Pontificia Universidad
JAVERIANA
Bogotá

Pontificia Universidad Javeriana

Departamento de Matemática
Análisis Numérico

Bono
Primer Corte

por

Monica Alejandra Alvarez Carrillo
monica_alvarez@javeriana.edu.co

Santiago Palacios Loaiza
palacios-santiago@javeriana.edu.co

Profesora: Eddy Herrera Daza

Bogotá D.C., 23 de Febrero del 2020

1. Método de Horner para f'(x)

1.1. Teoría

El teorema de Horner, también conocido como método de multiplicación anidada, es un teorema usado en el análisis numérico creado por el matemático ingles William George Horner en el siglo XVII que consiste en calcular el valor de un polinomio de grado n utilizando el menor numero de multiplicaciones. Esta característica de realizar el menor numero de operaciones posibles es quizás la razón por la que es ampliamente usado, debido a que aumenta en gran medida su eficiencia.

El teorema enuncia que un polinomio de la forma $P(x) = a_0x^n + a_1x^{n-1} + ... + a_n$ puede evaluarse en un número n de multiplicaciones y n adiciones

El teorema de Horner también puede ser usado para calcular el valor de la derivada en un punto dado, mientras que, al mismo tiempo se calcula el valor del polinomio con el menor numero de operaciones sobre el mismo punto. A continuación se enseña la demostración:

1.2. Solución

La implementación en R para el teorema de Horner se muestra a continuación: it

```
# coeficientes corresponde a un vector con los coeficientes del polinomio
#x0 es el punto donde evaluamos el polinomio
#valor corresponde al valor final del polinomio en el punto
valor = coeficientes[1] # Ajuste de las variables
  multi =0
  sumas=0
  for(i in coeficientes[2:length(coeficientes)]){
    valor <- x0*valor + i
  }
```

Lo que hace este algoritmo es tomar el primer valor de los coeficientes y guardarlo como resultado, luego multiplicarlo por el punto con el cual evaluamos el polinomio, $x0$, y sumarlo con el siguiente valor de los coeficientes. Este serie de pasos se repite la cantidad de términos que tenga el polinomio. A continuación una manera mas gráfica y clara de entender el algoritmo:

Sea $P(x) = 4x^3 + 2x^2 + x + 7$ un polinomio que evaluado en un punto $x_0 = 2$ corresponde a $P(x_0) = 49$, vamos a escribir los coeficientes ordenados y el valor de x a evaluar:

	4	2	1	7
x=2				

Tabla inicial

	4	2	1	7
x=2				
	4			

Sumamos el valor del primer coeficiente

	4	2	1	7
x=2		**8**		
	4			

Multiplicamos el 4 con el valor de x

	4	2	1	7
x=2		8		
	4	**10**		

Sumamos el 8 con el coeficiente correspondiente, es decir 2

	4	2	1	7
x=2		8	**20**	
	4	10		

Volvemos a multiplicar el anterior resultado con el valor de x; 10 multiplicado por 2

	4	2	1	7
x=2		8	20	
	4	10	**21**	

Volvemos a sumar el anterior resultado con su coeficiente

	4	2	1	7
x=2		8	20	42
	4	10	21	**49**

Continuamos con lo mostrado anteriormente para obtener el resultado final

Visto de manera gráfica vemos que Horner utiliza n multiplicaciones y n sumas para evaluar el polinomio de n términos.Esta manera gráfica de entender el algoritmo nos permite acercarnos mas a la solución para calcular tanto $P(x)$, como $P'(x)$, para un mismo x_0. Para esto necesitamos ver el proceso anterior de una manera diferente, en este caso lo veremos como una división sintética donde $P(x)$ es dividida por $(x - x_0)$ de la forma:

$$\frac{4x^3+2x^2+x+7}{(x-2)} = 4x^2 + 10x + 21 \text{ y nos queda un residuo de } 49$$

Donde el resultado de la división corresponde a la fila inferior de la última tabla del algoritmo propuesto para Horner anteriormente, y el residuo corresponde a el valor de $P(x_0)$. Por facilidad y con la intención de generalizar la solución vamos a asignar nombres a diferentes partes de la operación anterior, $Q(x)$ corresponderá a una función en términos de x resultado de la división; $4x^2 + 10x + 21$ y C corresponderá a una constante residuo; 49. Ahora reescribimos la operación:

$$\frac{P(x)}{(x-x_0)} = Q(x) + C$$

De aquí podemos despejar:

$$P(x) = (x - x_0) * Q(x) + C$$

Teniendo esta formula es mas sencillo aproximarnos al valor de la derivada que queremos calcular, vamos a comenzar derivando a ambos lados y simplificando:

$$P'(x) = ((x - x_0) * Q(x) + C)'$$

$$P'(x) = (x - x_0)' * Q(x) + (x - x_0) * Q'(x) + C'$$

$$P'(x) = 1 * Q(x) + (x - x_0) * Q'(x) + 0$$

$$P'(x) = Q(x) + (x - x_0) * Q'(x)$$

Hemos encontrado una aproximación al valor de $P'(x)$, sin embargo el valor de $Q'(x)$ puede ser igual de difícil de hallar actualmente por lo que es pertinente eliminar este termino. Recordemos que el objetivo principal es calcular $P(x)$ y $P'(x)$ para un mismo x por lo tanto podemos asegurar que $x = x_0$ para luego reemplazar en la formula:

$$P'(x_0) = Q(x_0) + (x_0 - x_0) * Q'(x_0)$$

$$P'(x_0) = Q(x_0) + (0) * Q'(x_0)$$

$$P'(x_0) = Q(x_0) + 0$$

$$\mathbf{P'(x_0) = Q(x_0)}$$

Finalmente encontramos un valor para $P'(x)$ que podemos calcular, lo que sugiere la formula es que para encontrar el valor de la derivada en el punto x_0 debemos evaluar el polinomio $Q(x)$ en el mismo punto, evidentemente procedemos a evaluarlo con el mismo método de Horner. Se muestra a continuación:

	4	2	1	7
x=2		8	20	42
	4	10	21	**49**

Tabla final de Horner $P(x)$, sobre la misma vamos a trabajar $Q(x)$

	4	2	1	7
x=2		8	20	42
	4	10	21	49
x=2				

Tabla inicial para $P'(x)$

	4	2	1	7
x=2		8	20	42
	4	10	21	49
x=2				
	4			

Sumamos el valor del primer coeficiente

	4	2	1	7
x=2		8	20	42
	4	10	21	49
x=2		**8**		
	4			

Multiplicamos el 4 con el valor de x

	4	2	1	7
x=2		8	20	42
	4	10	21	49
x=2		8		
	4	**18**		

Sumamos el 8 con el coeficiente correspondiente de $Q(x)$, es decir 10

	4	2	1	7
x=2		8	20	42
	4	10	21	49
x=2		8	36	
	4	18	**57**	

Continuamos con lo mostrado anteriormente para obtener el resultado final

Finalmente obtuvimos $Q(x_0) = 57$ para comprobar este resultado vamos a calcular la derivada del polinomio en el punto por métodos tradicionales:

$$P(x) = 4x^3 + 2x^2 + x + 7$$

$$P'(x) = 12x^2 + 4x + 1 + 0; x_0 = 2$$

$$P'(x_0) = 57; x_0 = 2$$

De esta manera comprobamos que es posible para todos los casos usar el teorema de Horner para calcular el valor de un polinomio y su derivada en el mismo punto. En el ejemplo calculamos una función $P(x) = 4x^3 + 2x^2 + x + 7$ con un $x_0 = 2$ para obtener $P(x_0) = 49; P'(x_0) = 57$. Para la implementación en código de R de este simplemente es necesario tener un par de consideraciones que se enfocan principalmente, en usar un coeficiente menos para hallar la derivada, que corresponde al residuo C y que el valor de la derivada es dependiente del valor del polinomio, el código y un par de casos de prueba se muestran a continuación:

```
# coeficientes corresponde a un vector con los coeficientes del polinomio
#x0 es el punto donde evaluamos el polinomio
#valor corresponde al valor final del polinomio en el punto
#Der almacena el valor de la derivada
  valor = coeficientes[length(coeficientes)] # Ajuste de las variables
  Der = 0
  multi = 0
  sumas = 0
  j = length(coeficientes) - 1
```

```
  while (j > 0) {
    i = coeficientes[j]
    Der = valor + (x0 * Der)
    valor <- x0 * valor + i
    j = j - 1
  }
```

1.3. Pruebas

* Con el polinomio $P(x) = 100x^4 + 70x^3 + 2x + 1$, cuya derivada es $P'(x) = 400x^3 + 210x^2 + 2$ y con un $x_0 = 1$) tenemos los valores en la siguiente tabla:

	Horner	Metodo Clasico
$P(x)$	171	171
$P'(x)$	612	612

Comparación de resultados

* Con el polinomio $P(x) = 12x^5 + 10x^4 + 8x^3 + 6x^2 + 4x + 2$, cuya derivada es $P'(x) = 60x^4 + 40x^3 + 24x^2 + 12x + 4$ y con un $x_0 = 1$) tenemos los valores en la siguiente tabla:

	Horner	Metodo Clasico
$P(x)$	4010	4010
$P'(x)$	6196	6196

Comparación de resultados

* Con el polinomio $P(x) = 0{,}0012x^2 + 0{,}5x + 1$, cuya derivada es $P'(x) = 0{,}0024x + 0{,}5$ y con un $x_0 = 10$) tenemos los valores en la siguiente tabla:

	Horner	Metodo Clasico
$P(x)$	6.120	6.120
$P'(x)$	0.524	0.524

Comparación de resultados

1.4. Cancelaciones

Para el caso del Método Horner, se debe tener en cuenta que pueden existir coeficientes demasiado pequeños en proporción respecto a otros. Una manera de solucionar esto y además hacer más efectivo el método se propone forzar dichos coeficientes a cero y de esta manera reducir el número de operaciones.

1.4.1. Pruebas

Se evalúa el polinomio $10x^5 + 25x^4 + 0{,}001x^3 + 0{,}0012^2 + 6x + 3$ con el método Horner y con las modificaciones al método Horner para tener en cuenta las cancelaciones, se varía la cota de cancelación en cada prueba y se obtienen los siguientes resultados: Se definió $x_0 = 10$.

Cota	Horner	Con Cancelaciones	Er. Absoluto	Er. Relativo
40 %	360261.3	360260	1.30	0.36x10^{-4} %
60 %	360261.3	360260	1.30	0.36x10^{-4} %
70 %	360261.3	360261.3	0.00	0.00 %
80 %	360261.3	360261.3	0.00	0.00 %
90 %	360261.3	360261.3	0.00	0.00 %
95 %	360261.3	360261.3	0.00	0.00 %
99 %	360261.3	360261.3	0.00	0.00 %

Comparación de resultados

Se evalúa el polinomio $15973x^4 + 7x^3 + 352719^2 + 12000x + 3$, se definió $x_0 = 134$.

Cota	Horner	Con Cancelaciones	Er. Absoluto	Er. Relativo
15 %	36173941083	35206686337	9.67x10^8	2.75 %
35 %	36173941083	35206686337	9.67x10^8	2.75 %
55 %	36173941083	35206686337	9.67x10^8	2.75 %
65 %	36173941083	36173941083	0.00	0.00 %
85 %	36173941083	36173941083	0.00	0.00 %
95 %	36173941083	36173941083	0.00	0.00 %

Comparación de resultados

2. Método de Horner para Números Complejos

2.1. Teoría

Los números complejos

$\mathbb{C}$, son la extensión del conjunto de los números reales $\mathbb{R}$que se pueden representar como la suma de un número real y un número imaginario múltiplo de la unidad imaginaria i. Los números complejos son un concepto matemático importante en el cálculo de raíces de un polinomio.

La unidad imaginaria i es la solución a la ecuación de grado 2, $x^2 + 1 = 0$, que se denota de esta manera por la no existencia de un número real con un cuadrado negativo, de esta manera, los números complejos garantizan que al menos exista una raíz para cada polinomio no constante $P(x)$.

2.2. Solución

Para la solución de polinomios que contienen números complejos la implementación del algoritmo de Horner mostrada previamente se mantiene igual puesto que, a términos generales, el procedimiento es el mismo ya que los números complejos, por lo menos a primera vista, no requieren ninguna actuación especial para operaciones simples como la adición y la multiplicación sobre las que trabaja Horner.

Sin embargo es importante recordar que los números complejos se suelen escribir de la siguiente forma binomial:

$$Z = a + bi$$

Por lo tanto la multiplicación de un numero complejo por un escalar tiene la forma:

$$C * Z$$

$$C * (a + bi)$$

$$C * a + C * bi$$

Podemos ver como la naturaleza binomial de los numero complejos aumenta la cantidad de operaciones básicas necesarias; en este ultimo caso una multiplicación de números reales solo necesita de un operador de multiplicación para resolverse, en cambio, cuando se multiplica por un numero complejo se duplican la cantidad de multiplicaciones debido a los dos términos del polinomio que representa el numero complejo. Esta operaciones extras se aumentan aun mas cuando operamos dos números complejos:

$$Z_1 = a + bi \; Z_2 = c + di$$

$$Z_1 * Z_2$$

$$(a + bi) * (c + di)$$

$$a * c + a * di + c * bi + bi * di$$

$$a * c + a * di + c * bi + bdi$$

Es sencillo apreciar que la cantidad de operaciones básicas de multiplicación se duplica otra vez, para un total de cuatro multiplicaciones necesarias por cada producto entre números complejos. Una de las principales características del Teorema de Horner es su capacidad de evaluar el polinomio con la menor cantidad de operaciones básicas posibles, anteriormente determinas que puede lograrlo con n multiplicaciones y n sumas, siendo n el numero de términos del polinomio, sin embargo esta propiedad deja de ser verídica cuando Horner trabaja con números complejos

La solución presentada en código de R anteriormente realiza básicamente una operación de la forma: $Valor = (x_0 * valor) + j$ por cada iteración, donde $Valor$ corresponde al valor acumulado de la evaluación del polinomio, x_0 corresponde al valor de x que estamos reemplazando en el polinomio, por ultimo, j representa el coeficiente del polinomio de la iteración actual. A partir de esta información podemos plantear una serie casos:

* Sea x_0 o $Valor$ un número complejo tendremos la siguiente expresión:

$$((a + bi) * x_0) + j$$

$$(x_0 * a + x_0 * bi) + j$$

$$A = x_0 * a \quad C = x_0 * b$$

$$(A + Ci) + j$$

$$B = A + j$$

$$B + Ci$$

$$Productos : 2 \quad Sumas : 1$$

* Sean x_0 y $Valor$ números complejos tendremos la siguiente expresión:

$$((a + bi) * (c + di)) + j$$

$$(a * c + a * di + c * bi + bi * di) + j$$

$$A = a * c \quad B = a * d \quad C = c * b \quad D = a * d$$

$$(A + Bi + Ci + Di) + j$$

$$E = B + C + D$$

$$(A + Ei) + j$$

$$F = A + j$$

$$F + Ci$$

$$Productos : 4 \quad Sumas : 3$$

* Sean x_0, $Valor$ yj un numero complejo tendremos la siguiente expresión:

$$((a + bi) * (c + di)) + (e + fi)$$

$$(a * c + a * di + c * bi + bi * di) + (e + fi)$$

$$A = a * c \quad B = a * d \quad C = c * b \quad D = a * d$$

$$(A + Bi + Ci + Di) + (e + fi)$$

$$E = B + C + D$$

$$(A + Ei) + (e + fi)$$

$$F = A + e \;\; G = E + f$$

$$F + Gi$$

$$Productos : 4 \quad Sumas : 4$$

* Sea j un numero complejo tendremos la siguiente expresión:

$$(Valor * x_0) + (a + bi)$$

$$A = Valor * x_0$$

$$A + (a + bi)$$

$$B = A + a$$

$$B + bi$$

$$Productos : 1 \quad Sumas : 1$$

* El caso de que ningún numero sea complejo es el base y corresponde a $Productos : 1 \quad Sumas : 1$

2.3. Pruebas

* A continuación se muestran en una tabla con polinomios, sus soluciones con Horner y la cantidad de operaciones necesarias:

	x_0	Resultados	Multiplicaciones	Sumas
$2x^2 + x + 3$	2	13	2	2
$2x^2 + (1 + 1i)x + 3$	2	12+2i	3	2
$2x^2 + (1 + 1i)x + 3$	2+2i	3+20i	6	5
$2x^5 + (2 + 3i)x^4 + (1 + 1i)x^3 + 3x^2 + 12$	2+3i	308-1792i	18	17

Comparación de resultados

3. Teorema de Taylor

3.1. Teoría

También conocido como método de multiplicación anidada, es un teorema usado en el análisis numérico creado por el matemático ingles William George Horner en el siglo XVII que consiste en calcular el valor de un polinomio de grado n utilizando el menor numero de multiplicaciones. Esta característica de realizar el menor numero de operaciones posibles es quizás la razón por la que es ampliamente usado, debido a que aumenta en gran medida su eficiencia.

El teorema enuncia que un polinomio de la forma $P(x) = a_0x^n + a_1x^{n-1} + ... + a_n$ puede evaluarse en un número n de multiplicaciones y n adiciones.

$$P_n(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots$$
$$+ \frac{f^{(\mu)}(a)}{k!}(x-a)^k + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n$$

Aproximación polinómica de Taylor

3.2. Solución

Dada la fórmula de la imagen anterior, sea $f(x) = sin(x)$ y el centro del intervalo $a = 0$, se tiene que

$$sin(0) + \frac{cos(0)}{2!} * x + \frac{-sin(0)}{2!} * x^2 + \frac{-cos(0)}{3!} * x^3 + \frac{sin(0)}{4!} * x^4 + \frac{cos(0)}{5!} * x^5 + \frac{-sin(0)}{6!} * x^6$$

De lo anterior podemos observar que todos los términos pares contienen la expresión sin(0) la cual siempre tiene un valor igual a 0, por lo tanto podemos eliminar de nuestro Polinomio de Taylor los términos pares, además es fácil notar que los términos impares que restan tendrán siempre una característica intercalada de positivos y negativos debido a la naturaleza cíclica de la derivada de la función. Entonces el polinomio de Taylor queda de la siguiente manera:

$$P(x) = x + \frac{-x^3}{3!} + \frac{x^5}{5!} + \frac{-x^7}{7!} + \frac{x^9}{9!}$$

A continuación se prueban los tres primeros términos del polinomio y sus respectivos errores:

x	$x + \frac{x^3}{3!}$	$x + \frac{x^3}{3!} + \frac{x^5}{5!}$	sin(x)
-0.049087385	-0.049107098	-0.049107101	-0.049067674
-0.029452431	-0.029456689	-0.029456689	-0.029448173
-0.009817477	-0.009817635	-0.009817635	-0.009817319
0.009817477	0.009817635	0.009817635	0.009817319
0.029452431	0.029456689	0.029456689	0.029448173
0.049087385	0.049107098	0.049107101	0.049067674

Tabla con pruebas

x	$x + \frac{x^3}{3!}$	$x + \frac{x^3}{3!} + \frac{x^5}{5!}$
0.000019711 ‖ 0,040154685 %	0.000000002 ‖ 0,000004840 %	0.000000000 ‖ 0,000000000 %
0.000004258 ‖ 0,014456801 %	0.000000000 ‖ 0,000000627 %	0.000000000 ‖ 0,000000000 %
0.000000158 ‖ 0,001606373 %	0.000000000 ‖ 0,000000008 %	0.000000000 ‖ 0,000000000 %
0.000000158 ‖ 0,001606373 %	0.000000000 ‖ 0,000000008 %	0.000000000 ‖ 0,000000000 %
0.000004258 ‖ 0,014456801 %	0.000000000 ‖ 0,000000627 %	0.000000000 ‖ 0,000000000 %
0.000019711 ‖ 0,040154685 %	0.000000002 ‖ 0,000004840 %	0.000000000 ‖ 0,000000000 %

Tabla con errores; Error absoluto ‖ Error relativo

De la tabla anterior podemos ver claramente como un polinomio de Taylor resulta ser mas preciso cuando tiene mas términos. Sin embargo esta precisión esta dada por una cantidad de operaciones extra que pueden representar mucho tiempo de computo en algunas situaciones, en nuestro caso desde el segundo termino se alcanza una precisión satisfactoria para la solución del problema, a continuación se probara mas a fondo este polinomio. Para estas pruebas se usa el término $x + \frac{-x^3}{3!}$ por razones de precisión y simplicidad que ayudan a mejorar el cómputo.

$sin(x)$	$x + \frac{x^3}{3!}$	Er. Absoluto	Er. Relativo
-0.049067674	-0.049067672	2x10^9	-4.840x10^{-6} %
-0.042062489	-0.042062487	1x10^9	-2.612x10^{-6} %
-0.035055234	-0.035055234	0x10^0	-1.260x10^{-6} %
-0.028046256	-0.028046256	0x10^0	-5.160x10^{-7} %
-0.021035899	-0.021035899	0x10^0	-1.630x10^{-7} %
-0.014024507	-0.014024507	0x10^0	-3.200^{-8} %
-0.007012426	-0.007012426	0x10^0	-2.000x10^{-6} %
0.000000000	0.000000000	0x10^0	0.000 %
0.007012426	0.007012426	0x10^0	2.000x10^{-9} %
0.014024507	0.014024507	0x10^0	3.200x10^{-8} %
0,021035899	0,021035899	$0x10^0$	$5{,}160x10^{-7}$ %
0,028046256	0,028046256	$0x10^0$	$1{,}216x10^{-7}$ %
0,035055234	0,035055234	$0x10^0$	$1{,}260x10^{-6}$ %
0,042062489	0,042062487	$1x10^{-9}$	$2{,}612x10^{-6}$ %
0,049067674	0,049067672	$2x10^0$	$4{,}840x10^{-6}$ %

Tabla con aproximaciones obtenidas, error absoluto y error relativo

4. Método de Remez

4.1. Teoría

El método de Remez es un algoritmo que se usa para encontrar aproximaciones simples a funciones, generalmente, estas funciones se encuentran en un subespacio de polinomios de Chebyshev de orden n continuos en un intervalo $[a, b]$ para obtener mejores aproximaciones. El polinomio de mejor aproximación dentro del subespacio se define como el que minimiza la diferencia absoluta máxima entre el polinomio y la función. En este caso, la solución se precisa mediante el teorema de equioscilación.

El algoritmo a grandes pasos es:

- Resolver el sistema de ecuaciones.
- Usar b_n como coeficiente formar el polinomio $P(n)$.
- Encontrar el conjunto M de puntos con el error máximo local $|P_n - f(x)|$

- Si los errores en cada $m \in M$ son de igual magnitud y se alternan en el signo, entonces P_n es el polinomio de aproximación minimax. Si no, se reemplaza x con M y se itera nuevamente.

4.2. Solución

Para comenzar la solución de la aproximación a la función $\sin(x)$ en el intervalo $[\frac{-pi}{64}, \frac{pi}{64}]$ mediante el método de Remez es necesario comenzar por elegir el grado del polinomio aproximación que queremos calcular. Para el caso actual, por la corta magnitud del intervalo y la simplicidad de la función en el mismo intervalo se elegirá un polinomio de grado $n = 3$. Procederemos a hallar una cantidad de puntos $n + 2$ en el intervalo dado, sin embargo, es muy recomendable que los puntos dados no sean equidistantes.

El fenómeno de Runge es un problema que sucede cuando se usa interpolación polinómica con nodos equidistantes y que puede hacer que el error de la interpolación misma tienda al infinito, la solución mas clásica para contrarrestar este fenómeno reside en los polinomios de Chebyshev. Nombrados en honor a Pafnuti Chebyshov, los polinomios de Chebyshev son muy conocidos en el análisis numérico por su capacidad para crear nodos de Chebyshev. En pocas palabras, mediante la creación de estos nodos podemos ubicar puntos en el intervalo que no son equidistantes pero que aun así son una representación fiel del segmento de recta del intervalo. La formula usada para la creación de los nodos es mostrada a continuación

$$x_k = \frac{1}{2}(a + b) + \frac{1}{2} * (b - a) * cos(\frac{2k-1}{2n} * \pi); \quad k = 1 \ldots n$$

Sean $a = \frac{-\pi}{64}$ y $b = \frac{\pi}{64}$ las variables que representan los limites del intervalo, k representa el iterado que va desde 1 hasta n, recordando que se le aumentan 2 unidades al grado del polinomio deseado, en nuestro caso $n = 5$, tras aplicar esta formula a nuestro caso obtuvimos la siguiente serie de puntos que nombraremos x:

$$x = [-0{,}04668488, \ -0{,}02885284, \ 0{,}00000000, \ 0{,}02885284, \ 0{,}04668488]$$

A este punto solo hemos comenzado por la obtención de los puntos necesarios para el método de Remez, evitando el fenómeno de Runge. El paso mas importante y característico del método de Remez es la solución del sistema de ecuaciones planteado a continuación:

$$\begin{pmatrix} 1 & x_0 & x_0^2 & ... & x_0^n & E \\ 1 & x_1 & x_1^2 & ... & x_1^n & -E \\ 1 & x_2 & x_2^2 & ... & x_1^n & E \\ ... & ... & ... & ... & ... & \\ 1 & x_{n+1} & x_{n+1}^2 & ... & x_{n+1}^n & (-1)^i E \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ ... \\ b_n \\ E \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ ... \\ f(x_{n+2}) \end{pmatrix}$$

Donde la matriz de tamaño $n{*}n$ tiene en su primera columna de izquierda a derecha una cantidad n de unos. Luego en el centro se ubican los nodos de Chebyshev, aumentando su potencia hasta que n es igual al grado del polinomio deseado, finalmente la ultima columna corresponde a una cantidad n de alternaciones positivas y negativas del error de tolerancia con la que nos queremos aproximar a la función objetivo. El vector siguiente contiene los valores que queremos hallar mediante el sistema de ecuaciones. Por ultimo, el vector de resultados contiene los resultados de la función objetivo, esto es el resultado de evaluar los nodos de Chebyshev en la función $\sin(x)$.Los valores reemplazados y el resultado del sistema de ecuaciones con única solución es mostrado a continuación. Es importante recordar que el grado del polinomio que estamos buscando en este ejemplo es de 3.

$$\begin{pmatrix} 1 & -0{,}04668488 & 0{,}0021794778 & -0{,}0001017487 & 0{,}000001 \\ 1 & -0{,}02885284 & 0{,}0008324864 & -0{,}0000240196 & -0{,}000001 \\ 1 & 0{,}00000000 & 0{,}0000000000 & 0{,}0000000000 & 0{,}000001 \\ 1 & 0{,}02885284 & 0{,}0008324864 & 0{,}0000240196 & -0{,}000001 \\ 1 & 0{,}04668488 & 0{,}0021794778 & 0{,}0001017487 & 0{,}000001 \end{pmatrix} \begin{pmatrix} -0{,}04666792 \\ -0{,}02884884 \\ 0{,}00000000 \\ 0{,}02884884 \\ 0{,}04666792 \end{pmatrix}$$

$Coef = [-0{,}000000000000,\ 0{,}999999984881,\ 0{,}000000000000,\ -0{,}166641568404,\ 0{,}000000000002]$

El vector que encontramos es el vector que contiene los coeficientes que nos permiten construir el polinomio solución. A continuación procedemos a verificar si hemos alcanzado un nivel de exactitud aceptable para la solución del problema:

$$P(x) = -0{,}999999984881x + 0{,}166641568404x^3, 0{,}000000000002x^4]$$

$sin(x)$	$x + \frac{x^3}{3!}$	Er. Absoluto	Er. Relativo
$-0{,}04666792$	$-0{,}04666792$	0,0000000000000094	0,0000000000202214 %
$-0{,}02884884$	0,02884884	0,0000000000000059	0,0000000000202884 %
0,00000000	0,00000000	0,0000000000000000	0,0000000000000000 %
0,02884884	0,02884884	0,0000000000000059	0,0000000000203004 %
0,04666792	0,04666792	0,0000000000000095	0,0000000000202660 %

Tabla con aproximaciones obtenidas, error absoluto y error relativo

Como podemos apreciar en la tabla, el polinomio aproximación cumplió su objetivo de manera precisa, presentando unos porcentajes de error extremadamente bajos. En caso de que esto no hubiera sucedido se hubieran tenido que seguir la siguiente serie de pasos: primero se debía restar las función objetivo con el polinomio solución creado para hallar una nueva función que nos permitiera la creación de nuevos puntos para la siguiente iteración, luego se tenían que hallar las raíces de la derivada de la función creada, usando cualquier método posible. Estas raíces de la derivada se convertirían en los nuevos puntos para reemplazar en la matriz del comienzo e iniciar una nueva iteración del método de Remez, reemplazando así a los nodos de Chebyshev

4.3. Referencias

- Muller, J. 2005. Elementary Functions, Algorithms and Implementation (2nd ed.)