede verificar realizando la siguiente ecuación

$$\frac{\mid f(c) \mid}{\mid f'(c) \mid} <= (b - a) \tag{7}$$

- **Proporcione una explicación geométrica del algoritmo**

  Sea $y = f(x)$ que se encuentra con el eje x en r y a la estimación actual de r. La linea tangente $y = f(x)$ en el punto $(a, f(a))$ tiene ecuación

$$y = f(a) + (x - a)f'(a) \tag{8}$$

  Y b es la intersección con el eje x de la linea tangente representado así:

$$b = a\frac{f(a)}{f'(a)} \tag{9}$$

  Gráficamente podría verse de esta manera:

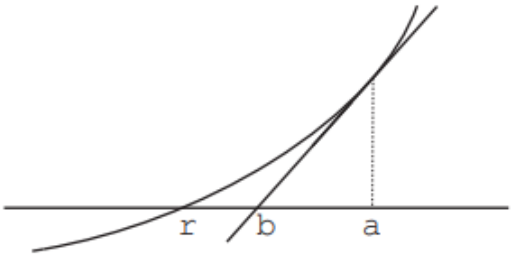

Fig 3. gráfica interpretación geométrica

Como muestra la figura, b es solo una estimación de Newton-Raphson de r. La nueva estimación b se obtiene trazando la recta tangente en $x = a$, y luego deslizándose hacia el eje x a lo largo de esta línea tangente. Después se dibuja la línea tangente en $(b, f(b))$ y se recorre la nueva línea tangente al eje x para obtener una nueva estimación C y se repite el proceso hasta hallar la aproximación más cercana.

- **Realice un diagrama de flujo que muestre como se debe operar el algoritmo**

  A continuación se presenta el diagrama de flujo del método de Newton-Raphson y como se comporta el algoritmo. Se tienen a las principales entradas: $\boldsymbol{f}$ que representa la función de la cual se quiere hallar sus raíces, $\boldsymbol{a}$ y $\boldsymbol{b}$ que representan el intervalo de la función en donde se quiere encontrar las raíces, juntos representan un valor inicial por el cual parte el algoritmo, $\boldsymbol{n}$ representa el número de iteraciones máximo que quiere que se llegue al valor aproximado de las raíces y, finalmente $\boldsymbol{tol}$ representa la tolerancia al error para que sea aceptado el valor aproximado a la respuesta. Después del proceso se presentan las siguientes salidas: $\boldsymbol{i}$ son las iteraciones que le tomó al algoritmo en encontrar la respuesta con mayor precisión y $\boldsymbol{x1}$ representa el valor resultante del algoritmo que representa el valor aproximado de la raíz de la función.

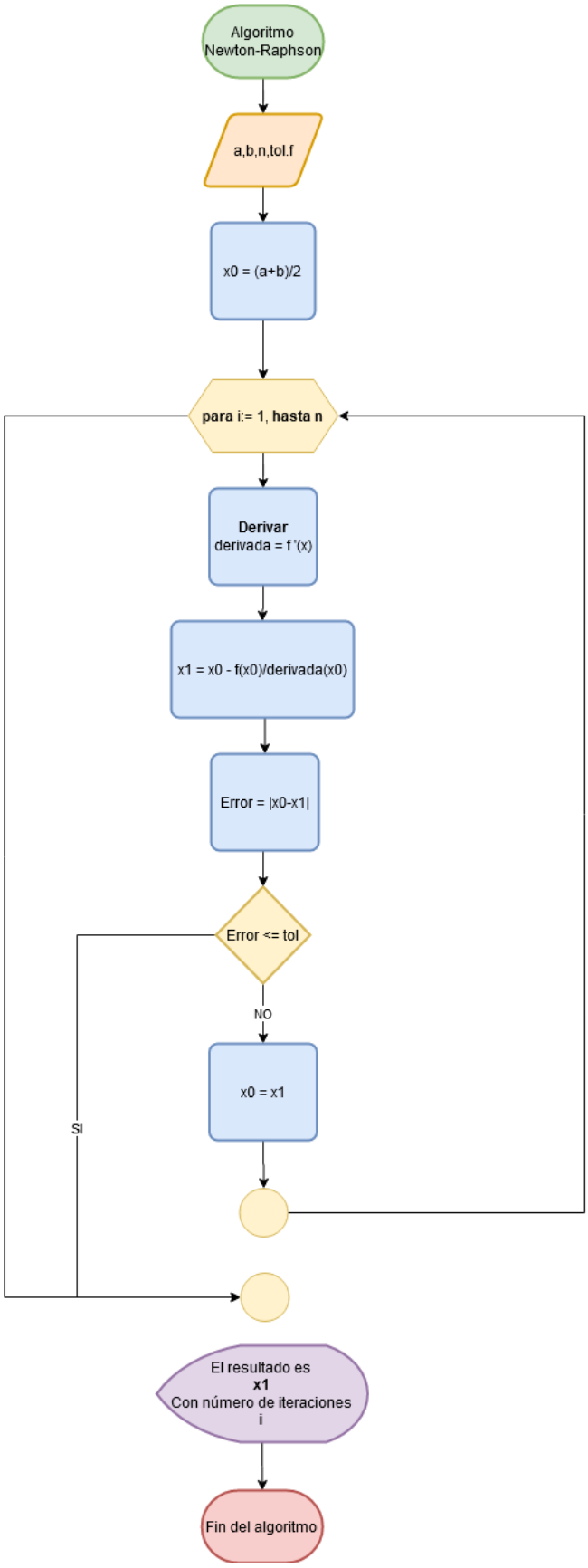

Fig 4. Diagrama de flujo del algoritmo de Newton-Raphson

- **¿Cuál son las raíces? Valide su resultado**
  Método de Newton

  Problema 1

$$f(x) = cos(2x)^2 - x^2 \tag{10}$$

  Tolerancia 10^{-8}

  Para este primer problema se probó con las tres tolerancias dadas. La primera que se va a analizar es una tolerancia de 10^{-8}, se encuentra que la raíz que da el algoritmo para este caso es de 0,514933264661129413801059258436909901619 9 realizando 5 iteraciones. Lo cual se puede contrastar con el resultado dado por Wolfram. Donde se puede observar que una de las raíces da 0.514933

  Tabla de resultados para la función $f(x) = cos(2x)^2 - x^2$ entre los puntos $[0, \frac{3}{2}]$ con una tolerancia de 10^{-8}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.75	-0.5574962483002227	0.3128064925362831
2	0.4371935074637169	0.2203420447288530	0.07750896042940039
3	0.5147024678931174	0.0006452465130544205	0.0002307800678113992
4	0.5149332479609288	4.668594644208213e-08	1.670020060115524e-08
5	0.5149332646611293	2.775557561562891e-16	8.766860431908015e-17

Tabla 1: Tabla de resultados de la función $f(x) = cos(2x)^2 - x^2$

  Tolerancia 10^{-16}

  Para la segunda tolerancia de 10^{-16} la raíz encontrada por el algoritmo fue de
  0.514933264661129413801059258436909901619 9
  realizando 5 iteraciones, a continuación la tabla de resultados.

  Tabla de resultados para la función $f(x) = cos(2x)^2 - x^2$ entre los puntos $[0, \frac{3}{2}]$ con una tolerancia de 10^{-16}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.75	-0.5574962483002227	0.3128064925362831
2	0.4371935074637169	0.220342044728853	0.07750896042940039
3	0.5147024678931176	0.0006452465130544205	0.0002307800678113992
4	0.5149332479609288	4.668594644208213e-08	1.670020060115524e-08
5	0.5149332646611293	2.775557561562891e-16	8.766860431908015e-17

Tabla 2: Tabla de resultados de la función $f(x) = cos(2x)^2 - x^2$

Tolerancia 10^{-36}

Finalmente el primer problema se resolvió también con una tolerancia de 10^{-32} con la cual se aumentó en uno el número de iteraciones (6). La raíz da como resultado
0.514933264661129413801059258436912317575 2

Tabla de resultados para la función $f(x) = cos(2x)^2 - x^2$ entre los puntos $[0, \frac{3}{2}]$ con una tolerancia de 10^{-32}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.75	-0.5574962483002227	0.3128064925362831
2	0.4371935074637169	0.220342044728853	0.07750896042940039
3	0.5147024678931174	0.0006452465130544205	0.0002307800678113992
4	0.5149332479609288	4.668594644208213e-08	1.670020060115524e-08
5	0.5149332646611293	2.775557561562891e-16	8.766860431908015e-17
6	0.5149332646611294	0	2.415955335678646e-33

Tabla 3: Tabla de resultados de la función $f(x) = cos(2x)^2 - x^2$

Resultado por Wolfram

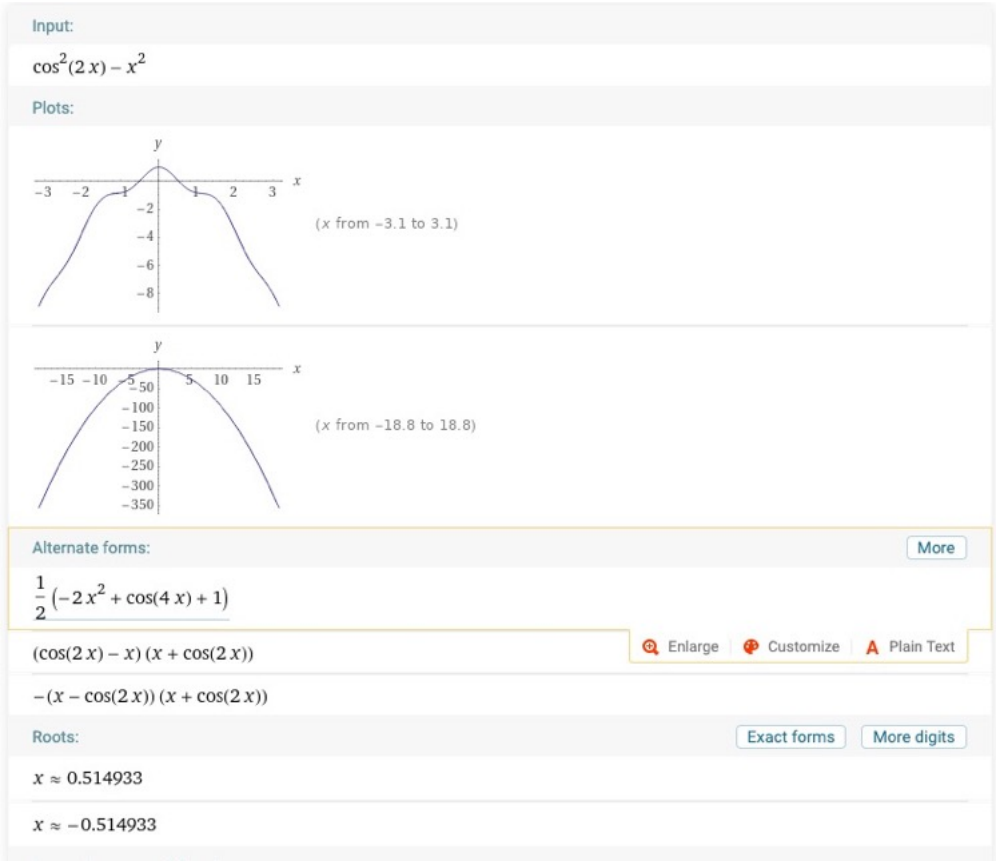

Fig 5. resultado problema1

Problema 2

$$f(x) = x * sin(x) - 1 \tag{11}$$

Tolerancia 10^{-8}

Se encuentra que la raíz que da el algoritmo para este caso es de
1.114157140871930087300525178169203903956
realizando 5 iteraciones. Lo cual se puede contrastar con el resultado dado por Wolfram. Donde se puede observar que una de las raíces da 1.11415714087193...

Tabla de resultados para la función $f(x) = x * sin(x) - 1$ entre los puntos $[-1, 2]$ con una tolerancia de 10^{-8}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.5	-0.7602872306978985	0.8280040340265233
2	1.328004034026523	0.2890542781976755	0.224083305304603
3	1.10392072872192	-0.01422204556605855	0.01023277732312792
4	1.114153506045048	-5.048082454051084e-06	3.634826319932153e-06
5	1.114157140871368	-7.803757640090225e-13	5.618661345273821e-13

Tabla 4: Tabla de resultados de la función $f(x) = x * sin(x) - 1$

Tolerancia 10^{-16}

Con dicha tolerancia el algoritmo concluye que la raíz es de
1.114157140871930087300525178169203903956

Tabla de resultados para la función $f(x) = x * sin(x) - 1$ entre los puntos $[-1, 2]$ con una tolerancia de 10^{-16}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.5	-0.7602872306978985	0.8280040340265233
2	1.328004034026523	0.2890542781976755	0.224083305304603
3	1.10392072872192	-0.01422204556605855	0.01023277732312792
4	1.114153506045048	-5.048082454051084e-06	3.634826319932153e-06
5	1.114157140871368	-7.803757640090225e-13	5.618661345273821e-13
6	1.11415714087193	2.220446049250313e-16	1.342640076688285e-26

Tabla 5: Tabla de resultados de la función $f(x) = x * sin(x) - 1$

Tolerancia 10^{-32}

Disminuyendo la tolerancia el resultado de la raíz es de
1.114157140871930087300525178169203903956

Tabla de resultados para la función $f(x) = x * sin(x) - 1$ entre los puntos $[-1, 2]$ con una tolerancia de 10^{-32}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.5	-0.7602872306978985	0.8280040340265233
2	1.328004034026523	0.2890542781976755	0.224083305304603
3	1.10392072872192	-0.01422204556605855	0.01023277732312792
4	1.114153506045048	-5.048082454051084e-06	3.634826319932153e-06
5	1.114157140871368	-7.803757640090225e-13	5.618661345273821e-13
6	1.11415714087193	2.220446049250313e-16	1.342640076688285e-26
7	1.11415714087193	2.220446049250313e-16	0

Tabla 6: Tabla de resultados de la función $f(x) = x * sin(x) - 1$

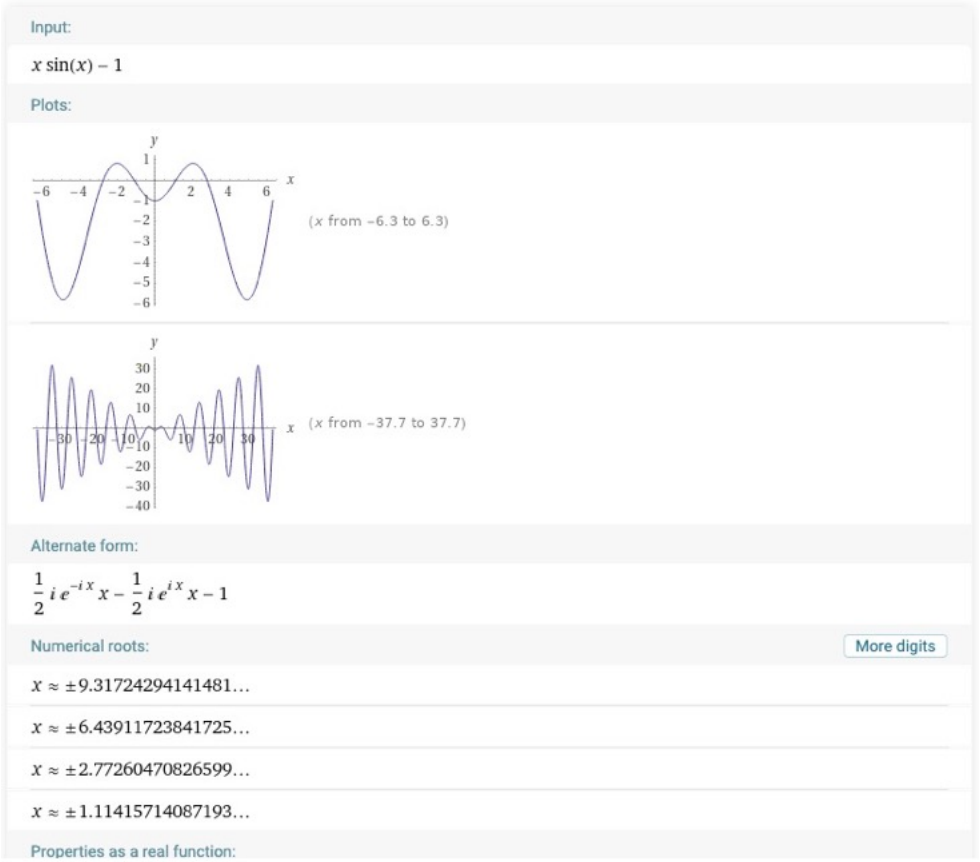

Fig 6. resultado problema2

Problema 3

$$f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27} \tag{12}$$

Tolerancia 10^{-8}

Se encuentra que la raíz que da el algoritmo para este caso es de
0.66666987081929470460568798142275091303 69
realizando 32 iteraciones. Lo cual se puede contrastar con el resultado dado por Wolfram. Donde se puede observar que la raíz da 0.66667

Tabla de resultados para la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ entre los puntos $[0, 2]$ con una tolerancia de 10^{-8}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	1	0.03703703703703698	0.1111111111111109
2	0.8888888888888891	0.01097393689986281	0.07407407407407383
3	0.8148148148148152	0.00325153685921864	0.04938271604938224
4	0.7654320987654329	0.0009634183286573172	0.03292181069958748
5	0.7325102880658455	0.0002854572825652379	0.0219478737997235
6	0.710562414266122	8.457993557464683e-05	0.01463191586647905
7	0.6959304983996429	2.506072165187057e-05	0.009754610577645397
8	0.6861758878219976	7.425399007998656e-06	0.006503073718413978
9	0.6796728141035836	2.20011822460453e-06	0.004335382478906219
10	0.6753374316246774	6.518868813643053e-07	0.00289025498585582
11	0.6724471766388216	1.931516685482748e-07	0.001926836657053955
12	0.6705203399817676	5.723012397318428e-08	0.001284557770957683
13	0.6692357822108099	1.695707374516076e-08	0.000856371846380044
14	0.6683794103644298	5.024318294744035e-09	0.0005709145621735373
15	0.6678084958022563	1.488686729445021e-09	0.0003806097034384853
16	0.6674278860988179	4.410924958619944e-10	0.000253739791772211
17	0.6671741463070456	1.306941221912439e-10	0.0001691598375166254
18	0.667004986469529	3.872413500971561e-11	0.0001127731717732978
19	0.6668922132977557	1.147371087029114e-11	7.518199474282684e-05
20	0.6668170313030128	3.399613923704692e-12	5.012106035940353e-05
21	0.6667669102426534	1.007305350242405e-12	3.34134339705358e-05
22	0.666733496808683	2.984279490192421e-13	2.227425865226685e-05
23	0.6667112225500307	8.837375276016246e-14	1.484643737010649e-05
24	0.6666963761126605	2.620126338115369e-14	9.890725121937056e-06
25	0.6666864853875386	7.771561172376096e-15	6.578322766946832e-06
26	0.6666799070647716	2.331468351712829e-15	4.350916829534097e-06
27	0.6666755561479422	6.661338147750939e-16	2.824399148946189e-06
28	0.6666727317487932	2.220446049250313e-16	1.723604961062276e-06
29	0.6666710081438322	1.110223024625157e-16	8.653994635635709e-07
30	0.6666701427443685	0	2.512076879097136e-07
31	0.6666698915366807	1.110223024625157e-16	2.058458232551137e-08
32	0.6666698709520983	1.110223024625157e-16	1.328036332778127e-10

Tabla 7: Tabla de resultados de la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$)

Tolerancia 10^{-16}

Con esta tolerancia, la raíz da
0.6666698708192892000817759409978696760077
Y el número de iteraciones pasa de 32 a 34

Tabla de resultados para la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ entre los puntos $[0, 2]$ con una tolerancia de 10^{-16}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	1	0.03703703703703698	0.1111111111111109
2	0.8888888888888891	0.01097393689986281	0.07407407407407383
3	0.8148148148148152	0.00325153685921864	0.04938271604938224
4	0.7654320987654329	0.0009634183286573172	0.03292181069958748
5	0.7325102880658455	0.0002854572825652379	0.0219478737997235
6	0.710562414266122	8.457993557464683e-05	0.01463191586647905
7	0.6959304983996429	2.506072165187057e-05	0.009754610577645397
8	0.6861758878219976	7.425399007998656e-06	0.006503073718413978
9	0.6796728141035836	2.20011822460453e-06	0.004335382478906219
10	0.6753374316246774	6.518868813643053e-07	0.00289025498585582
11	0.6724471766388216	1.931516685482748e-07	0.001926836657053955
12	0.6705203399817676	5.723012397318428e-08	0.00128455770957683
13	0.6692357822108099	1.695707374516076e-08	0.000856371846380044
14	0.6683794103644298	5.024318294744035e-09	0.0005709145621735373
15	0.6678084958022563	1.488686729445021e-09	0.0003806097034384853
16	0.6674278860988179	4.410924958619944e-10	0.000253739791772211
17	0.6671741463070456	1.306941221912439e-10	0.0001691598375166254
18	0.667004986469529	3.872413500971561e-11	0.0001127731717732978
19	0.6668922132977557	1.147371087029114e-11	7.518199474282684e-05
20	0.6668170313030128	3.399613923704692e-12	5.012106035940353e-05
21	0.6667669102426534	1.007305350242405e-12	3.34134339705358e-05
22	0.666733496808683	2.984279490192421e-13	2.227425865226685e-05
23	0.6667112225500307	8.837375276016246e-14	1.484643737010649e-05
24	0.6666963761126605	2.620126338115369e-14	9.890725121937056e-06
25	0.6666864853875386	7.771561172376096e-15	6.578322766946832e-06
26	0.6666799070647716	2.331468351712829e-15	4.350916829534097e-06
27	0.6666755561479422	6.661338147750939e-16	2.824399148946189e-06
28	0.6666727317487932	2.220446049250313e-16	1.723604961062276e-06
29	0.6666710081438322	1.110223024625157e-16	8.653994635635709e-07
30	0.6666701427443685	1.110223024625157e-16	2.512076879097136e-07
31	0.6666698915366807	1.110223024625157e-16	2.058458232551137e-08
32	0.6666698709520983	1.110223024625157e-16	1.328036332778127e-10
33	0.6666698708192947	0	5.504523902583903e-15
34	0.6666698708192892	0	9.456521864019275e-24

Tabla 8: Tabla de resultados de la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27})$

Tolerancia 10^{-32}

FInalmente con la tolerancia más pequeña de 10^{-32} , la raíz da
0.6666698708192892000817759410932840828687

Y el número de iteraciones pasa de 34 a 36

Tabla de resultados para la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ entre los puntos $[0, 2]$ con una tolerancia de 10^{-32}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	1	0.03703703703703698	0.1111111111111109
2	0.8888888888888891	0.01097393689986281	0.07407407407407383
3	0.8148148148148152	0.00325153685921864	0.04938271604938224
4	0.7654320987654329	0.0009634183286573172	0.03292181069958748
5	0.7325102880658455	0.000285457282565238	0.0219478737997235
6	0.710562414266122	8.457993557464683e-05	0.01463191586647905
7	0.6959304983996429	2.506072165187057e-05	0.009754610577645397
8	0.6861758878219976	7.425399007998656e-06	0.006503073718413978
9	0.6796728141035836	2.20011822460453e-06	0.004335382478906219
10	0.6753374316246774	6.518868813643053e-07	0.00289025498585582
11	0.6724471766388216	1.931516685482748e-07	0.001926836657053955
12	0.6705203399817676	5.723012397318428e-08	0.001284557770957683
13	0.6692357822108099	1.695707374516076e-08	0.000856371846380044
14	0.6683794103644298	5.024318294744035e-09	0.0005709145621735373
15	0.6678084958022563	1.488686729445021e-09	0.0003806097034384853
16	0.6674278860988179	4.410924958619944e-10	0.000253739791772211
17	0.6671741463070456	1.306941221912439e-10	0.0001691598375166254
18	0.667004986469529	3.872413500971561e-11	0.0001127731717732978
19	0.6668922132977557	1.147371087029114e-11	7.518199474282684e-05
20	0.6668170313030128	3.399613923704692e-12	5.012106035940353e-05
21	0.6667669102426534	1.007305350242405e-12	3.34134339705358e-05
22	0.666733496808683	2.984279490192421e-13	2.227425865226685e-05
23	0.6667112225500307	8.837375276016246e-14	1.484643737010649e-05
24	0.6666963761126605	2.620126338115369e-14	9.890725121937056e-06
25	0.6666864853875386	7.771561172376096e-15	6.578322766946832e-06
26	0.6666799070647716	2.331468351712829e-15	4.350916829534097e-06
27	0.6666755561479422	6.661338147750939e-16	2.82439914894619e-06
28	0.6666727317487932	2.220446049250313e-16	1.723604961062276e-06
29	0.6666710081438322	1.110223024625157e-16	8.653994635635709e-07
30	0.6666701427443685	1.110223024625157e-16	2.512076879097136e-07
31	0.6666698915366807	1.110223024625157e-16	2.058458232551137e-08
32	0.6666698709520983	1.110223024625157e-16	1.328036332778127e-10
33	0.6666698708192947	0	5.504523902583903e-15
34	0.6666698708192892	0	9.456521864019275e-24
35	0.6666698708192892	0	9.54144068609456e-29
36	0.6666698708192892	0	0

Tabla 9: Tabla de resultados de la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$)

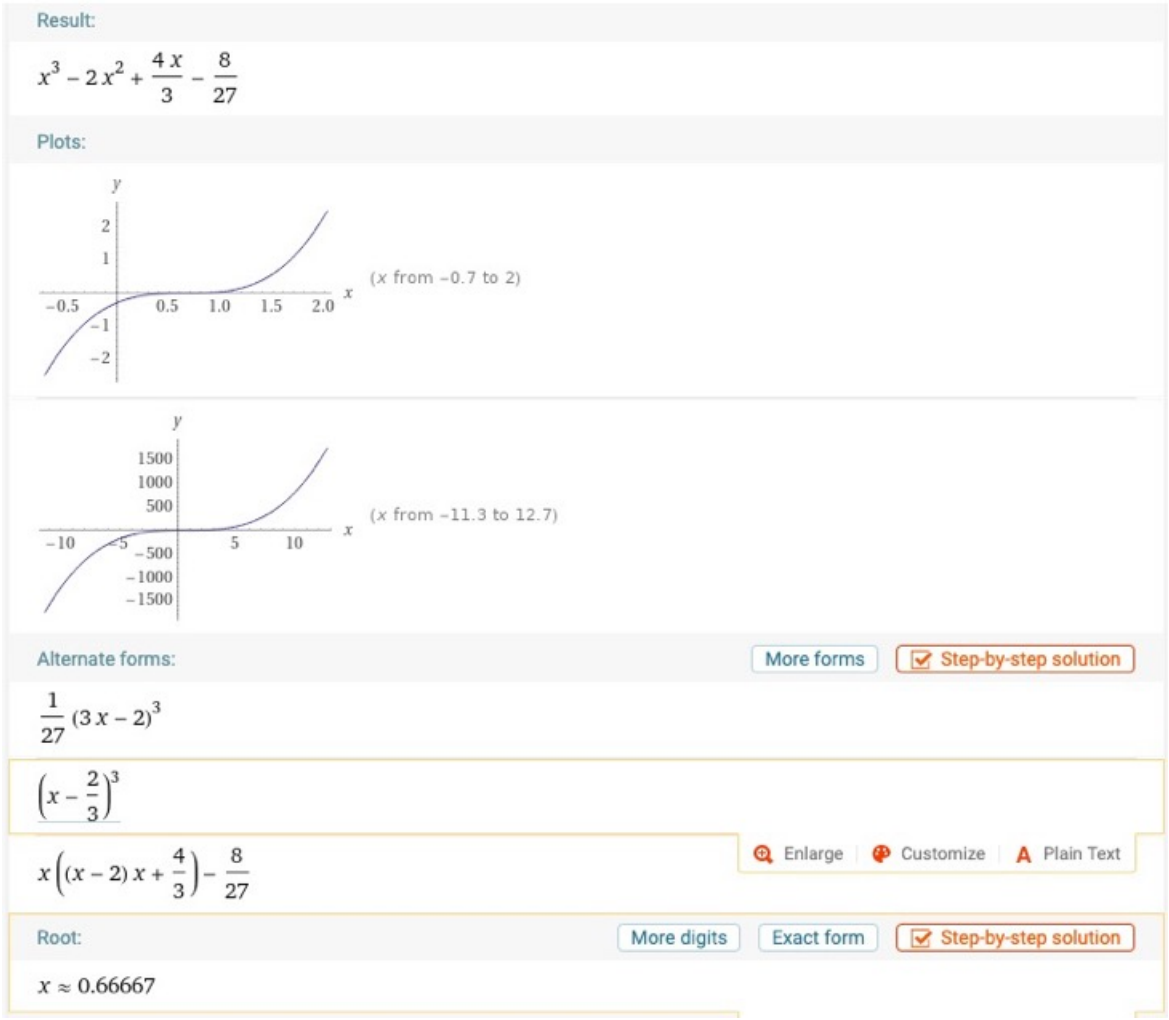

Fig 7. Resultado problema 1

Método mejorado con Aitken

A continuación se muestran los resultados del algoritmo de Newton Rahpson acelerado con Aitken, donde se puede ver que en todos los casos se reduce el número de iteraciones, significativamente se puede ver en el último que pasa de hacer 36 iteraciones a solo 3.

Problema 1

$$f(x) = cos(2x)^2 - x^2 \tag{13}$$

Tolerancia 10^{-8}

El algoritmo de Aitken para este caso, es útil para disminuir el número de iteraciones, en este caso se debería evaluar si realmente vale la pena, ya que este número solo disminuye en uno.

Tabla de resultados para la función $f(x) = cos(2x)^2 - x^2$ entre los puntos $[0, \frac{3}{2}]$ con una tolerancia de 10^{-8}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.75	-0.5574962483002227	0.3128064925362831
2	0.4371935074637169	0.2203420447288530	0.07750896042940042
3	0.5149339371521008	-1.879969667717418e-06	6.724911135735923e-07
4	0.5149332646609872	3.975708651182686e-13	1.422027422634871e-13

Tabla 10: Tabla de resultados de la función $f(x) = cos(2x)^2 - x^2$

Tolerancia 10^{-16}

Para esta tolerancia más pequeña tenemos que la raíz nos da más cifras significativas siendo 0.514933264661129413801059258436912317575 2

Tabla de resultados para la función $f(x) = cos(2x)^2 - x^2$ entre los puntos $[0, \frac{3}{2}]$ con una tolerancia de 10^{-16}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.75	0.3128064925362831	-0.5574962483002227
2	0.4371935074637169	0.07750896042940042	0.220342044728853
3	0.5149339371521008	6.724911135735923e-07	-1.879969667717418e-06
4	0.5149332646609872	1.422027422634871e-13	3.975708651182686e-13
5	0.5149332646611294	6.356480771724951e-27	0

Tabla 11: Tabla de resultados de la función $f(x) = cos(2x)^2 - x^2$

Tolerancia 10^{-32}

Finalmente para esta tolerancia más pequeña, se tiene que la raíz da 0.514933264661129413801059258436912317575 2

Tabla de resultados para la función $f(x) = cos(2x)^2 - x^2$ entre los puntos $[0, \frac{3}{2}]$ con una tolerancia de 10^{-32}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.75	0.3128064925362831	-0.5574962483002227
2	0.4371935074637169	0.07750896042940042	0.220342044728853
3	0.5149339371521008	6.724911135735923e-07	-1.879969667717418e-06
4	0.5149332646609872	1.422027422634871e-13	3.975708651182686e-13
5	0.5149332646611294	6.356480771724951e-27	0

Tabla 12: Tabla de resultados de la función $f(x) = cos(2x)^2 - x^2$

Problema 2

$$f(x) = x * sin(x) - 1 \tag{14}$$

Tolerancia 10^{-8}

Para este siguiente problema la raíz dió
1.114157140871930084183058152699082259792

Tabla de resultados para la función $f(x) = x * sin(x) - 1$ entre los puntos $[0, \frac{3}{2}]$ con una tolerancia de 10^{-8}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.5	0.8280040340265233	-0.7602872306978985
2	1.328004034026523	0.224083305304603	0.2890542781976755
3	1.113706632187435	0.0004505001229131445	-0.0006256826268691285
4	1.114157132310348	8.561582009117027e-09	-1.189040543803088e-08

Tabla 13: Tabla de resultados de la función $f(x) = x * sin(x) - 1$

Tolerancia 10^{-16}

La raíz con esta toleracia da
1.114157140871930087300525178169203903538

Tabla de resultados para la función $f(x) = x * sin(x) - 1$ entre los puntos $[-1, 2]$ con una tolerancia de 10^{-16}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.5	-0.7602872306978985	0.8280040340265233
2	1.328004034026523	0.2890542781976755	0.224083305304603
3	1.113706632187435	-0.0006256826268691285	0.0004505001229131445
4	1.114157132310348	-1.189040543803088e-08	8.561582009117027e-09
5	1.11415714087193	2.220446049250313e-16	3.117467025470122e-18

Tabla 14: Tabla de resultados de la función $f(x) = x * sin(x) - 1$

Tolerancia 10^{-32}

La raíz con esta toleracia da
1.114157140871930087300525178169203903956

Tabla de resultados para la función $f(x) = x * sin(x) - 1$ entre los puntos $[-1, 2]$ con una tolerancia de 10^{-32}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	0.5	0.8280040340265233	-0.7602872306978985
2	1.328004034026523	0.224083305304603	0.2890542781976755
3	1.113706632187435	0.0004505001229131445	-0.0006256826268691285
4	1.114157132310348	8.561582009117027e-09	-1.189040543803088e-08
5	1.11415714087193	3.117467025470122e-18	2.220446049250313e-16
6	1.11415714087193	4.173004945419121e-37	2.220446049250313e-16

Tabla 15: Tabla de resultados de la función $f(x) = x * sin(x) - 1$

Problema 3

$$f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27} \tag{15}$$

En este ultimo ejercicio es donde se ve aún más la eficacia de la aceleración con el algoritmo de Aitken que pasa de necesitar entre 32 y 36 iteraciones dependiendo la tolerancia, a tener solo 3 en todos los casos con las tres tolerancias manejadas, esto se da porque la derivada se acerca a 0 entonces tiene que dejar de correr el algoritmo y dar la respuesta que se tenga. Aún así la respuesta puede ser comparada con el resultado de Wolfram y no son muy alejadas, es bastante acertada.

Esta eficacia se da porque en este caso especifico, el método de Newton tiene una convergencia que se vuelve lineal, en los otros ejercicios su convergencia no es lineal sino cuadrática. El algoritmo de Aitken ayuda a volver esta convergencia lineal en cuadrática para necesitar menos iteraciones para llegar a la respuesta.

Tolerancia 10^{-16}

Tabla de resultados para la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ entre los puntos $[0, 2]$ con una tolerancia de 10^{-8}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	1	0.1111111111111109	0.03703703703703698
2	0.8888888888888891	0.07407407407407396	0.01097393689986281

Tabla 16: Tabla de resultados de la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$)

Tolerancia 10^{-16}

Tabla de resultados para la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ entre los puntos $[0, 2]$ con una tolerancia de 10^{-16}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	1	0.1111111111111109	0.03703703703703698
2	0.8888888888888891	0.07407407407407396	0.01097393689986281

Tabla 17: Tabla de resultados de la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27})$

Tolerancia 10^{-32}

Tabla de resultados para la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ entre los puntos $[0, 2]$ con una tolerancia de 10^{-32}

Número de iteraciones	X_i	$f(x_i)$	ϵ_a
1	1	0.1111111111111109	0.03703703703703698
2	0.8888888888888891	0.07407407407407396	0.01097393689986281

Tabla 18: Tabla de resultados de la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27})$

- **Como se comporta el método en cuanto: perdida de significancia, el número de iteraciones, la convergencia.**

  Significancia

  El método de NewtonRaphson es un método que no es muy sensible a la perdida de significancia, sin embargo, al momento de ejecutarse en algún lenguaje de programación como python o R es posible que se pierda un poco de precisión por la utilización de variables intermedias y la cantidad de iteraciones necesarias para alcanzar un error muy pequeño previamente definido, por lo que se puede decir que el problema de significancia no se presenta por el método si no por la maquina y la tecnología que lo ejecute.

  Número de iteraciones

  Problema 1: Para la ecuación $(x) = cos(2x)^2 - x^2$ se definieron tres tolerancias diferentes, 10^{-8}, 10^{-16} y 10^{-32} con las cuales se ejecuto el método dando como resultados 5 iteraciones y un valor en la raíz de $0,5149332646611294138010592584369099016199$ para la tolerancia 1, 5 iteraciones y un valor en la raíz de 0,5149332646611294138010592584369099016199 para la tolerancia 2 y finalmente 6 iteraciones con un valor de raíz de 0.5149332646611294138010592584369123175752 para la tolerancia 3, por lo que se puede decir que la cantidad de iteraciones no varia mucho según la tolerancia aplicada.

  Problema 2: Para la ecuación $f(x) = xsin(x) - 1$ se definieron tres tolerancias diferentes, 10^{-8}, 10^{-16} y 10^{-32} con las cuales se ejecuto el método dando como resultados 5 iteraciones y un valor en la raíz de 1,1141571408719300873005251781692039039 56 para la tolerancia 1, 6 iteraciones y un valor en la raíz de 1,1141571408719300873005251781692039039 56 para la tolerancia 2 y finalmente 6 iteraciones con un valor de raíz de 1,1141571408719300873005251781692039039 56 para la tolerancia 3. En este caso a medida que la tolerancia aumenta es necesario realizar una iteración mas para llegar al resultado esperado.

Problema 3: Para la ecuación $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ se definieron tres tolerancias diferentes, 10^{-8}, 10^{-16} y 10^{-32} con las cuales se ejecuto el método dando como resultados 32 iteraciones y un valor en la raíz de 0,66666987081929470460568798142275091303 69 para la tolerancia 1, 34 iteraciones y un valor en la raíz de 0,66666987081928920008177594099786960760 77 para la tolerancia 2 y finalmente 36 iteraciones y un valor en la raíz de 0.66666987081928920008177594109328408286 87 para la tolerancia 3. Basándonos en estos valores podemos ver que esta ecuación requirió de muchas más iteraciones en comparación con anteriores, sin embargo, el cambio entre las tolerancias no genero cambios muy extensos en la cantidad adicional de iteraciones necesarias.

Convergencia:

El orden de convergencia de Newton'Raphson es, por lo menos, cuadrático. Sin embargo, si la raíz buscada es de multiplicidad algebraica mayor a uno, el método de pierde su convergencia cuadrática y pasa a ser lineal.

Bajo la anterior afirmación podemos decir que las ecuaciones 1 y 2 tienen un orden de convergencia cuadrático debido a que no presentan multiplicidad algebraica mayor a uno, sin embargo, la ecuación $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ si presenta multiplicidad mayor a uno y en ese caso la convergencia pasa a ser lineal, es decir se requieren más iteraciones para poder llegar al resultado.

- **¿Cómo se puede solucionar el problema de significancia? ¿Es remediable o está destinado al fracaso?**

  Como se mencionó anteriormente, este método no pierde significancia en su algoritmo, sin embargo dependiendo de la herramienta en la que se ejecute el algoritmo, los valores intermedios y el resultado final puede producir pérdida de precisión, lo cuál puede conllevar a hacer varias iteraciones cuando se tiene una tolerancia bastante baja.

  Una solución que se puede usar es prescindir de funciones adicionales que pueden reducir la significancia, como la función de evaluar y derivar una función. El corregimiento de bits no solo ofrece una vigorosidad en el algoritmo si no que también da una mayor precisión. En R existe la librería Rmpf que permite dar la mayor precisión de un número flotante al sugerirle el número de bits; en este caso se tiene una precisión de hasta 40 decimales si se le indica 128 bits al número que se le quiere obtener la una mayor precisión. afectacion

- **¿Qué pasa con el método cuando hay más de dos raíces, explique su respuesta?**

  Cuando hay más de dos raíces el método solo calcula una de las dos raíces tomando el intervalo, si se quiere encontrar la otra raíz, habría que cambiar el intervalo. Es decir dividir el problema. Esto se da por la naturaleza del método de Newton donde se va iterando hasta encontrar una raíz y allí para de iterar, al converger no da pie para hallar nuevas raíces a menos que se cambie el intervalo dado.

- **¿Qué pasa con el método cuando la función es periódica, par o impar, estas características influyen?**

  En Newton-Raphson encontramos que no se encuentra ninguna afectación en el método dependiendo de la naturaleza de la función (par o impar) así como vemos en las ecuaciones 1 y 2

que son de naturaleza par o en la ecuación 3 que es de naturaleza impar, en las tres ecuaciones el método funciona de manera correcta bajo las mismas condiciones. Sin embargo, teniendo en cuenta la ecuación $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$, podemos ver que el método disminuye su convergencia cuando la raíz que quiere aproximar es periódica (en este caso el valor de la raíz es 0.666667) y por esta razón su cantidad de iteraciones aumenta considerablemente.

- **Realice una gráfica que muestre la relación entre ϵ_{i+1}y ϵ_i, qué representa esa gráfica y encuentre una relación de la forma $\epsilon_{i+1} = f(\epsilon_i)$**

  En las siguientes tres figuras representan las gráficas de relación de error entre ϵ_{i+1}y ϵ_i utilizando el método de Newton-Raphson para las funciones $f(x) = cos(2x)^2 - x^2$, $f(x) = xsin(x) - 1$ y $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ respectivamente, con una tolerancia de 10^{-32}.

Fig 8. Gráfica de relación de error $f(x) = cos(2x)^2 - x^2$

Fig 9. Gráfica de relación de error $f(x) = xsin(x) - 1$

Fig 10. Gráfica de relación de error $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$

A continuación se presentan dos figuras que representan las gráficas de relación de error entre ϵ_{i+1}y ϵ_i utilizando el método de Newton-Raphson con convergencia acelerada por el método de Aitken para las funciones $f(x) = cos(2x)^2 - x^2$ y $f(x) = xsin(x) - 1$ respectivamente, con una tolerancia de 10^{-32}.

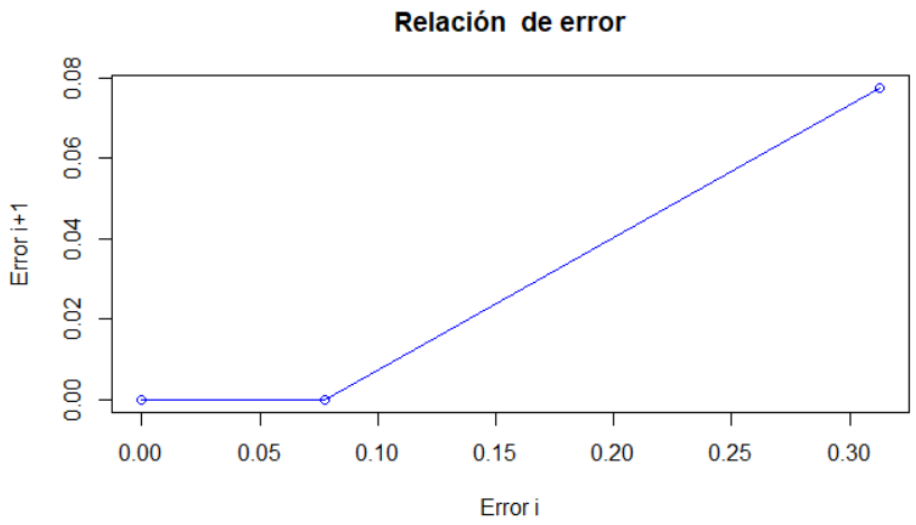

Fig 11. Gráfica de relación de error $f(x) = cos(2x)^2 - x^2$

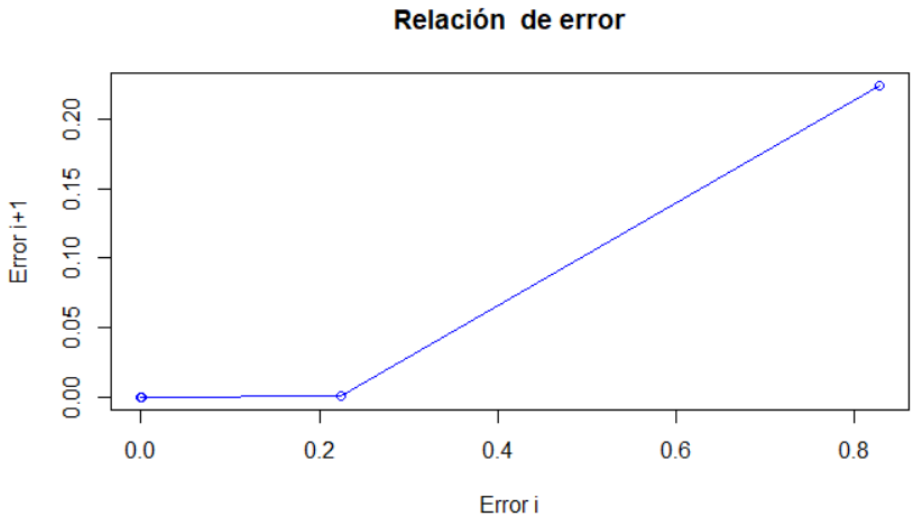

Fig 12. Gráfica de relación de error $xsin(x) - 1$

La relación de la forma $\epsilon_{i+1} = f(\epsilon_i)$ de las gráficas previas representan la convergencia del método. Se puede ver claramente en cualquiera de las gráficas que la convergencia es como se explicó anteriormente. En las gráficas de las funciones $f(x) = cos(2x)^2 - x^2$ y $xsin(x) - 1$, se puede ver que converge de manera cuadrática, sin embargo en la gráfica de la función $f(x) = x^3 - 2x^2 + \frac{4}{3}x - \frac{8}{27}$ se puede ver la excepción donde el método converge linealmente, esto se debe a que esta función a la multiplicidad de esta función y como se explicó anteriormente

este es uno de los casos donde se pierde la convergencia cuadrática y el algoritmo deja de ser tan eficiente.

Por otro lado, la gráficas con la convergencia mejorada no distan del método de Newton-Raphson tradicional, por lo que que puede ver de manera gráfica que el método de Aitken no genera un eficiencia significativa mayor a lo que ya es el método tradicional.

- **Realice una gráfica que muestre como se comporta el método en cada caso con respecto a la tolerancia y al número de iteraciones**

  Las siguientes gráficas muestran la relación entre las tres tolerancias dadas y el número de iteraciones para cada ejercicio usando los métodos de Newton Raphson y Newton Raphson mejorado con Aitken.

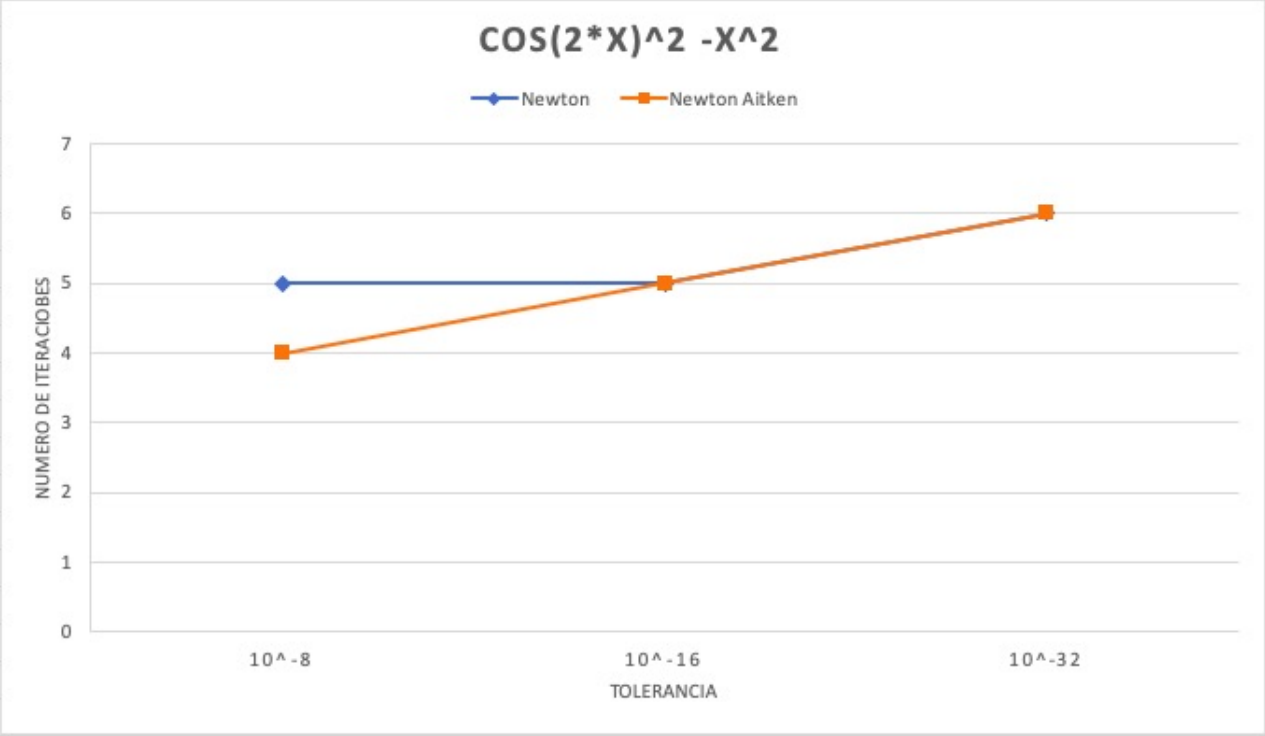

Fig 13. Gráfica problema 1

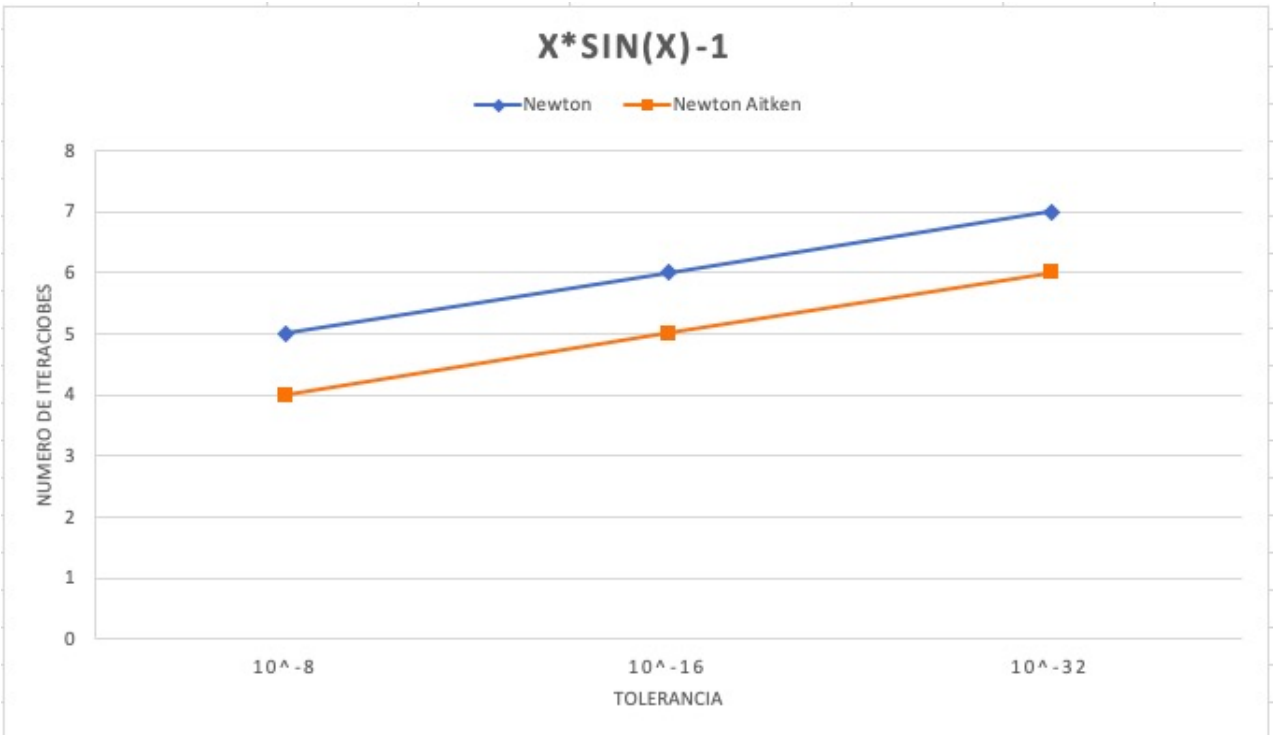

Fig 14. Gráfica problema 2

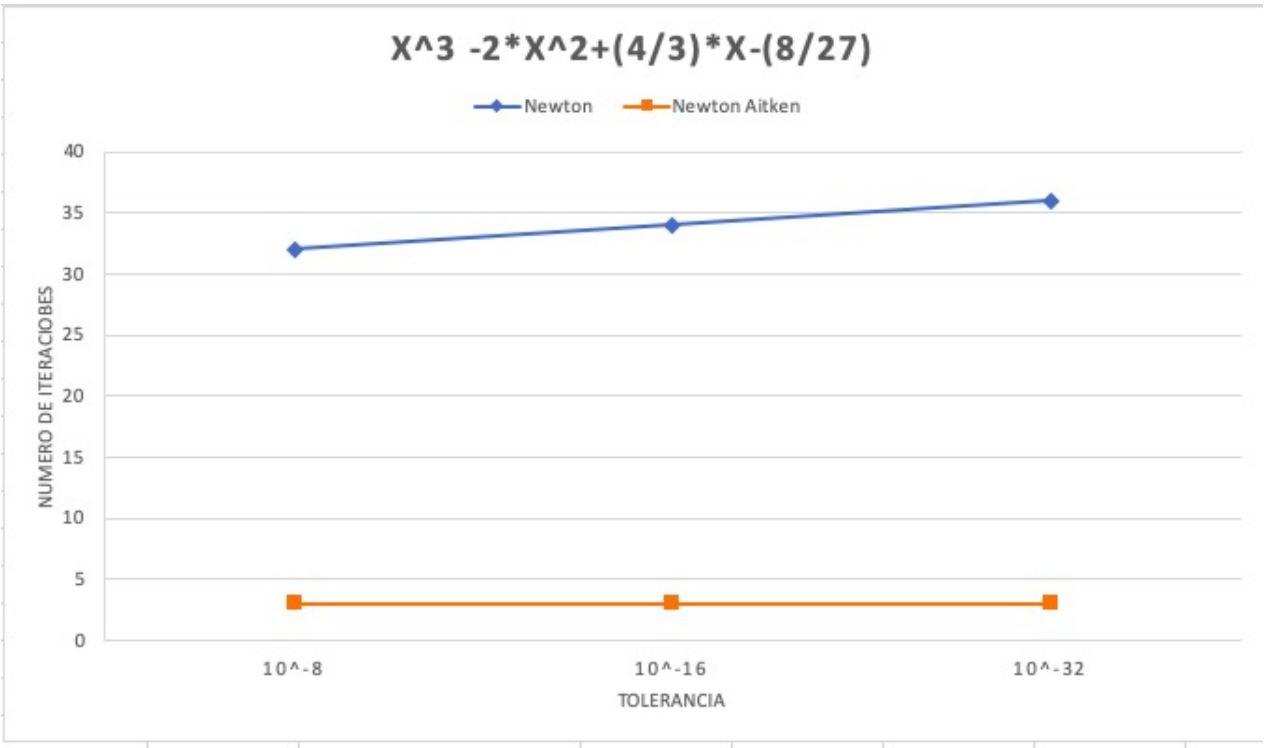

Fig 15. Gráfica problema 3

A continuación se mostrarán las gráficas que relacionan tolerancia con número de iteraciones de todos los ejercicios por cada método.

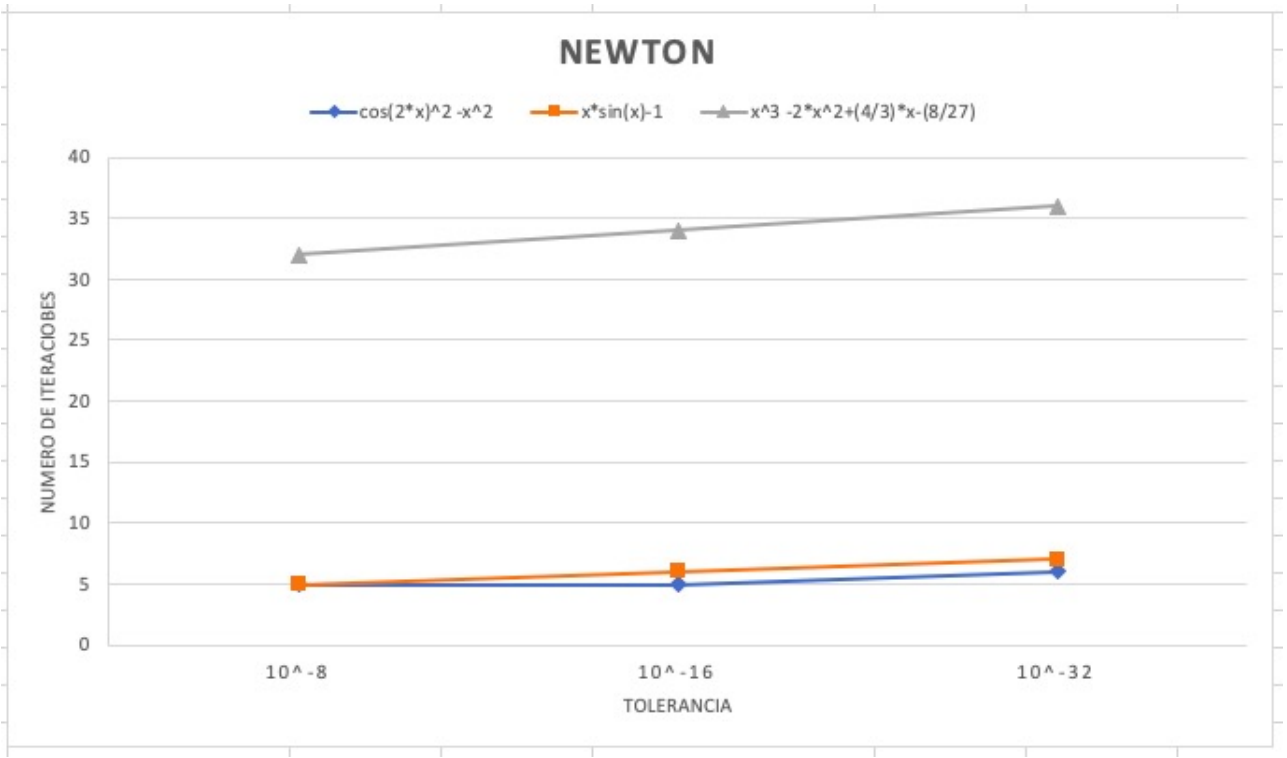

Fig 16. Gráfica Newton

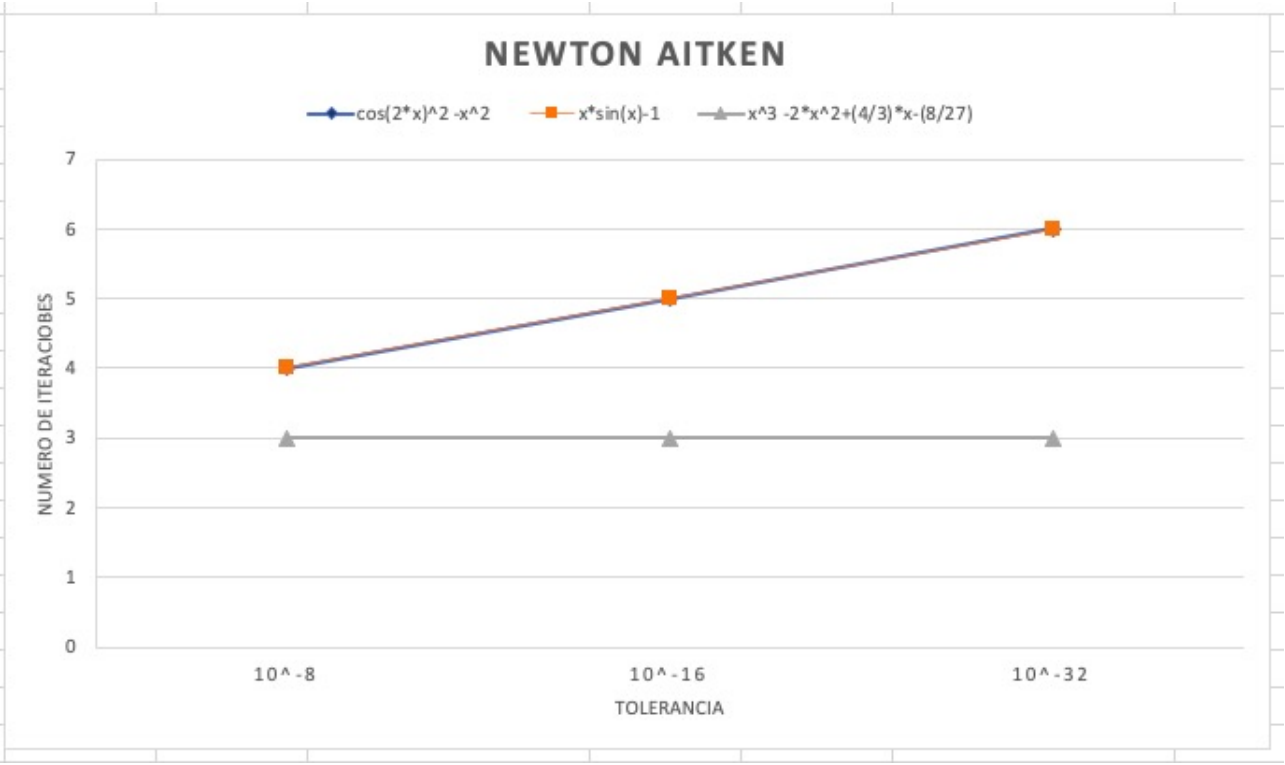

Fig 17. Resultado problema 1

En las gráficas se puede observar que el algoritmo de Aitken es efectivo para mejorar el método de Newton Raphson, debido a que gracias a este algoritmo se han podido disminuir la cantidad de iteraciones necesarias para llegar a la raíz y que el error sea menor que la tolerancia dada.

También se puede observar que Por ejemplo para el primer ejercicio, el algoritmo de Aitken fue útil únicamente para una tolerancia más grande (de $10^{-}8$). Ya que para tolerancias más pequeñas, el número de iteraciones no cambia con un método o con el otro.

En el segundo ejercicio se nota una mejoría entre un método y el mejorado con Aitken. A pesar de que ambos métodos aumentan su número de iteraciones a medida que se disminuye la tolerancia, con este ejercicio se observa una disminución en el número de iteraciones para las tres tolerancias dadas.

Para el tercer ejercicio se observa una diferencia entre el método mejorado y el Newton normal. Mientras que en el método que no está mejorado se ve que a medida que disminuye la tolerancia aumenta el número de iteraciones. Para el método mejorado el número de iteraciones no varía si se disminuye la tolerancia. Además que el número de iteraciones es significativamente más pequeño comparado con el número de iteraciones que debe hacer con el método que no está mejorado

- **Como se comporta el método con respecto al de bisección**

  Con respecto a el método de bisección, encontramos que existe una diferencia grande en la convergencia debido a que Newton-Raphson converge cuadraticamente y por el contrario

bisección presenta una convergencia lineal, lo que hace que sea un poco mas lento al momento de generar un resultado aproximado. Esto puede verse reflejado en las siguientes gráficas donde se muestra la cantidad de iteraciones vs la medición de error

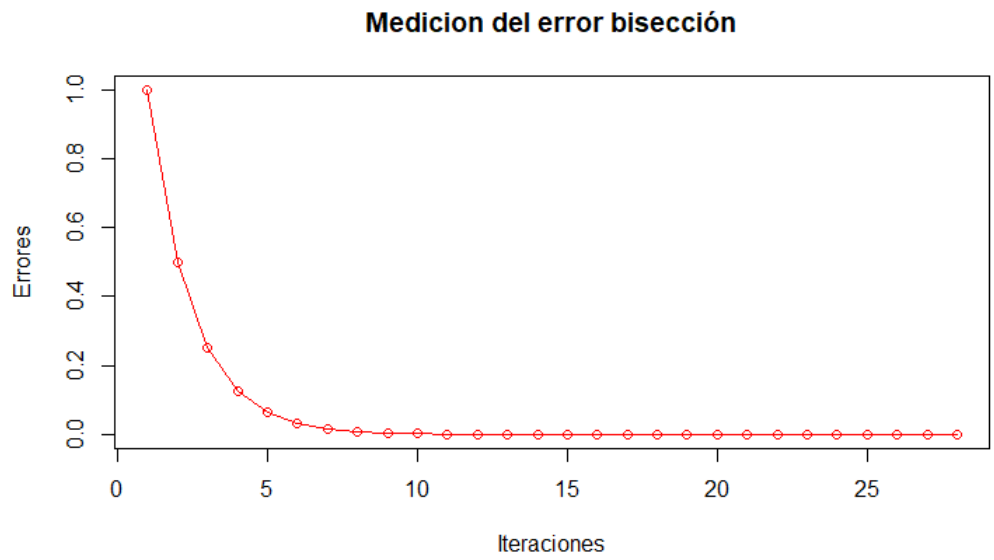

Fig 18. Medición de error Bisección

En este caso (bisección) vemos que se realizaron 27 iteraciones en total bajo una tolerancia mínima de 10^{-8}.

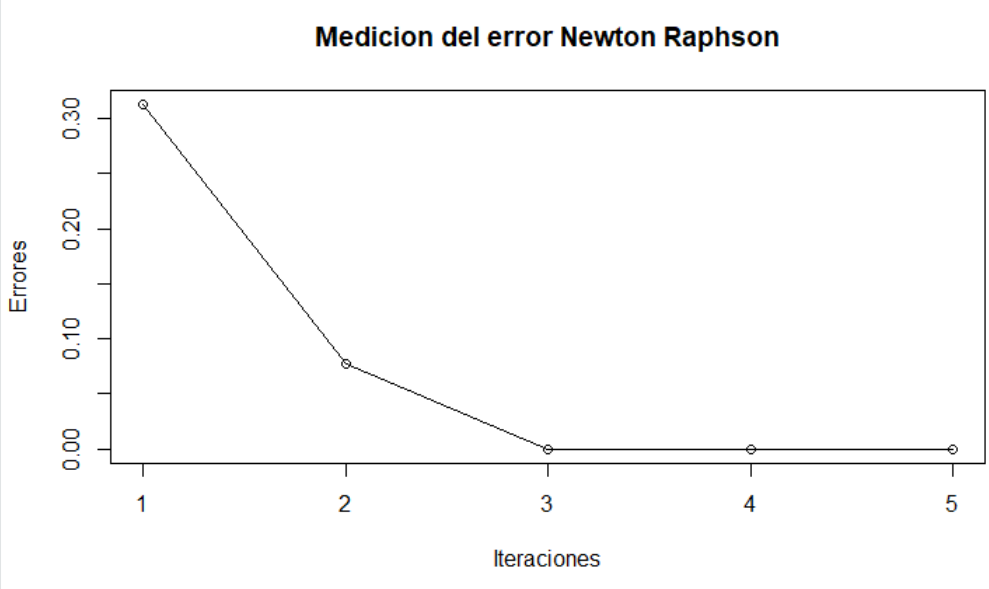

Fig 19. Medición de error NewtonRaphson

En cuento a Newton-Raphson encontramos que solo fueron necesarias 5 iteraciones, reducción

de un poco más del 80 %, trabajando también con una tolerancia mínima de 10^{-8}, entonces se puede decir que Newton-Raphson es un método mas eficiente y mas rápido que bisección.

Este comportamiento confirma de manera practica que entre mayor sea el orden de convergencia, mejor va a ser el algoritmo debido a que necesita de menos iteraciones para llegar a un buen resultado.

2. Bibliografía

- Burden, R. L., & Faires, J. D. (1997). Numerical Analysis, Brooks. Cole, Belmont, CA