Untitled

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_125011.png"
include_graphics(img1_path)
```

Dada la siguiente función definida a trozos:

$$f(x) = \begin{cases} a \cdot x^2 + -4 \cdot x + 4 \text{ si } x \in (-\infty, -1] \\ 2 \cdot x^2 + x - 2 \text{ si } x \in [4, \infty) \end{cases}$$

Y sabiendo que, usando interpolación lineal, la función pasa por el punto $(2, \frac{124}{5})$, calcula el valor de a

crearemos la funcion lineal para los do puntos que tenemos y evaluaremos en el 3er punto donde se encuentra "a"

```
xn <- c(2,4)
yn <- c(124/5,34)


myPhi <- function(x, n) {
  Phi <- matrix(1, length(x), n + 1)
  for (i in 1:n) {
    Phi[, i + 1] <- x^i # funciones base para el ajuste polinómico, segun el grado
  }
  return(Phi)
}

mylssolve <- function(A, y) {
  AT <- t(A)
  return((solve(AT %*% A)) %*% AT %*% y) # la función solve nos devuelve la inversa de la matriz
}
```

```r
mypolyfit <- function(x, y, n) {
  Phi <- myPhi(x, n) # construimos la matriz con myPhi
  c <- mylssolve(Phi, y) # resolvemos el sistema de ecuaciones normales con la función mylssolve
  return(c)
}

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

coef <- mypolyfit(xn,yn,1)

myeval(-1,coef)-8
```

```
## [1] 3
```

pregunta 2

```r
img1_path <- "p2_2022-06-02_130141.png"
include_graphics(img1_path)
```

Usando el método de Romberg, queremos calcular

$$\int_0^1 \frac{dx}{1+x}.$$

Una parte del esquema triangular toma la siguiente forma:

Nivel 0	*Nivel* 1	...
0.75000000		
0.70833333	0.69444444	
0.69702381	0.69325397	α
0.69412185	0.69315453	0.69314790
0.69339120	0.69314765	0.69314719

Entonces, el valor de α y la aproximación de la integral es:

```r
# primero definir la funcion
f2 <- function(x){
  return(1/(1+x))
}
# luego formula del trapecio compuesta para funcion f
```

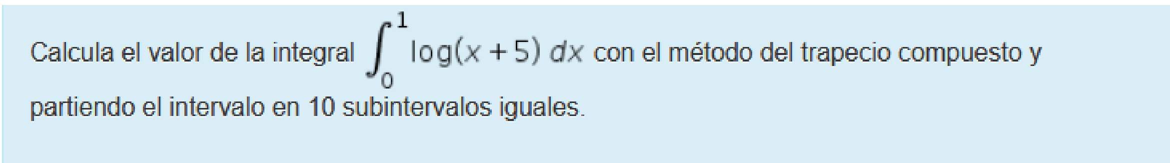
```r
trap=function(f,a,b,m)
{
  x=seq(a,b,length.out=m+1)
  y=f(x)
  p.area=sum((y[2:(m+1)]+y[1:m]))
  p.area=p.area*abs(b-a)/(2*m)
  return(p.area)
}
# para generar el primer nivel 0 ver el numero de valores en nivel 0:

n0 <- function(valores,f,a,b){

  n=c()
  for(i in 2^(0:(valores-1))){
    n=c(n,trap(f2,0,1,i))
  }
  return(n)
}
v0 <- n0(5,f2,0,1)
k=length(v0)
nk <- function(ncero){

  mat <- matrix(0, length(ncero), k)
  mat[,1] <- ncero
  # ((4*n1[i+1]-n1[i])/(4-1))
  # for(i in 2:k){
  #   a=i
  #   b=i+1
  #   mat[,k] <- 4**(k-1)*mat[k,k-1]
  # }
  return (mat)
}
mat <- nk(v0)

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]
      pot= 4**i
      vect <- c(vect ,(pot*a-b)/(pot-1))
    }
  }
  if(i<k){
     mat[,i+1] <- vect[1:k]
  }
}

mat
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] 0.7500000 0.6944444 0.6931746 0.6931475 0.6931472
```

```
## [2,] 0.7083333 0.6932540 0.6931479 0.6931472        NA
## [3,] 0.6970238 0.6931545 0.6931472        NA        NA
## [4,] 0.6941219 0.6931477        NA        NA        NA
## [5,] 0.6933912        NA        NA        NA        NA
```

pregunta 3

```r
img1_path <- "p3_2022-06-02_131607.png"
include_graphics(img1_path)
```



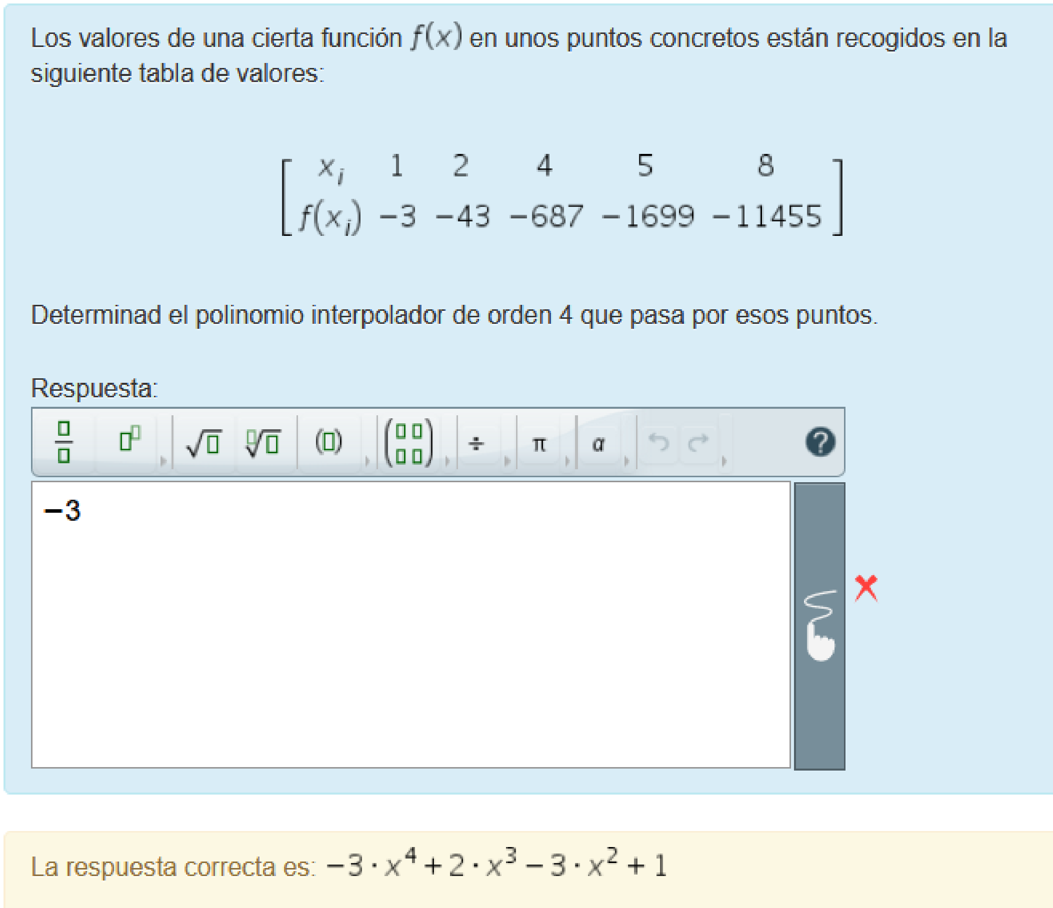
Calcula el valor de la integral $\int_0^1 \log(x+5)\,dx$ con el método del trapecio compuesto y partiendo el intervalo en 10 subintervalos iguales.

```r
func1 <- function(x){
  log((x+5),10)
}

trap=function(f,a,b,m)
{
  x=seq(a,b,length.out=m+1)

  y=f(x)
  p.area=sum((y[2:(m+1)]+y[1:m]))
  p.area=p.area*abs(b-a)/(2*m)
  return(p.area)
}
# para la funcion evaluada de 0 a 1 en 10 intervalos

trap(func1,0,1,10)
```

```
## [1] 0.7397509
```

pregunta 4

```r
img1_path <- "p4_2022-06-02_132623.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:

$$\begin{bmatrix} x_i & 1 & 2 & 4 & 5 & 8 \\ f(x_i) & -3 & -43 & -687 & -1699 & -11455 \end{bmatrix}$$

Determinad el polinomio interpolador de orden 4 que pasa por esos puntos.

Respuesta:

-3

La respuesta correcta es: $-3 \cdot x^4 + 2 \cdot x^3 - 3 \cdot x^2 + 1$

```
x1 <- c(1,2,4,5,8)
y1 <- c(-3,-43,-687,-1699,-11455)

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
```

```r
  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)
}

polyinterp(x1,y1)
```

```
## [1]  1  0 -3  2 -3
```