Pontificia Universidad Javeriana
Departamento de Matemática
Análisis Numérico

Reto 1

Marco Antonio Valencia Dueñas
ma_valencia@javeriana.edu.co

Nicolás David Gil Hernandez
nicolas_gil@javeriana.edu.co

Paula Catalina Piñeros Pardo
pineros.paula@javeriana.edu.co

Profesora: Eddy Herrera Daza

Bogotá D.C., 14 de Febrero del 2020

1. Evaluación de un polinomio

Evaluar polinomios o funciones en general tiene muchos problemas, aún para el software profesional. Como se requiere poder evaluar el polinomio en las raíces 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 más eficiente. Un método visto es Horner:

$$b_0 = a_0$$

$$b_k = a_k + b_{k-1}x_0(k = 1, ..., n-1)$$

$$b_n = P(x_0)$$

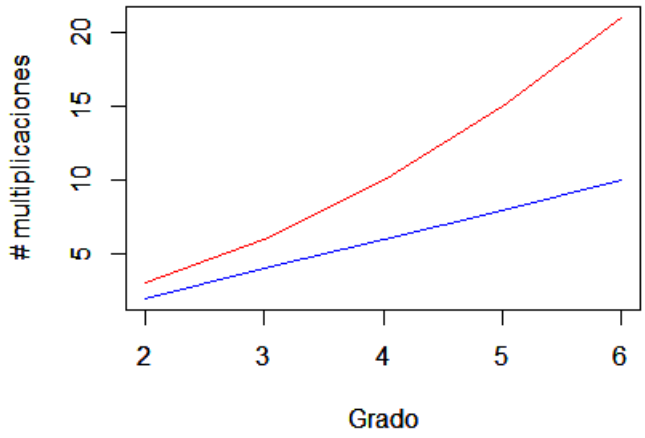
1. Implemente en R el método de Horner para evaluar f'(x0), tenga en cuenta la precisión de la computadora o unidad de redondeo.
2. Implemente el método de Horner si se realiza con números complejos, tenga en cuenta la precisión.

1.1. Implementar en R el método de Horner para evaluar f'(x0)

El método de Horner, también conocido como 'multiplicación anidada', busca hallar de manera eficiente el valor del polinomio haciendo uso del menor número de veces posibles la multiplicación. Este método, también es útil para hallar el valor de la derivada de un polinomio, y a pesar que la cantidad de multiplicaciones que se usarían para evaluar un polinomio sin derivar es menor que la cantidad de multiplicaciones que se aplican al calcular la derivada sigue siendo mejor utilizar este método, debido que si se calculara la derivada de la manera tradicional se utilizarían más multiplicaciones.Horner A continuación una comparación

Grado	Tradicional	Horner
6	21	10
5	15	8
4	10	6
3	6	4
2	3	2

Cuyo comportamiento se ve mejor evidenciado en la siguiente gráfica:



Lo que evidencia que el método de Horner es mejor que el método tradicional en lo que respecta a algortimos, debido a que la cantidad de operaciones que debe hacer es menor.

R maneja doble precisión, por lo que los errores de truncamiento que pueda realizar la máquina internamente son muy pequeños. Por ende, como tal en el código no se tuvo en cuenta dichos posbiles errores.

1.2. Implementar el método de Horner con números complejos

Se debe tener en cuenta que los números complejos son el conjunto de números más grandes, abarca el conjunto de números reales y el conjunto de números imaginarios. Un número complejo se expresa de la forma

$$z = x + y * i$$

Teniendo en cuenta que tanto x, como y pertenencen al conjunto de los reales.Complejos Es importante tener claro esto debido a que la cantidad de multiplicaciones usando el método de Horner cambia debido a los diferentes casos que se pueden presentar:

1. Se multipliquen dos expresiones con parte real y parte imaginaria, pues esto representaría 4 multiplicaciones

$$(x + y * i)(a + b * i)$$

2. Se multiplique una expresión con parte real y la otra expresión con ambas partes, o una expresión con parte imaginaria y la otra expresión con ambas partes, esto implicaría 2 multiplicaciones

$$(x)(a + b * i)$$

$$(y * i)(a + b * i)$$

$$(x + y * i)(a)$$

$$(x + y * i)(b * i)$$

3. Se multipliquen dos expresiones en la que cada una sola tiene una parte(real o imaginaria)

$$(x)(a)$$

$$(x)(b * i)$$

$$(y * i)(a)$$
$$(y * i)(b * i)$$

En sí, el algoritmo para evaluar un polinomio o la derivada del polinomio no cambia, debido a que R reconoce este tipo de número y lo opera sin problema, esto si se defina la variable antes de usarla: (ejemplo)

$$sqrt(-2 + 0i)$$

Así el programa calcularía bien la raíz cuadrada de -2

$$sqrt(-2)$$

El programa generaría error y no calcularía la raíz de -2

ComplejosR

Es por esto que en este punto se realizó una función de más, la cual se llama 'numMult', cuyos parámetros son dos números. En dicha función a cada número se le saca su parte real e imaginaria usando las funciones Re() e Im() respectivamente. Teniendo en cuenta esos datos se hizo una sumatoria con el número de partes reales e imaginaras que sean diferentes de 0 y dependiendo del total de dicha sumatoria se retornará el número de multiplicaciones necesarias si dichos números ingresados en los parametros se multiplicaran.
Al tener una función que ya calcula el número de multiplicaciones lo único que cambia de los códigos originales es la forma de calcular el número de multiplicaciones.
En este caso se decidió que el algortimo calcule el valor del polinomio y el valor de la derivada, así como el número de multiplicaciones y sumas de cada caso. Usando el polinomio:

$$2x^4 - 3x^2 + 3x - 4$$

El valor del polinomio evaluado en 2 - 3i, es igual a -221 + 267i
Número de multiplicaciones: 14
Número de sumas: 4 El valor de la derivada es -377 - 54i
Número de multiplicaciones: 20
Número de sumas: 6

A pesar que al usar números complejos aumenta el número de multiplicaciones siguen siendo menor que si se calculará de la forma tradicional A continuación se presentarán algunas comparaciones usando el siguiente polinomio:

$$2x^4 - 3x^2 + 3x - 4$$

Valor	Forma tradicional	Método Horner
2 – 3i	20	14
-2i	8	5
6	8	5

Si se usara un polinomio de grado 10, se puede ver que la diferencia entre ambos se vuelve mayor.

Valor	Forma tradicional	Método Horner
$2-3i$	164	38
-2i	56	17
6	56	17

Teniendo en cuenta esto, podemos demostrar aun más la utilidad del método de Horner, pues la cantidad de multiplicaciones se ve reducida de manera importante.

2. Óptima aproximación polinómica

La implementación de punto flotente de una función en un intevalo a menudo se reduce a una aproximación polinómica, siendo el polinomio de Remez a veces conduce a cancelaciones catastróficas. El algoritmo Remez es una metodología para localizar la aproximación racional minimax a una función. Las cancelaciones que también hay que considerar en el caso del método de Horner, suceden cunado 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 más grande. Aplicar esta técnica para

$$f(x) = sin(x) en el intervalo [-\pi/64, \pi/64]$$

con una precisión deseada doble. Para cada caso, evalue la aproximación polinómica de la función, el error relativo y el número de operaciones necesarias.
1. Aplique una aproximación de Taylor
2. Implemente el método de Remez

2.1. Aproximación de Taylor

El teorema de Taylor nos permite dar una representación de una función de una manera mucho más sencilla, pues la convierte en un polinomio de grado n y un error. Para el caso de sen(x):

$$sen(x) = [x - (1/3!) + ... + ((-1^n)/(2n+1)!) * x^{2n+1}$$

Esta aproximación es bastante buena puesto que potencialmente con una se lograría una precisión de 2 elevado a la -68 solo mediante ocho iteraciones o un polinomio de grado ocho. Se utilizaron distintas funciones para aproximar el resultado de este ejercicio a este resultado potencial, entre las cuales se encuentran las siguientes:

- factorial:saca el factorial de un entero x
- redondeoPiso: Función para redondear números según precisión requerida
- cifras : transforma una precisión en un entero de cifras significativas

- taylorseno: realiza iterativamente el polinomio de taylor

El algortimo funciona mediante un ciclo que va aumentando el grado del polinomio para ir aumentando la precisión del valor que la función evalúa. Cunado la enésima iteración es igual a la enésima más 1 el porgrama y evalúa cual fue la cantidad de iteraciones que realizó, que en otras palabras es el grado al que el polinomio tuvo que llegar para cumplir el requisito de precisió n dado.

2.2. Método de Remez

Es un algoritmo iterativo eficiente que calcula el polinomio minimax. Para inicializar el algoritmo, necesitamos un conjunto de n + 2 puntos en el intervalo [a, b]. Se podrían elegir diferentes puntos iniciales, pero una opción común son los nodos Chebyshev. Esto se debe a que los nodos de Chebyshev no son propensos al fenómeno Runge y minimizan el error de interpolación. Por lo tanto, el polinomio que pasa a través de los nodos de Chebyshev es una buena estimación inicial para el polinomio minimax. Deje que el polinomio Pn (x) que pasa a través de los nodos de Chebyshev.

Se puede en matris de la siguiente forma:

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

El teorema de Taylor nos permite dar una representación de una función de una manera mucho más sencilla, pues la convierte en un polinomio de grado n y un error. Para el caso de sen(x):