Reto 1

Nicolás Pérez,Juan Pablo Amorocho, Diego Fernando Trujillo, Jose Alejandro López

13 de Septiembre 2020

1 Punto 1

En el primer punto se busca evaluar las raices de un polinomio implementando distintos metodos siendo primero el metodo de horner modificado para que este evalue de manera eficiente la primera y la segunda derivada en cualquier punto teniendo en cuenta el redondeo y la presicion

En la segunda parte del punto se busca comparar junto los resultados anteriores y el metodo de Laguerre para obtener el Metodo de Newton Horner aplicando el polinomio con el algorimo creado y comparar los resultados con el algoritmo de Laguerre

El método de Horner logra encontrar la respuesta de P(x0) y sus derivadas a través de utilizar división sintética. Esto se puede usar despues en un algoritmo llamado Newton-Horner para encontrar las raices de un polinomio.

A continuación, se puede ver un ejemplo hecho a mano del polinomio

$$x^4 - 5x^3 - 9x^2 + 155x - 255$$

(Mirar la Figura 1 y 2 en la siguiente página).

Esto fue exactamente lo que se hizo en el código, el cual resulto con estos resultados para el mismo algoritmo: (Mirar la figura 3 en la siguiente página).

Los resultados del algoritmo planteado de Horner soporta raices y coeficientes complejos, y tiene un numero de significancia de digitos en 16, aunque puede soportar distintos gracias al uso de la libreria Rmpfr. Esto tambien es aplicable para el método de Newton-Horner descrito a continuación:

El metodo Utiliza division sintetica para encontrar las raices de la funcion formando distintos niveles de iteraciones dependiendo de el valor de x0 como por ejemplo en las siguientes imagenes para la funcion $x^4 - 9x^2 - 5x^3 + 155x - 250$

Se puede ver como el algoritmo puede tener un numero significativo de iteraciones para una misma funcion teniendo como diferencia el valor inicial de la funcion

2 Punto 2

Algoritmo de brent Es un método que utiliza lo más conveniente de las estrategias de bisección y secante. Se aplica a un intervalo [a, b] en cuyos ex-

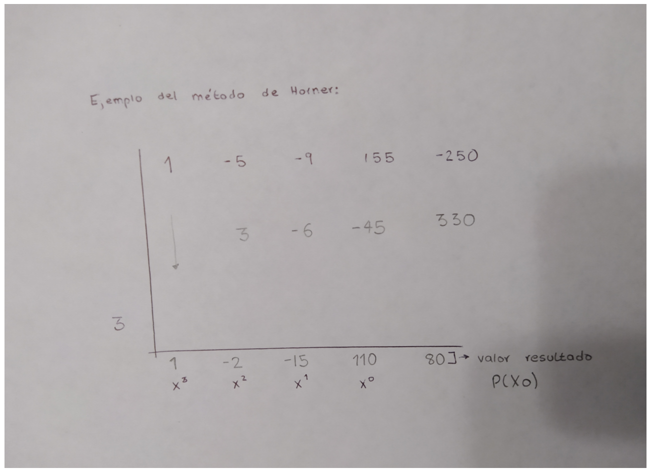

Figure 1: Figura P(X0)

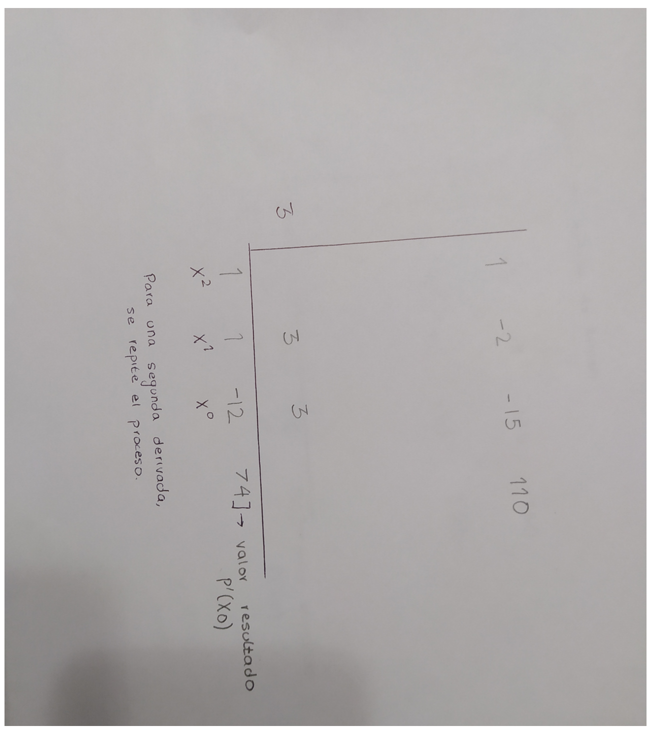

Figure 2: Figura P'(X0)

P(Xo) = 80+0i Q(Xo) = 74+0i (Derivada) R(Xo) = 0+0i (2nda Derivada)

Figure 3: Figura Resultados Horner

```
x_ 1  =  1.857142857142857+0i en la iteracion:  1
x_ 2  =  1.996622948244718+0i en la iteracion:  2
x_ 3  =  1.999998119701312+0i en la iteracion:  3
x_ 4  =  1.999999999999417+0i en la iteracion:  4
x_ 5  =  2+0i en la iteracion:  5
```

Figure 4: Resultados Newton-Horner con x0=2

```
x_ 1  =  -20.05882352941176+0i en la iteracion:  1
x_ 2  =  -14.89343826522795+0i en la iteracion:  2
x_ 3  =  -11.09873513157429+0i en la iteracion:  3
x_ 4  =  -8.379439749604931+0i en la iteracion:  4
x_ 5  =  -6.544065321609981+0i en la iteracion:  5
x_ 6  =  -5.480412548436405+0i en la iteracion:  6
x_ 7  =  -5.06495523090802+0i en la iteracion:  7
x_ 8  =  -5.001405118860213+0i en la iteracion:  8
x_ 9  =  -5.000000676491318+0i en la iteracion:  9
x_ 10  =  -5.00000000000157+0i en la iteracion:  10
x_ 11  =  -5+0i en la iteracion:  11
```

Figure 5: Resultados Newton-Horner con x0=3

```
El resultado es:  0.6666641235351562 Se obtuvo el resultado en la iteracion numero:  17
```

Figure 6: Resultado

```
>> f1=@(x) x^3-2*x^2+x*4/3-8/27;
>> tic, x=Brent(0,1,eps,eps,f1), toc
x =
   0.666670143350873
Elapsed time is 0.000297 seconds.
```

```
>> tic, fzero(f1,[0 1],optimset('Display','iter')), toc
 Func-count    x          f(x)             Procedure
    2               1      0.037037        initial
    3        0.888889     0.0109739        interpolation
    4        0.843776    0.00555553        interpolation
    5        0.798619    0.00229748        interpolation
    6        0.798619    0.00229748        bisection
    7        0.755766    0.00070733        interpolation
    8        0.737384   0.000353658        interpolation
    9         0.71944   0.000146977        interpolation
   10         0.71944   0.000146977        bisection
   11        0.702418   4.56958e-05        interpolation
   12        0.695012   2.27741e-05        interpolation
   13        0.687828   9.47672e-06        interpolation
   14        0.687828   9.47672e-06        bisection
   15        0.681016   2.95457e-06        interpolation
   16         0.67804   1.47117e-06        interpolation
   17        0.675159     6.124e-07        interpolation
   18        0.675159     6.124e-07        bisection
   19        0.672426   1.91084e-07        interpolation
   20        0.671232   9.51209e-08        interpolation
   21        0.670075   3.95999e-08        interpolation
   22        0.670075   3.95999e-08        bisection
   23        0.668979   1.23591e-08        interpolation
   24        0.668499   6.15183e-09        interpolation
   25        0.668035   2.56116e-09        interpolation
   26        0.668035   2.56116e-09        bisection
```

```
   27        0.667595   7.99392e-10        interpolat
   28        0.667402   3.97894e-10        interpolat
   29        0.667216   1.65654e-10        interpolat
   30        0.667216   1.65654e-10        bisection
   31        0.667039   5.17053e-11        interpolat
   32        0.666962    2.5736e-11        interpolat
   33        0.666887   1.07147e-11        interpolat
   34        0.666887   1.07147e-11        bisection
   35        0.666816   3.34444e-12        interpolat
   36        0.666785   1.66456e-12        interpolat
   37        0.666755   6.93112e-13        interpolat
   38        0.666755   6.93112e-13        bisection
   39        0.666727   2.16271e-13        interpolat
   40        0.666714   1.07692e-13        interpolat
   41        0.666702    4.4964e-14        interpolat
   42        0.666702    4.4964e-14        bisection
   43        0.666691   1.39888e-14        interpolat
   44        0.666686   6.99441e-15        interpolat
   45        0.666681   2.88658e-15        interpolat
   46        0.666681   2.88658e-15        bisection
   47        0.666676   9.99201e-16        interpolat
   48        0.666674   3.33067e-16        interpolat
   49        0.666673   3.33067e-16        interpolat
   50        0.666673   3.33067e-16        bisection
   51         0.66667  -1.11022e-16        interpola
   52        0.666671   1.11022e-16        interpola
   53         0.66667             0        bisection
Zero found in the interval [0, 1]
ans =  0.666670143350873
Elapsed time is 0.193985 seconds.
```

Figure 7: Resultado libro

tremos la función adopta signos distintos, si no, la función no tiene una raíz en ese intervalo, también Se utilizó como guía y comparación el libro algoritmos y métodos numéricos de José Luis de Fuente. Se tiene la siguiente función: $x^3 - 2x^2 + x * 4/3 - 8/27$

Donde tiene una raíz en el intervalo [0,1], esto se sabe ya que se comprobó por medio de wolfram en donde se encuentra sus raices, se ejecutó el algoritmo realizado en r varias veces y dando como resultado siempre el mismo:

Pará corroborar se remplazó el resultado en la función, donde su obtuvo que f(0,6666641235351562) = 0 en la iteración número 17. Pero al compararlo con la respuesta del libro es diferente, según el libro da:

Se encuentra que tanto el resultado y las iteraciones son diferentes, ya que según el libro, el resultado es 0.66667 y se logra en la iteración 53. Lo anterior es debido a la pérdida de significancia que hay en RStudio, ya que al hacer cada cálculo de la función se pierde significancia, dando como resultado que f(sb) sea directamente 0, sin importar el valor de la tolerancia, y esto se puede deducir ya que se cambió los valores hasta las décima cifra significativa de sb y seguía dando como resultado 0, también se cambió los valores de tolerancia pero tampoco importaba.

3 Punto 3

En el Punto 3 el objetivo era aproximar mediante el agoritmo de Remez la siguiente función:

$f(x) = e^{sin(x)-cos(x^2)}$

Para aproximarlo se utilizan los polinimos de Chebyshov, que se genran de la siguiente manera:

$T_0(x) = 1$

$T_1(x) = x$

$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$

De igual manera que en el métdo de Taylor, este es un algoritmo iterativo, y dependendiendo el orden del polinoimio que se desean aproximar, se obtienen los valores a_i del polinomio.

Estos polinomios tienen una definición trigonometrica que es la siguiente:

$T_n(x) = cos(narccos(x)) = cosh(narccoh(x))$

References

[1] Richard L. Burden y J Douglas Faires, *Numerical Analysis*, novena edicion, Nueva York, 2010.

[2] Quarteroni, A., Saleri, F y Gervasio, P., *Scientific Computing with MathLab and Octave*, cuarta edicion,Boston MA, 2010.

[3] Tetsure, Yamamoto, *Historical Developments in Convergence Analysis for Newton's and Newton-like Methods"*, cuarta edicion, Nueva York, 2010.