Área de dos curvas

El objetivo de la tarea es calcular el área bajo la curva de dos curvas, en este caso de las funciones de seno y de coseno. La siguiente implementación se realizó basándonos en el teorema de los trapecios. A continuación, se presenta el paso a paso de cómo se realizó el procedimiento.

Primer Paso

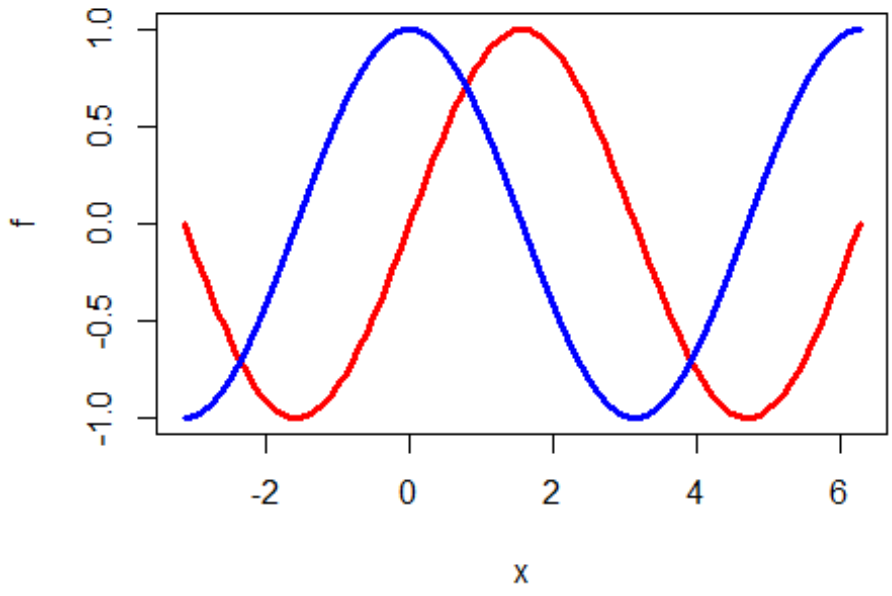
Definimos las funciones a las que se les va a calcular el área bajo la curva.

```
#Definición de las funciones
f<-function(x)
{
  sin(x)
}
g<-function(x)
{
  cos(x)
}
```

Segundo Paso

Graficamos las funciones.

```
#Graficas de las dos funciones
x <- seq(-pi,pi*2, by = 0.001)
plot(f, -pi, pi*2, lwd = 3, col = "red")
lines(x, g(x),lwd = 3, col = "blue")
```



Tercer Paso

Hacemos las particiones correspondientes para calcular el área bajo la curva y calculamos los valores en "**x**" y "**y**", guardando los valores en vectores de las respectivas particiones.

```
#Número de particiones
particiones <- 100

#Longitud entre los puntos de las particiones
n <- (pi+(pi*2))/particiones

#Vector con los valores en x de las particiones
xi <- seq(-pi,pi*2, by = n)

#Vectores de cada función con sus y correspondientes a los valores en x
de las particiones
y1 <- c(f(xi))
y2 <- c(g(xi))

approx.df <- data.frame(cbind(xi, y1, y2))
colnames(approx.df) <- c('X', 'Y1', 'Y2')
approx.df
```

```
               X            Y1            Y2
 1   -3.14159265 -1.224606e-16 -1.000000e+00
 2   -3.04734487 -9.410831e-02 -9.955620e-01
 3   -2.95309709 -1.873813e-01 -9.822873e-01
 4   -2.85884931 -2.789911e-01 -9.602937e-01
 5   -2.76460154 -3.681246e-01 -9.297765e-01
 6   -2.67035376 -4.539905e-01 -8.910065e-01
 7   -2.57610598 -5.358268e-01 -8.443279e-01
 8   -2.48185820 -6.129071e-01 -7.901550e-01
 9   -2.38761042 -6.845471e-01 -7.289686e-01
 10  -2.29336264 -7.501111e-01 -6.613119e-01
 11  -2.19911486 -8.090170e-01 -5.877853e-01
 12  -2.10486708 -8.607420e-01 -5.090414e-01
 13  -2.01061930 -9.048271e-01 -4.257793e-01
 14  -1.91637152 -9.408808e-01 -3.387379e-01
 15  -1.82212374 -9.685832e-01 -2.486899e-01
 16  -1.72787596 -9.876883e-01 -1.564345e-01
 17  -1.63362818 -9.980267e-01 -6.279052e-02
 18  -1.53938040 -9.995066e-01  3.141076e-02
 19  -1.44513262 -9.921147e-01  1.253332e-01
 20  -1.35088484 -9.759168e-01  2.181432e-01
 21  -1.25663706 -9.510565e-01  3.090170e-01
 22  -1.16238928 -9.177546e-01  3.971479e-01
 23  -1.06814150 -8.763067e-01  4.817537e-01
 24  -0.97389372 -8.270806e-01  5.620834e-01
 ……
```

```
 76   3.92699082 -7.071068e-01 -7.071068e-01
 77   4.02123860 -7.705132e-01 -6.374240e-01
 78   4.11548638 -8.270806e-01 -5.620834e-01
 79   4.20973416 -8.763067e-01 -4.817537e-01
 80   4.30398194 -9.177546e-01 -3.971479e-01
 81   4.39822972 -9.510565e-01 -3.090170e-01
 82   4.49247749 -9.759168e-01 -2.181432e-01
 83   4.58672527 -9.921147e-01 -1.253332e-01
 84   4.68097305 -9.995066e-01 -3.141076e-02
 85   4.77522083 -9.980267e-01  6.279052e-02
 86   4.86946861 -9.876883e-01  1.564345e-01
 87   4.96371639 -9.685832e-01  2.486899e-01
 88   5.05796417 -9.408808e-01  3.387379e-01
 89   5.15221195 -9.048271e-01  4.257793e-01
 90   5.24645973 -8.607420e-01  5.090414e-01
 91   5.34070751 -8.090170e-01  5.877853e-01
 92   5.43495529 -7.501111e-01  6.613119e-01
 93   5.52920307 -6.845471e-01  7.289686e-01
 94   5.62345085 -6.129071e-01  7.901550e-01
 95   5.71769863 -5.358268e-01  8.443279e-01
 96   5.81194641 -4.539905e-01  8.910065e-01
 97   5.90619419 -3.681246e-01  9.297765e-01
 98   6.00044197 -2.789911e-01  9.602937e-01
 99   6.09468975 -1.873813e-01  9.822873e-01
 100  6.18893753 -9.410831e-02  9.955620e-01
 101  6.28318531 -2.449213e-16  1.000000e+00
```

Cuarto Paso

Buscamos las intersecciones entre las funciones y los cortes en el eje y, cuando este vale 0, en las dos funciones

```r
#Ciclo que encuentra las intersecciones entre funciones
xInterseccion <- seq(-pi,pi*2, by = 0.1)
y1Interseccion <- c(f(xInterseccion))
y2Interseccion <- c(g(xInterseccion))
nInsterseccion <- (pi+(pi*2))/0.1
interseccion <- c(0,0,0)
contador <- 0

for(i in 1:nInsterseccion)
{
  if(y1Interseccion[i] == y2Interseccion[i])
  {
    interseccion[contador] <- xInterseccion[i]
    contador <- contador + 1
  }
}
contador <- 0
```

```r
#Ciclo que encuentra los cortes con y=0 de las dos funciones
corteF <- c(0,0)
corteG <- c(0,0,0)
contF  <- 0
contG  <- 0

for(i in 1:nInsterseccion)
{
  if(y1Interseccion[i] == 0)
  {
    corteF[contador] <- xInterseccion[i]
    contF <- contF + 1
  }
  if(y2Interseccion[i] == 0)
  {
    corteG[contador] <- xInterseccion[i]
    contG <- contG + 1
  }
}

#Imprimir el vector con valores x de las particiones
print(xi)
```

```
##   [1] -3.14159265 -3.04734487 -2.95309709 -2.85884931 -2.76460154
##   [6] -2.67035376 -2.57610598 -2.48185820 -2.38761042 -2.29336264
##  [11] -2.19911486 -2.10486708 -2.01061930 -1.91637152 -1.82212374
##  [16] -1.72787596 -1.63362818 -1.53938040 -1.44513262 -1.35088484
##  [21] -1.25663706 -1.16238928 -1.06814150 -0.97389372 -0.87964594
##  [26] -0.78539816 -0.69115038 -0.59690260 -0.50265482 -0.40840704
##  [31] -0.31415927 -0.21991149 -0.12566371 -0.03141593  0.06283185
##  [36]  0.15707963  0.25132741  0.34557519  0.43982297  0.53407075
##  [41]  0.62831853  0.72256631  0.81681409  0.91106187  1.00530965
##  [46]  1.09955743  1.19380521  1.28805299  1.38230077  1.47654855
##  [51]  1.57079633  1.66504411  1.75929189  1.85353967  1.94778745
##  [56]  2.04203522  2.13628300  2.23053078  2.32477856  2.41902634
##  [61]  2.51327412  2.60752190  2.70176968  2.79601746  2.89026524
##  [66]  2.98451302  3.07876080  3.17300858  3.26725636  3.36150414
##  [71]  3.45575192  3.54999970  3.64424748  3.73849526  3.83274304
##  [76]  3.92699082  4.02123860  4.11548638  4.20973416  4.30398194
##  [81]  4.39822972  4.49247749  4.58672527  4.68097305  4.77522083
##  [86]  4.86946861  4.96371639  5.05796417  5.15221195  5.24645973
##  [91]  5.34070751  5.43495529  5.52920307  5.62345085  5.71769863
##  [96]  5.81194641  5.90619419  6.00044197  6.09468975  6.18893753
## [101]  6.28318531
```

Quinto Paso

Finalmente calculamos la integral con los puntos de intersección

```r
contF <- 0
contF <- 0
```

```r
#Valores auxiliares
acumulado <- 0
aux <-0

#Ciclo que halla la integral entre dos funciones
for(i in 1:particiones)
{
  if( !is.null(xi[i] < interseccion[contador] & interseccion[contador] <
xi[i+i]))  ## area cuando no hay interseccion
{
  if (!is.null(xi[i] < corteF[contF] & corteF[contF] < xi[i+1] ) )
## area cuando no hay corte con y = 0 en f(x)
  {
    area1 <- n*((y1[i]+y1[i+1])/2)
  }
  if(!is.null(xi[i] < corteG[contG] & corteG[contG] < xi[i+1] ) )
## area cuando no hay corte con y = 0 en g(x)
  {
    area2 <- n*((y2[i]+y2[i+1])/2)
  }
  if (is.null(xi[i] < corteF[contF] & corteF[contF] < xi[i+1])  )
## area cuando hay corte con y = 0 en f(x)
  {
    v <- x[i+1]-corteF[contF]
    b1 <- n-v
    b2 <- v
    triangulo1 <- (b1*y1[i])/2
    triangulo2 <- (b2*y1[i+1])/2
    area1 <- triangulo1 + triangulo2
    contF <- contF + 1
  }
  if(is.null(xi[i] < corteG[contG] & corteG[contG] < xi[i+1] ) )
## area cuando hay corte con y = 0 en g(x)
  {
    v <- xi[i+1]-corteG[contG]
    b1 <- n-v
    b2 <- v
    triangulo1 <- (b1*y2[i])/2
    triangulo2 <- (b2*y2[i+1])/2
    area1 <- triangulo1 + triangulo2
    contG <- contG + 1
  }
  if(area1*area2 < 0)
  {
    print("opuestos")
    aux <- area1+area2
    print(aux)
  }
```

```
  if(area1*area2 > 0)
  {
    if(y1[i] > 0 )
    {
      ifelse(area1 > area2, aux <- area1-area2, aux <- area2-area1)
    }
    if(y1[i] < 0)
    {
      ifelse(area1 < area2, aux <- area1-area2, aux <- area2-area1)
    }
  }
}

if( is.null(xi[i] < interseccion[contador] & interseccion[contador] <
xi[i+i]))   ## area cuando hay interseccion de curvas
{
  print("inter")
  v <- xi[i+1]-interseccion[contador]
  h1 <- n-v
  h2 <- v
  trapecioF1 <- h1*((y1[i]+f(interseccion[contador]))/2)
  trapecioF2 <- h2*((y1[i]+f(interseccion[contador]))/2)
  trapecioG1 <- h1*((y2[i]+f(interseccion[contador]))/2)
  trapecioG2 <- h2*((y2[i]+f(interseccion[contador]))/2)

  trapecioF <- trapecioF1 + trapecioF2
  trapecioG <- trapecioG1 + trapecioG2
  aux <- trapecioF + trapecioG
}

  ifelse(-0.0001 < acumulado & acumulado < 0.0001, acumulado <- 0 ,
acumulado <- acumulado + aux )


print("area1")
print(area1)
print("area2")
print(area2)
print("aux")
print(aux)
print("acumulado")
print(acumulado)
print("-----")
}

## [1] "area1"
## [1] -0.00443475
## [1] "area2"
## [1] -0.09403864
## [1] "aux"
```

```
## [1] -0.08960389
## [1] "acumulado"
## [1] 0
## [1] "-----"
## [1] "area1"
## [1] -0.01326489
## [1] "area2"
## [1] -0.09320395
## [1] "aux"
## [1] -0.07993906
## [1] "acumulado"
## [1] 0
## [1] "-----"
## [1] "area1"
## [1] -0.02197728
## [1] "area2"
## [1] -0.09154197
## [1] "aux"
## [1] -0.06956469
## [1] "acumulado"
## [1] 0
## [1] "-----"
## [1] "area2"
## [1] -0.08177571
## [1] "aux"
## [1] -0.03513167
## [1] "acumulado"
## [1] 0

## [1] "-----"
print("Total: ")

## [1] "Total: "

print(acumulado)

## [1] 0
```

Gráfica

Gráfica con la integral calculada, en base a las áreas bajo la curva.

```
plot(f, -pi, pi*2, lwd = 3, col = "red")
lines(x, g(x),lwd = 3, col = "blue")
vectorx <- c(-pi, pi*2)
vectory <- c(0,0)
lines(vectorx,vectory)

for(i in 1:particiones)
{
  ax <- c(xi[i],xi[i])
  ay1 <- c(0,y1[i])
  ay2 <- c(0,y2[i])
  bx <- c(xi[i+1],xi[i+1])
  by1 <- c(0,y1[i+1])
  by2 <- c(0,y2[i+1])
  hx <- c(xi[i],xi[i+1])
  hy1 <- c(y1[i],y1[i+1])
  hy2 <- c(y2[i],y2[i+1])
  lines(ax,ay1,lwd = 2, col="red")
  lines(bx,by1, col="red")
  lines(hx,hy1, col="red")
  lines(ax,ay2, col="blue")
  lines(bx,by2, col="blue")
  lines(hx,hy2, col="blue")
}
```