Taller Derivadas

Nicolas Camacho PlazasDaniela Cortes Antonio Mateo Florido Sanchez

a

-Teniendo en cuenta lo anterior, genere una tabla para evaluar el valor aproximado $f(x) = xcos$ de $f(1.8)$ para los siguientes valores de h:
0.1, 0.01, 0.001, 0.0001.

```
library(numDeriv)
library(pracma)

##
## Attaching package: 'pracma'

## The following objects are masked from 'package:numDeriv':
##
##     grad, hessian, jacobian

vectorh = c(0.1,0.01,0.001,0.0001)

f = function(x) x*cos(x)
for (i in 1:length(vectorh)){
  print(fderiv(f,1.8,,h=vectorh[i]))
}

## [1] -1.976073
## [1] -1.980087
## [1] -1.980127
## [1] -1.980128
```

b

-Estime el valor aproximado de las cotas del error para el problema anterior.
Teniendo en cuenta la siguiente formula del error:
$|\frac{hM}{2}|$
donde $M = f''(E(x))$
Entonces se puede decir que la cuota del error de truncamiento es:

```
library(numDeriv)
library(pracma)
f = function(x) x*cos(x)
fp = function(x) cos(x)- x*sin(x)
```

```
vectorh = c(0.1,0.01,0.001,0.0001)
resul=c()
cotas=c()
for (i in 1:length(vectorh)){
  resul[i]<-fderiv(f,1.8,h=vectorh[i])
  print(abs(vectorh[i]*grad(fp, x = 1.8)/2))
}
```

```
## [1] 0.07693657
## [1] 0.007693657
## [1] 0.0007693657
## [1] 7.693657e-05
```

c

-Cual es elvalor de h que proporciona una aproximacion con una precision de 10^{-4}
Para identificar el h necesario se va a utilizar el error absoluto.

```
library(numDeriv)
library(pracma)
f = function(x) x*cos(x)
fp = function(x) cos(x)- x*sin(x)
h_aux=0.1
maxIter=100

exacto=fp(1.8)
for (i in 1:maxIter){
 if(abs(round(fderiv(f,1.8,h=h_aux),4)-round(exacto,4))==0){
   print("El valor de h necesario para alcanzar la precision deseada 
es:")
   print(h_aux)
   break;
 }

  h_aux=h_aux/10
}
```

```
## [1] "El valor de h necesario para alcanzar la precision deseada es:"
## [1] 0.01
```

d

-Supongase que se tienen tres puntos dados por $x_0; x_1 = x_0 + h; x_2 = x_0 + 2h$
Encuentre la formula conocida de tres puntos para determinar una aproximación de $f'(x_0)$ donde ξ_0 se encuentra entre x_0 y $x_0 + 2h$. Utilice esta formula para encontrar $f'(1.8)$
La formula que permite hayar esto es:

$$f'(x_j) = \frac{-f(x_{j+2}) + 4f(x_{j+1}) - 3f(x_j))}{2h}$$

Suponiendo que:
$x_j = x_0$
$x_{j+1} = x_1 = x_0 + h$
$x_{j+2} = x_2 = x_0 + 2h$
$f(x) = xcos(x)$
Entonces al remplazar conseguimos la siguiente ecuacion:
$f'(x_0) = \frac{-f(x_2)+4f(x_1)-3f(x_0)}{2h}$, reemplazando tenemos que
$f'(x_0) = \frac{-x_0Cos(x_0+2h)+4x_0Cos(x_0+h)-3x_0Cos(x_0)}{2h}$ luego,
$$f'(x_0) = \frac{x_0Sin(\frac{h}{2})(Sin(x_0 + \frac{3h}{2}) - 3Sin(x_0 + \frac{h}{2}))}{h}$$

```
fp=function(x,h) (x*sin(h/2))*(sin(x+3*h/2)-3*sin(x+h/2))/h
h=0.01
print(fp(1.8,h))

## [1] -1.752984
```

e

-Realice una modificacion de la formula de los tres puntos, tomando valores entre $(x_0 - h)$ y $(x_0 + h)$ y compare la magnitud del error.
Teniendo en cuenta que ahora los puntos no estan adelante de x_0 sino a su alrededor, entonces la formula seria:
$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h}$$
Suponiendo que $x_i = x_0$; $x_{i-1} = x_0 - h$; $x_{i+1} = x_0 + h$ entonces la nueva formula sería:
$f'(x) = \frac{f(x+h)-f(x-h)}{2h}$ lo que es igual a:
$$f'(x) = \frac{x(Cos(x + h) - Cos(x - h))}{2h}$$

```
library(numDeriv)
library(pracma)
real=function(x) cos(x)-x*sin(x)
fp2= function(x,h) (x*(cos(x+h)-cos(x-h)))/(2*h)
h=0.01
print("Al evaluar la formula nueva en 1.8 se obtiene : ")

## [1] "Al evaluar la formula nueva en 1.8 se obtiene : "

print(fp2(1.8,h))

## [1] -1.752897

print("Esto tiene un error absoluto de: ")

## [1] "Esto tiene un error absoluto de: "
```

```
print(abs(real(1.8)-fp2(1.8,h)))

## [1] 0.2272313
```

f

-Utilice la formula para cinco puntos alrededor de x_0 y apliquela y comparela con todas las formulas anteriores.
La formula para cinco puntos alrededor de x_0 está dada por:

$$f'(x_0) = \frac{-x_5 * Cos(x_5) + 8x_4Cos(x_4) - 8x_2Cos(x_2) + x_1Cos(x_1)}{12h}$$

```
library(numDeriv)
library(pracma)
fp=function(x) cos(x)-x*sin(x)
fp1 = function(x,h) (x*sin(h/2))*(sin(x+3*h/2)-3*sin(x+h/2))/h
fp2 = function(x,h) (x*(cos(x+h)-cos(x-h)))/(2*h)
fp3 = function (x,h){
  x1=x-0.2
  x2=x-0.1
  x4=x+0.1
  x5=x+0.2
  return ((-x5*cos(x5)+8*x4*cos(x4)-8*x2*cos(x2)+x1*cos(x1))/(12*h))


}
h=0.1
cat(formatC(c("Valor h", "Valor Derivada", "Valor Real", "Error
Absoluto"), width = -20, format = "f", flag = " "),"\n")

## Valor h                Valor Derivada       Valor Real           Error
Absoluto

cat(formatC(c(h, fp1(1.8,h), fp(1.8), abs(fp(1.8-fp1(1.8,h)))), width = -
20, format = "f", flag = " "),"\n")

##  0.1000              -1.7586              -1.9801              0.5272

cat(formatC(c(h, fp2(1.8,h), fp(1.8), abs(fp(1.8-fp2(1.8,h)))), width = -
20, format = "f", flag = " "),"\n")

##  0.1000              -1.7500              -1.9801              0.4921

cat(formatC(c(h, fp3(1.8,h), fp(1.8), abs(fp(1.8)-fp3(1.8,h))), width = -
20, format = "f", flag = " ",digits = 10),"\n")

##  0.1000000000        -1.9801182137        -1.9801278303
0.0000096165
```

g

-Aplique la formula adecuada para aproximar $f'(1.8)$ justifique su respuesta.
Depues de analizar los datos que se pueden apreciar en la tabla del punto anterior, se puede concluir que la mejor fórmula es la planteada en el punto f:

$$f'(x_0) = \frac{-x_5 * Cos(x_5) + 8x_4Cos(x_4) - 8x_2Cos(x_2) + x_1Cos(x_1)}{12h}$$

El criterio que permite llegar a esta afirmación es el de el error absoluto puesto que este expresa directamente la diferencia que existe entre el valor real y la formula elegida. Su error absoluto fue, por mucho, menor de todos y por esto se considera la mas indicada.

```r
library(numDeriv)
library(pracma)
fp=function(x) cos(x)-x*sin(x)
fp3 = function (x,h){
  x1=x-0.2
  x2=x-0.1
  x4=x+0.1
  x5=x+0.2
  return ((-x5*cos(x5)+8*x4*cos(x4)-8*x2*cos(x2)+x1*cos(x1))/(12*h))
}
h=0.1
cat(formatC(c("Valor h", "Valor Derivada", "Valor Real", "Error
Absoluto"), width = -20, format = "f", flag = " "),"\n")

## Valor h                Valor Derivada       Valor Real           Error
Absoluto

cat(formatC(c(h, fp3(1.8,h), fp(1.8), abs(fp(1.8)-fp3(1.8,h))), width = -
20, format = "f", flag = " ",digits = 10),"\n")

##  0.1000000000        -1.9801182137        -1.9801278303
0.0000096165
```

h

Para encontrar el h óptimo se va a tomar como ejemplo la función $f(x) = xe^x$. Cuando el error deje de disminuir con forme disminuya h, entonces ese será el h óptimo.

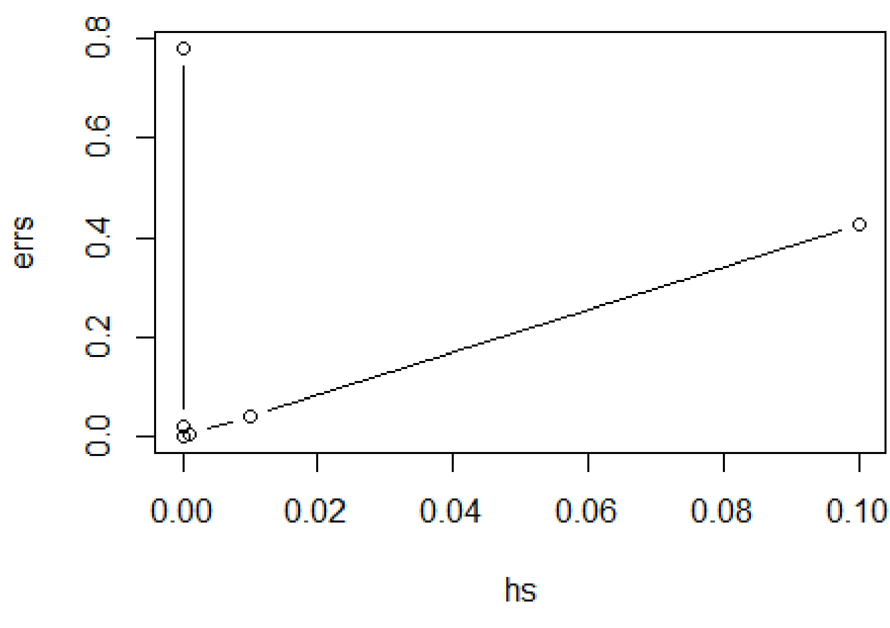
```r
library(numDeriv)
library(pracma)
f=function(x) x*exp(x)
r=5.43656365691809l
h=0.1
h_optimo=h
e_anterior=30
for (i in 1:15){
  d=(f(1+h)-f(1))/h
  e=abs(r-d)
```

```
  if(e_anterior<e){
  h_optimo=h
  break;
  }
  h=h/10
  e_anterior=e
}
print("El h optimo es:")

## [1] "El h optimo es:"

print(h_optimo,digists=10)

## [1] 1e-09
```

i

```
library(numDeriv)
library(pracma)
f=function(x) x*exp(x)
r=5.436563656918091
h=0.1
hs=c()
errs=c()
cat(formatC(c("h", "r", "d", "e"), width = -20, format = "f", flag = "
"),"\n")

## h                        r                        d                        e

for (i in 1:15){
  d=(f(1+h)-f(1))/h
  e=abs(r-d)
  errs[i]<-e
  hs[i]<-h
  cat(formatC(c(h, r, d, e), width = -20, format = "f", flag = " ",digits
= 15),"\n")
  h=h/10
}

##  0.100000000000000    5.436563656918091    5.863007978820320
0.426444321902229
##  0.010000000000000    5.436563656918091    5.477519670804032
0.040956013885941
##  0.001000000000000    5.436563656918091    5.440642892414527
0.004079235496436
##  0.000100000000000    5.436563656918091    5.436971417318581
0.000407760400490
##  0.000010000000000    5.436563656918091    5.436604431396929
0.000040774478838
##  0.000001000000000    5.436563656918091    5.436567733774210
0.000004076856119
```

```
##  0.000000100000000    5.436563656918091    5.436564070038229
0.000000413120138
##  0.000000010000000    5.436563656918091    5.436563643712588
0.000000013205503
##  0.000000001000000    5.436563656918091    5.436564087801797
0.000000430883706
##  0.000000000100000    5.436563656918091    5.436566752337056
0.000003095418965
##  0.000000000010000    5.436563656918091    5.436584515905450
0.000020858987359
##  0.000000000001000    5.436563656918091    5.437428285404166
0.000864628486075
##  0.000000000000100    5.436563656918091    5.435651928564766
0.000911728353326
##  0.000000000000010    5.436563656918091    5.417888360170763
0.018675296747328
##  0.000000000000001    5.436563656918091    6.217248937900876
0.780685280982785

plot(hs,errs,type = 'b')
```



j

-En un circuito con un voltaje $E(t)$ y una inductancia L se tiene que:
$E(T) = L\frac{di}{dt} + Ri$ donde R es la resistencia e i es la corriente. Medida de corriente en varios instantes de tiempo con $R = 0.0142\ ohms$ y $L = 0.98\ henries$.
t=1.00, 1.01, 1.02, 1.03, 1.0

i=3.10, 3.12, 3.14, 3.18, 3.24
Aproxime el voltaje para los valores de t
Suponiendo que el valor central de t es 1.02, entonces:

```r
e=function (h) {
  #Valor centrarl está en la posición 3
  c=3
  i=c(3.10,3.12,3.14,3.18,3.24)
  didt=(-i[c+2]+8*i[c+1]-8*i[c-1]+i[c-2])/(12*h)
  l=0.098
  r=0.142
  voltaje = l*didt+r*i[c]
  print(voltaje)
}
h=0.01
e(h)

## [1] 0.7235467
```