Reto I

Juan Jose Camacho
j_camacho@javeriana.edu.co

Gabriela Maria Camacho
gabriela.m.camacho@javeriana.edu.co

24 de febrero de 2020

1. Evaluacion de un polinomio

Evaluar polinomios o funciones en general tiene muchos problemas, aun para el software profesional. Como se requiere poder evaluar el polinomio en las raices encontradas, es necesario dedicarle un momento a los detalles. Sea $P(x) = a_0x^n + a_1x^{n-1} + ... + a_n$ un polinomio entonces, uno de los problemas que se enfrentan es evaluar el polinomio en valor dado x0 de la manera mas eficiente. Un metodo visto es Horner:

- Implemente en R o Python el metodo de Horner para evaluar $f'(x_0)$, tenga en cuenta la precisión de la computadora o unidad de redondeo.

  El método de Horner, es un método que permite evaluar un polinomio de forma monomio, es decir que lo realiza por medio de multiplicaciones y sumas. Lo primero que se realizo fue la comprobación y cancelación de coeficientes en este se busca seleccionar y eliminar los números son suficientemente pequeños, para convertirlos en cero con el fin de que estos no intervengan con el resultado, esto lo realizamos encontrando los valores mas pequeños, ubicamos su posición y fanalmente convertimos el valor de esta posición en cero.

  Pero el encontrar evaluar Horner en la derivada del polinomio se utilizo la división sintética:

  Seleccionamos el polinomio dado en nuestro caso y como también lo demuestra el libro Análisis Numérico de Richard Buerden en la pagina 93, ejemplo 2 $f(x) = 2x^4 - 3x^2 + 3x - 4$ y le aplicaremos la división sintética para encontrar Horner del polinomio, finalmente esto nos entrega el resultado de 2 -4 5 7 10, cada uno de estos resultados en su orden representa el tipo de coeficiente que este tiene

  Finalmente igualamos a estos coeficientes encontrados por Horner, ignoramos el el ultimo que es la variable independiente porque es la derivad y se vuelve a aplicar la

división sintética con el mismo dividendo es decir -2 de esta manera encontramos el resultado de la derivada por medio de Horner y se obtiene como resultado 2 -8 21 -49

Este procedimiento se podría seguir realizando pero nuestro dividendo tendría que empezar a cambiar en este caso es necesario realizar la siguiente ecuación

$$nuevodividendo = dividendo - \frac{variableindependientedelpolinomio}{valiableindependientedelpolinomioderivado}$$

De esta manera también podemos evidenciar el error absoluto que se presenta que en nuestro caso es de $0{,}02e^{-9}$.También es necesario aclara que como se tiene que realizar mas operaciones este presenta un error mas grande En nuestra caso como se menciono en taller la unidad de redondeo que responde R es la IEEE 754.

Horner obtenido	Horner Esperado	E.Absoluto	E.Relativo
2	2	-0.0000002468	-0.000012341
-4	-4	-0.0000004936	-0.000012341
5	5	-0.0000006170	-0.000012341
7	7	-0.0000008639	-0.000012341
10	10	-0.0000001234	-0.000012341

Cuadro 1: Resultados obtenidos con Horner en un polinomio sin derivar

Horner obtenido	Horner Esperado	E.Absoluto	E.Relativo
2	2	-0.000002468	-0.000012341
-8	-8	-0.0000009873	-0.00012341
21	21	-0.000025916	-0.00012341
-49	-49	-0.000060471	-0.00012341

Cuadro 2: Resultados obtenidos con Horner en un polinomio derivado

- Implemente el método de Horner si se realiza con números complejos, tenga en cuenta la precisión.
  Para el desarrollo de este punto se llevo acabo el mismo procedimiento anteriormente descrito, solo que para poder operar números complejos es necesario especificarlos al programa y su forma de evaluar la división sintética compleja también, el polinomio que se uso en este caso es $polinomiof(x) = (1+i)x3+2$ y dividendo complejo sera $1-i$
  Aplicando el mismo procedimiento obtenemos como resultado $HornerComplejo = 1+1i2+0i2-2i2-3i$. Pero segundo el ejemplo los resultados esperados eran $1+1i2+0i2-2i2-4i$. Igualmente si se deseara encontrar el Horner de la derivada de este polinomio con números complejos ser realizaría exactamente igual que el procedimiento de un polinomio no complejo

Antes de hablar del error encontrado, es necesario especificar que aunque Horner si elimina o reduce las multiplicaciones y las sumas, el simple hecho de que sean números complejos obliga a la realización de mas operación lo cual equivale a un error más grande.

Horner obtenido	Horner Esperado
1+1i	1+1i
2+0i	2+0i
2-2i	2-2i
2-3i	2-4i

Cuadro 3: Resultados obtenidos con Horner en un polinomio complejo

2. Optima Aproximacion Polinomica

La implementación de punto flotante de una función en un intervalo a menudo se reduce a una aproximación polinómica, siendo el polinomio tıpicamente proporcionado por el algoritmo Remez. Sin embargo, la evaluación de coma flotante de un polinomio de Remez a veces conduce a cancelaciones catastróficas.El algoritmo Remez es una metodología para localizar la aproximación racional minimax a una función. La cancelaciones que también hay que considerar en el caso de del método de Horner, suceden cuando algunos de los coeficientes polinómicos son de magnitud muy pequeña con respecto a otros. En este caso, es mejor forzar estos coeficientes a cero, lo que también reduce el recuento de operaciones. Esta técnica, usada clásicamente para funciones pares o impares, puede generalizarse a una clase de funciones mucho mas grande.
Aplicar esta técnica para $f(x) = sin(x)$ en el intervalo $[-\pi/64, \pi/64]$ con una precisión deseada doble. Para cada caso, evalué la aproximación polinómica de la función, el error relativo y el numero de operaciones necesarias.

- Aplique una aproximación de Taylor.

  Para el uso de la aproximación del polinomio de Taylor se hizo uso de la función $taylor(F, x_0, n)$ la cual proviene de la librería Pracma incluida dentro de el script R adjuntado con este documento. Siendo así f: la función diferenciable, x_0: el punto en el cual la expansión tendrá lugar y por ultimo n: el grado del polinomio que se quiere generar.

  Al momento de escoger en cual grado se generaría el polinomio de Taylor se llego a la conclusión de usar un polinomio de grado $n = 3$ ya que los el numero de operaciones a partir de este punto es mayor y el error no disminuía de manera relevante como se muestra en la siguiente tabla:

x_i	$\sin(x_i)$	$P_3(x_i)$	$P_7(x_i)$	E.Absoluto P_3	E.Absoluto P_7
-0.04908739	-0.04906767	-0.04906767	-0.04906767	2.372496e-09	2.377008e-12
-0.03926991	-0.03925982	-0.03925981	-0.03925982	7.770807e-10	1.13367e-12
-0.02945243	-0.02944817	-0.02944817	-0.02944817	1.842764e-10	4.005164e-13
-0.01963495	-0.01963369	-0.01963369	-0.01963369	2.426765e-11	5.221865e-14
-0.009817477	-0.009817319	-0.009817319	-0.009817319	7.984568e-13	3.845535e-14
0	0	0	0	0	0
0.009817477	0.009817319	0.009817319	0.009817319	7.984568e-13	3.845535e-14
0.01963495	0.01963369	0.01963369	0.01963369	2.426765e-11	5.221865e-14
0.02945243	0.02944817	0.02944817	0.02944817	1.842764e-10	4.005164e-13
0.03926991	0.03925982	0.03925981	0.03925982	7.770807e-10	1.13367e-12

Cuadro 4: Comparacion de polinomios basado en el error

Siendo así, se basó la decision de tomar el polinomio de grado 3 puesto que el intervalo en el cual sera evaluada la función es muy pequeño y cercano al origen, por lo que también teniendo en cuenta que θ es muy pequeño al momento de evaluar $sin(\theta) \approx \theta$. Por esto, un polinomio de grado muy alto no se observaba necesario para tener una buena aproximación, por lo que se decidió finalmente tomar el siguiente Polinomio de Taylor:

$$P_3(x) = -\frac{1}{6}x^3 + x \tag{1}$$

- Implemente el método de Remez

  Para implementar el algoritmo de remez se hizo una intensiva búsqueda de la manera de como implementar el algoritmo, así que los siguieron los siguientes pasos :

  1. Obtener números de Chebyshev.

     Se obtuvieron los nodos de Chebyshev ya que estos números no son susceptibles al fenómeno de Runge que es un una larga oscilación que ocurren al final de los intervalos. Por lo que en primer lugar se deben obtener estos nodos para así enviarlos evaluarlos en la función $sin(x)$. La formula para calcular los números de ChebyShev es:

     $$x_i = \frac{1}{2}\Big(a+b\Big) + \frac{1}{2}\,(b-a)\cdot\cos\left(\frac{2i-1}{2n}\pi\right) \tag{2}$$

     Siendo a,b el intervalo en el cual se requieren calcular los nodos, i la iteración actual del nodo, y n el numero total de nodos a calcular.

  2. Nodos para calcular en el algoritmo de Remez

     Para hacer uso adecuado del algoritmo de remez, además de que se deben calcular los nodos de Chevyshev, se deben tomar $n+2$ tomando $n=3$ debido a que daba un error suficientemente bajo, además de que si se tomaba un numero mas alto el algoritmo se volvería ineficiente, los nodos generados son: $\{-4{,}668488e-02, -2{,}885284e-02, -3{,}005636e-18, 2{,}885284e-02, 4{,}668488e-02\}$.

  3. Ingresar nodos de ChebyShev dentro de un sistema de ecuaciones lineales especial:

     Para hallar los coeficientes del polinomio interpolado, el algoritmo de Remez sugiere un sistema de ecuaciones lineales utilizando los nodos de Chebyshev y los valores calculados de la función en esos puntos. Así, se genera el siguiente sistema de ecuaciones donde E es un error y b_i son los coeficientes del polinomio minimax interpolado

     $$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & E \\ 1 & x_1 & x_1^2 & \cdots & -E \\ \vdots & & & & \\ 1 & x_{n+1} & x_{n+1}^2 & \cdots & (-1)^i E \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \\ E \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ \vdots \\ f(x_n+1) \end{pmatrix}$$

Notese que esta matriz es de tamaño (n+2)(n+2) siendo asi para usarse con el comando solve() ya definido en R. Esta matriz cuenta con un E que corresponde al error permitido, en este caso se hizo uso de un error de 10^{-5}

4. Resolver el sistema de ecuaciones para asi obtener el Minimax:

   Al momento de resolver el sistema de ecuaciones se obtiene el siguiente polinomio:

   $$P(x) = x - 1{,}666416 \cdot 10^{-1}x^3 \tag{3}$$

   Siendo este el Polinomio esperado y que se tomara para comparar con la función $\sin(x)$.

5. **Notas Aclaratorias:** Al momento de tomar este polinomio y compararlo con la función $sin(x)$ se denotó que el error absoluto era diminuto, oscilando entre 0 y $1{,}5 \cdot 10^{-10}$. A su vez, el error relativo también es muy bajo, oscilando entre $0\,\%$ y $1{,}5 \cdot 10^{-8}\,\%$ como se puede observar en las siguientes gráficas:

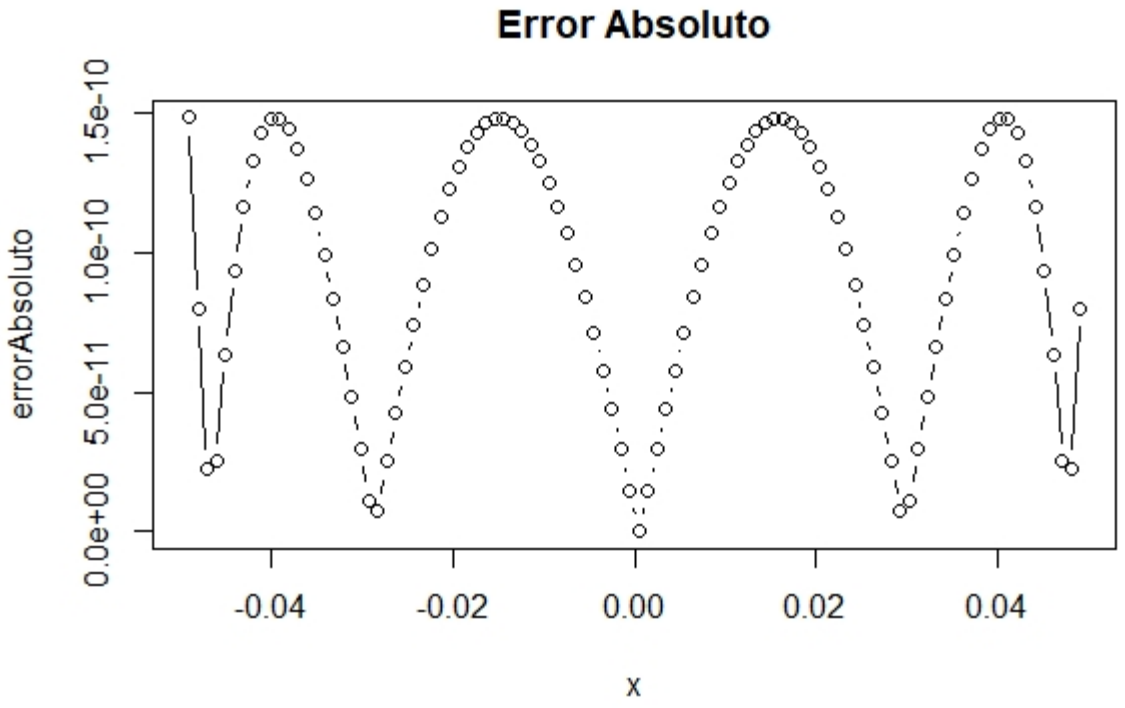

Figura 1: Error absoluto del polinomio

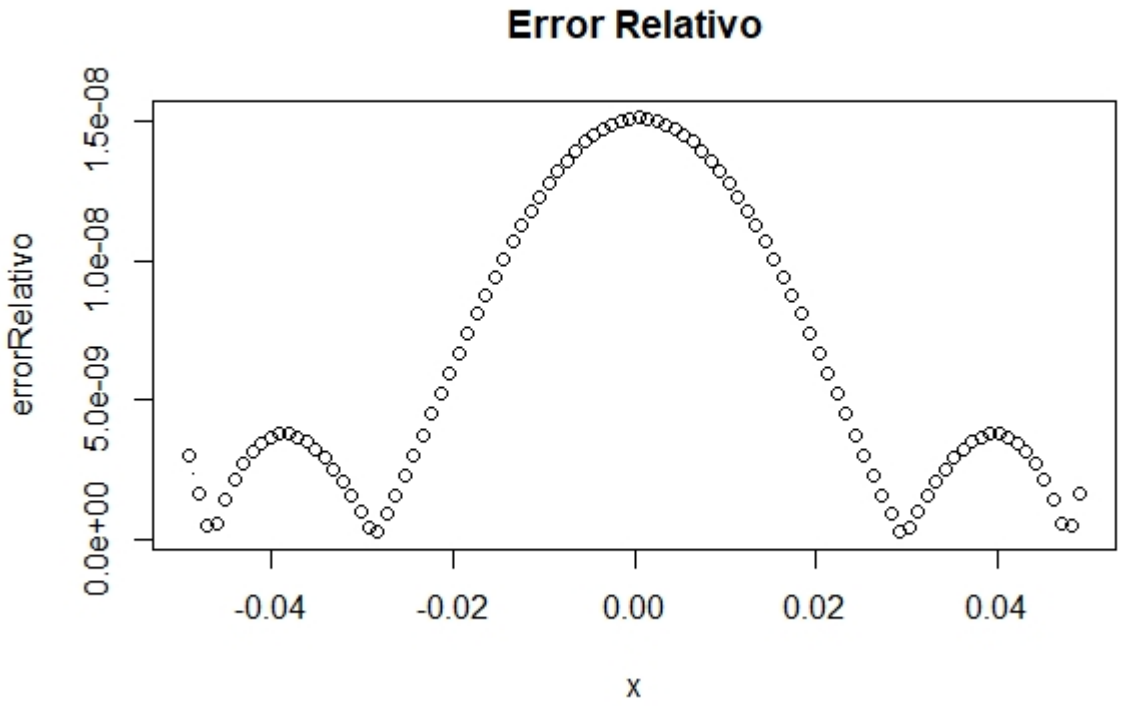

Figura 2: Error relativo del polinomio

6. Comparación del polinomio con la función $\sin(x)$:
   Para concluir este algoritmo, las siguientes gráficas muestran el resultado obtenido por el polinomio en el intervalo requerido:

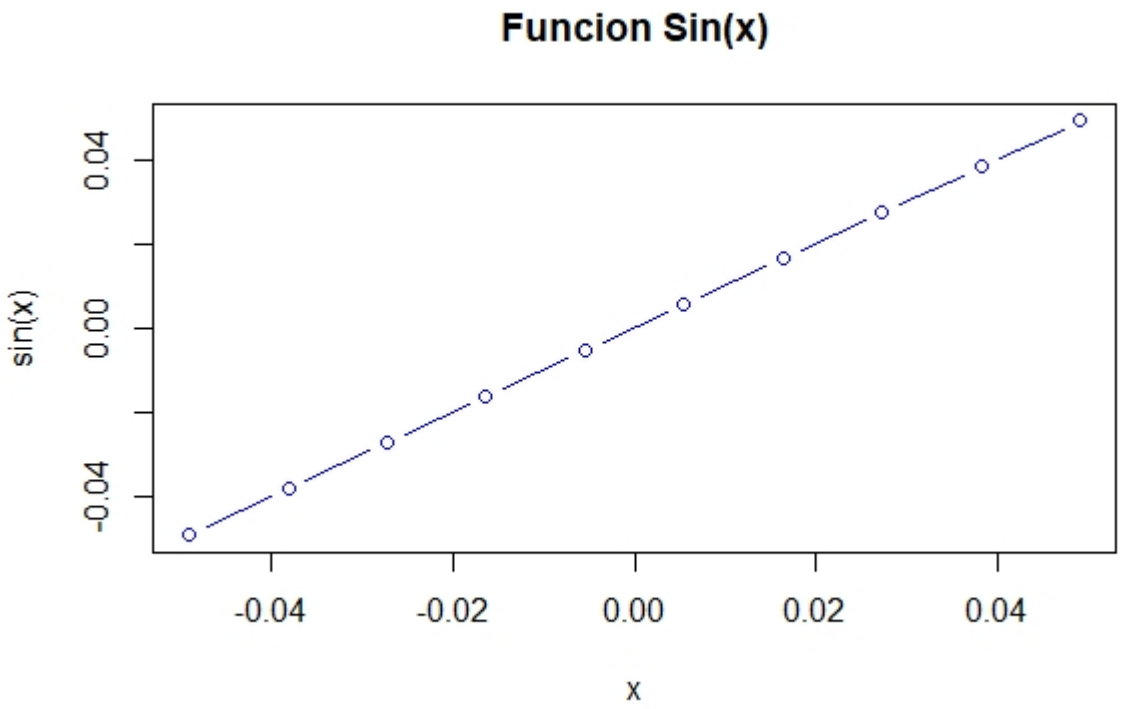

Figura 3: Función Seno

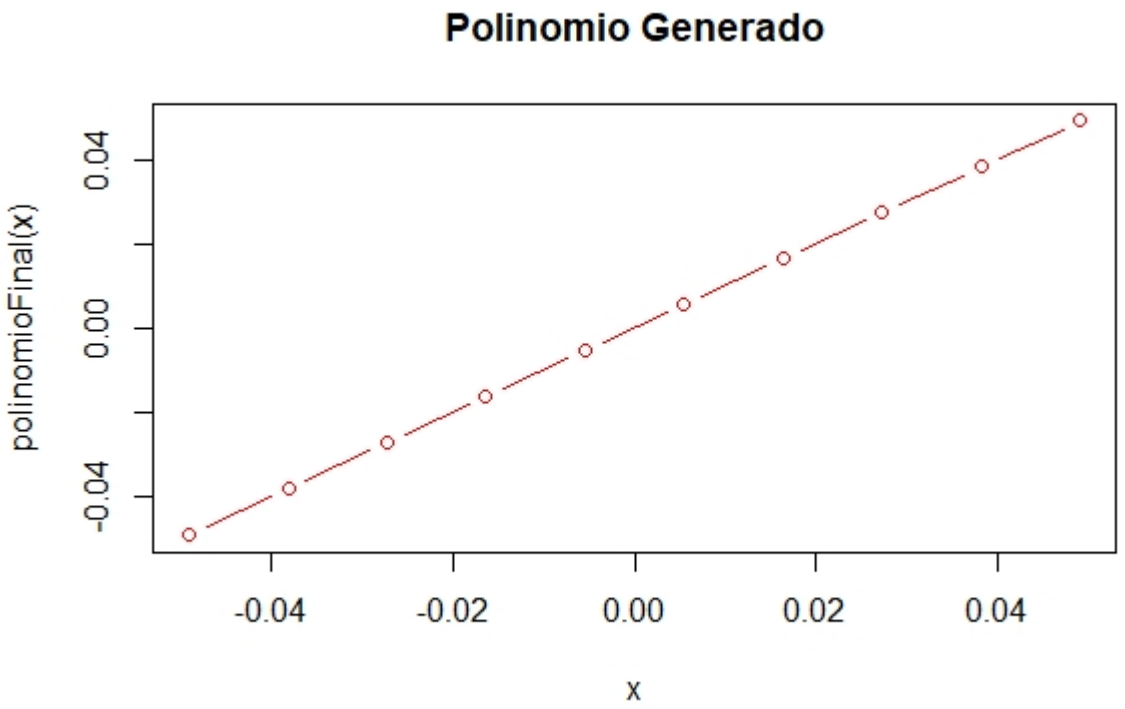

Figura 4: Polinomio Generado

Se puede observar que para el intervalo propuesto, el polinomio generado es casi igual a la funcion seno que pasa por el mismo intervalo.