moodle_2_03-06-12-28

Alan Coila Bustinza

2022-06-02

```
library(knitr)     # For knitting document and include_graphics function
library(ggplot2)   # For plotting
library('png')
```

pregunta 1

```
img1_path <- "p1_2022-06-02_170205.png"
include_graphics(img1_path)
```

En un estudio de fotosíntesis se necesita el area de la hoja de una planta. Midiendo la anchura de la hora en intervalos de $\frac{1}{4}$ cm al largo del eje, el resultado es:

$$\begin{bmatrix} \textit{Distancia} & 0 & \frac{1}{4} & \frac{1}{2} & \frac{3}{4} & 1 & \frac{5}{4} & \frac{3}{2} \\ \textit{Anchura} & 3.466 & 3.474 & 3.481 & 3.489 & 3.497 & 3.504 & 3.512 \end{bmatrix}$$

Usad la regla de Simpson compuesta para encontrar una aproximación del area de la hoja.

```
x1 <- c(0,1/4,1/2,3/4,1,5/4,3/2)
y1 <- c(3.466,3.474,3.481,3.489,3.497,3.504,3.512)


polyinterp=function(x,y){
  if(length(x)!=length(y))
    stop ("La longitud de los vectores x e y debe ser la misma" )
  n=length(x)-1
  vandermonde=rep(1,length(x))
  for(i in 1:n)
  {
    xi=x^i
```

```r
    vandermonde=cbind(vandermonde,xi)
  }
  beta=solve(vandermonde,y, tol=1e-22)
  names(beta)=NULL
  return(beta)
}

myeval <- function(x, c) {
  f <- 0
  for (i in 1:length(c)) {
    f <- f + c[i] * x^(i - 1)
  }
  return(f)
}


simp=function(x,y)
{
  m <- length(x)-1
  a <- x[1]
  b <- x[m+1]
  cof <- polyinterp(x,y)
  x.ends=seq(a,b,length.out=m+1)
  y.ends=myeval(x.ends,cof)
  x.mids=(x.ends[2:(m+1)]-x.ends[1:m])/2+x.ends[1:m]
  y.mids=myeval(x.mids,cof)
  p.area=sum(y.ends[2:(m+1)]+4*y.mids [1: m ]+y.ends[1:m])
  p.area=p.area * abs(b - a) / (6 * m)
  return(p.area)
}


simp(x1,y1)
```

```
## [1] 5.2335
```

pregunta 2

```r
img1_path <- "p2_2022-06-03_090151.png"
include_graphics(img1_path)
```

Sabemos que cierta función pasa por los siguientes puntos:

$$\begin{bmatrix} x_i & 0 & 2 & 5 & 6 & 8 \\ f(x_i) & 8 & -1.314 & -2.946 & 13.06 & -7.604 \end{bmatrix}$$

A partir de estos puntos, da uno de los L_i necesarios para calcular el polinomio interpolador de Lagrange.

```r
x <- c(0,2,5,6,8)

Lx <- function(x){
  n <- length(x)
  mult <- function(a,b){
    k <- outer(a,b)
    u <- as.vector(tapply(k, row(k) + col(k), sum))
    return(u)
  }
  for(i in 1:n){
    f <- c(1)
    m <- c(1)
    for(j in 1:n){
      if(i!=j){
        tn <- c(-x[j],1)
        f <- mult(f,tn)

      # }
      # if(j<n+1){

        v <- x[i]-x[j]
        # print(cbind(x[i],x[j]))
        if(v!=0){
          m <- prod(m,v)
        }
      }
    }

    print('.......................')
    print(f)
    print(m)
  }
}
Lx(x)
```

```
## [1] "......................."
## [1]  480 -476  156  -21    1
## [1] 480
## [1] "......................."
## [1]    0 -240  118  -19    1
## [1] -144
## [1] "......................."
## [1]   0 -96  76 -16   1
## [1] 45
## [1] "......................."
## [1]   0 -80  66 -15   1
## [1] -48
## [1] "......................."
## [1]   0 -60  52 -13   1
## [1] 288
```

pregunta 3

```r
img1_path <- "p3_2022-06-03_114534.png"
include_graphics(img1_path)
```

Los valores de una cierta función $f(x)$ en unos puntos concretos están recogidos en la siguiente tabla de valores:

$$\begin{bmatrix} x_i & 1 & 2 & 4 & 5 & 8 \\ f(x_i) & 10 & 5 & 2 & 4 & 14 \end{bmatrix}$$

Seleccione una:

- a. El grado del polinomio interpolador es 4 y la última diferencia dividida es 0
- b. El grado del polinomio interpolador es 4 y la última diferencia dividida es $\frac{-5}{252}$
- c. El grado del polinomio interpolador es 4 y la última diferencia dividida es $\frac{7}{6}$

```r
x3 <- c(1,2,4,5,8)
y3 <- c(10,5,2,4,14)

dif_n <- function(x,y){
  k=length(x)
  nk <- function(x,y){
    mat <- matrix(0, length(x), length(x))
    mat[,1] <- y
    return (mat)
  }
  mat <- nk(x,y)
  val=1
  for(i in 1:k){
    vect=c()
    for(j in 1:(k-i)){
      if(j>0){
        a=mat[j+1,i]
        b=mat[j,i]
        vect <- c(vect ,(a-b)/(x[j+val]-x[j]))

      }
    }
    val=val+1
    if(i<k){
      mat[,i+1] <- vect[1:k]
```

```r
    }
  }

return(list(mat,k-1))
}
dif_n(x3,y3) # retorna la matriz de diferencias divididas y el grado del polinomio
```

```
## [[1]]
##      [,1]      [,2]      [,3]       [,4]        [,5]
## [1,]   10 -5.000000 1.1666667  0.0000000 -0.01984127
## [2,]    5 -1.500000 1.1666667 -0.1388889          NA
## [3,]    2  2.000000 0.3333333         NA          NA
## [4,]    4  3.333333        NA         NA          NA
## [5,]   14        NA        NA         NA          NA
##
## [[2]]
## [1] 4
```

```r
22
```

```
## [1] 22
```

pregunta 4

```r
img1_path <- "p4_2022-06-03_120503.png"
include_graphics(img1_path)
```

etodos numericos en ciencia de datos aula 2 ▶ General ▶ Cuestionario Bloque 2

Un río tiene una anchura de $\frac{13}{18}$ m. En función de la distancia a la orilla, la profundidad del río, medida en sección recta, viene dada por la tabla siguiente:

$$\begin{bmatrix} \textit{Distancia} & 0 & \frac{5}{36} & \frac{1}{3} & \frac{1}{2} & \frac{5}{9} & \frac{23}{36} & \frac{13}{18} \\ \textit{Profundidad} & 2.773 & 2.841 & 2.933 & 3.008 & a & 3.069 & 3.105 \end{bmatrix}$$

Si sabemos que es area de la sección es, aproximadamente, $2.125\ m^2$, usando la regla del trapecio compuesto dad una aproximación del valor a

```r
x3 <- c(0,5/36,1/3,1/2,23/36,13/18)
y3 <- c(2.773,2.841,2.933,3.008,3.069,3.165)

polyinterp=function(x,y)
```

```
{
  # comprobamos que la longitud de los vectores sea la misma
  if(length(x)!=length(y))
    stop ("La longitud de los vectores x e y debe ser la misma" )
  # calculamos el valor de n que es el grado del polinomio
  # a partir del numero de puntos menos 1
  n=length(x)-1
  # creamos la primera columna de la matriz de vandermonde
  vandermonde=rep(1,length(x))
  # iteramos para ir agregando las columnas siguientes según
  # el grado del polinomio
  for(i in 1:n)
  {
    # la matriz contiene columnas sucesivas de los valores de x
    # elevados a la nth potencia
    xi=x^i
    vandermonde=cbind(vandermonde,xi)
  }
  # resolvemos el sistema de ecuaciones
  beta=solve(vandermonde,y, tol=1e-22)
  # borramos los nombres de las columnas (xi)
  names(beta)=NULL
  # nos retorna los coeficientes del polinomio
  return(beta)
}

coef <- polyinterp(x3,y3)
horner=function(x,coefs)
  # esta función auxiliar recibe los argumentos:
  #       x     :     puntos a evaluar
  #       coefs :     los coeficientes del polinomio
{
  y=rep(0,length(x))
  for(i in length(coefs):1)
    y=coefs[i]+x*y

  return(y)
}
coef
```

```
## [1]   2.7730000   0.7798721  -3.9604271  17.4293756 -31.2304485  19.4688583
```

```
horner(5/9,coef)
```

```
## [1] 3.02781
```