Informe laboratorio análisis numérico
Práctica No. 8

Brayan Barajas
Escuela de ingeniería de sistemas e informática
Universidad Industrial de Santander
01 de marzo de 2020

1 Introducción

En esta práctica de laboratorio se estudiará la diferenciación numérica por medio de MATLAB; se implementará un código ya dado junto con algunas fórmulas de aproximación de derivadas. Usando el algoritmo, se aproximarán las derivadas de dos funciones en un punto x dado, así mismo se usarán las fórmulas para determinar la aproximación de otra función para comparar y analizar la eficiencia de cada fórmula de acuerdo a h y sus posibles errores. Para el éxito de la práctica de laboratorio se cuenta con la documentación de MATLAB, además de los conocimientos adquiridos previamente en clase y la ayuda del docente de laboratorio.

2 Desarrollo

2.1. Use el Programa 6.1 para aproximar la derivada de cada una de las siguientes funciones en el punto x dado; las aproximaciones deberían tener precisión de 13 cifras decimales. Nota: Puede que sea necesario cambiar los valores de max1 y del punto inicial h del programa.

Para analizar la tolerancia, se tiene que

$$error < \frac{10^{1-d}}{2}, donde\ d\ son\ las\ cifras\ significativas$$

Con esto, se puede estimar que la tolerancia es de 5e-13. Para ambos casos, se tomará como h inicial h=7/8.
Además, para poder visualizar la precisión de 13 cifras solicitada, es necesario utlizar *format long,* para ver todos los dígitos necesarios.

a) $f(x) = 60x^{45} - 32x^{33} + 233x^5 - 47x^2 - 77\,, x = 1/\sqrt{3}$

Al aplicar la función dada, se obtienen los siguientes resultados:

```
L1 =

   1.0e+08 *

   0.000000008750000   6.694028185801526                   0
   0.000000000875000   0.000000811332767   6.694027374468759
   0.000000000087500   0.000000752329587   0.000000059003181
   0.000000000008750   0.000000751740893   0.000000000588693
   0.000000000000875   0.000000751735006   0.000000000005887
   0.000000000000087   0.000000751734948   0.000000000000059
   0.000000000000009   0.000000751734947   0.000000000000001
   0.000000000000001   0.000000751734947                   0
   0.000000000000000   0.000000751734936   0.000000000000011
```

Figura 1. Resultado función inciso a)

Con esto, se tiene que $f'\left(\frac{1}{\sqrt{3}}\right) = 75.1734947$ según los resultados arrojados por la función y teniendo en cuenta el menor error obtenido, que es en n=8; además se puede apreciar que el error estuvo muy por debajo del límite establecido de 5e-13, tanto que no se alcanza a conocer con 15 cifras decimales, porlo que el grado de precisión es bastante alto.

b) $\boldsymbol{f(x) = tan\left(cos\left(\frac{\sqrt{5}+sen(x)}{1+x^2}\right)\right), x = \frac{1+\sqrt{5}}{3}}$

Es importante aclarar que al implementar f(x) en el script para correr la función, se tuvieron en cuenta las funciones trigonométricas radianes.

Al aplicar la función dada para evaluar la derivada en el punto dado, se obtiene:

```
L2 =

   0.875000000000000   1.056815853559831                   0
   0.087500000000000   1.229003060544123   0.172187206984292
   0.008750000000000   1.228601648345573   0.000401412198550
   0.000875000000000   1.228597465642412   0.000004182703162
   0.000087500000000   1.228597423798629   0.000000041843782
   0.000008750000000   1.228597423373414   0.000000000425215
   0.000000875000000   1.228597423336935   0.000000000036479
   0.000000087500000   1.228597419990406   0.000000003346529
```

Figura 2. Resultado función inciso b)

Con esto, se tiene que $f'\left(\frac{1+\sqrt{5}}{3}\right) = 1.228597423336935$ según los resultados arrojados por la función y teniendo en cuenta el menor error obtenido, que es en n=7; además, se puede apreciar el error a la derecha del resultado.

2.2. Sea f(x) = cos(x).
Se parte de que se utiliza la función en **radianes**.

- Fórmula (6):

$$f''(x_0) = \frac{f_1 - 2f_0 + f_{-1}}{h^2} + E(f,h)$$
$$E(f,h) = \frac{4\varepsilon}{h^2} + \frac{h^2 f^{(4)}(c)}{12}$$

- Fórmula (12):

$$f''(x_0) = \frac{-f_2 + 16f_1 - 30f_0 + 16f_{-1} - f_{-2}}{12h^2} + E(f,h)$$
$$E(f,h) = \frac{16\varepsilon}{3h^2} + \frac{h^4 f^{(6)}(c)}{90}$$

Se usa Matlab para realizar los cálculos, por lo que se utiliza el ε de matlab que es $\varepsilon = 2^{-52}$ para calcular el error.

a) Use la fórmula (6) con h = 0,05 para aproximar f ''(1)

$$f_1 = \cos(1 + 0.05) = 0.497571047891727$$
$$f_0 = \cos(1) = 0.540302305868140$$
$$f_{-1} = \cos(1 - 0.05) = 0.581683089463883$$

$$E(f,h) = \frac{4\varepsilon}{0.05^2} + \frac{0.01^2 * 1}{12} = 2.083333336886047 * 10^{-4}$$

$$f''(1) = -0.540189752267617 + 2.083333336886047 * 10^{-4}$$
$$f''(1) = -0.539981418933928$$

b) Use la fórmula (6) con h = 0,01 para aproximar f ''(1)

$$f_1 = \cos(1 + 0.01) = 0.531860721374355$$
$$f_0 = \cos(1) = 0.540302305868140$$
$$f_{-1} = \cos(1 - 0.01) = 0.548689860581588$$

$$E(f,h) = \frac{4\varepsilon}{0.01^2} + \frac{0.01^2 * 1}{12} = 8.333342215117531 * 10^{-6}$$

$$f''(1) = -0.540297803365286 + 8.333342215117531 * 10^{-6}$$
$$f''(1) = -0.540289470023071$$

c) Use la fórmula (12) con h = 0,1 para aproximar f ′′(1)

$$f_1 = \cos(1 + 0.1) = 0.453596121425577$$
$$f_0 = \cos(1) = 0.540302305868140$$
$$f_{-1} = \cos(1 - 0.1) = 0.621609968270664$$
$$f_2 = \cos(1 + 2 * 0.1) = 0.362357754476674$$
$$f_{-2} = \cos(1 - 2 * 0.1) = 0.696706709347165$$

$$E(f, h) = \frac{16\varepsilon}{3 * 0.1^2} + \frac{0.1^4 * 1}{12} = 1.111111466382479 * 10^{-6}$$

$$f''(1) = -0.540301706068029 + 1.111111466382479 * 10^{-6}$$
$$f''(1) = -0.540300594956563$$

d) ¿Qué respuesta, (a),(b) o (c), es más precisa?
Para saber esto, debemos comparar cada una de las aproximaciones con el valor "real" de $f''(1)$, teniendo en cuenta que $f''(x) = -\cos(x)$. Por lo tanto,

$$f''(1) = -\cos(1) = -0.540302305868140$$

Al comparar cada uno de los valores obtenidos en cada inciso, se puede observar que el del inciso c) es el más preciso ya que se acerca más a este resultado. Esto nos dice que la fórmula (12) permite utilizar h más grandes para obtener un resultado más cercano.

$$Error = |(-0.540302305868140) - (-0.540300594956563)|$$
$$= -1.710911576968677 * 10^{-6}$$

3 Anexos

difflim.m

```
function [L,n]=difflim( f ,x, toler )
    %Input ? f is the function input as a string ’ f ’
    % ? x is the differentiation point
    % ? toler is the desired tolerance
    %Output ? L=[H’ D’ E’]: H is the vector of step sizes
    % D is the vector of approximate derivatives
    % E is the vector of error bounds
    % ? n is the coordinate of the ”best approimation”
    % NUMERICAL METHODS: MATLAB Programs
    %(c) 1999 by John H. Mathews and Kurtis D. Fink
    %To accompany the textbook:
    %NUMERICAL METHODS Using MATLAB,
    %by John H. Mathews and Kurtis D. Fink
    %ISBN 0?13?270042?5, (c) 1999
    %PRENTICE HALL, INC.
    %Upper Saddle River, NJ 07458
    max1=30;
    h=1/10;
    H(1)=h;
    D(1)=(feval(f,x+h)-feval(f,x-h))/(2*h);
    E(1)=0;
    R(1)=0;
    for n=1:2
       h=h/10;
       H(n+1)=h;
       D(n+1)=(feval(f,x+h)-feval(f,x-h))/(2*h);
       E(n+1)=abs(D(n+1)-D(n));
       R(n+1)=2*E(n+1)*(abs(D(n+1))+abs(D(n))+eps);
    end
    n=2;
    while((E(n)>E(n+1))&(R(n)>toler))&n<max1
     h=h/10;
     H(n+2)=h;
     D(n+2)=(feval(f,x+h)-feval(f,x-h))/(2*h);
     E(n+2)=abs(D(n+2)-D(n+1));
     R(n+2)=2*E(n+2)*(abs(D(n+2))+abs(D(n+1))+eps);
     n=n+1;
    end
    n=length(D)-1;
    L=[H' D' E'];
end
```

Run_1_difflim_Brayan_Barajas.m

```
format long
f=@(x) 60*x.^45-32*x.^33+233*x.^5-47*x.^2-77;
[L1,n1]=difflim(f,1/sqrt(3),10^(-13))

g=@ (x) tan(cos((sqrt(5)+sin(x))/(1+x.^2)));
[L2,n2]=difflim(g,(1+sqrt(5))/3,10^(-13))
```