Método del trapecio compuesto

Guillermo Palomo Fernández

100430523

**Índice**

Introducción…. 3

Algoritmo y código…. 3

Chequeo…. 4

Resultados…. 5

Conclusiones…. 8

Anexo código de gráficos y tablas(R)…. 9

**Introducción**

La regla del trapecio compuesto es un método numérico para aproximar integrales definidas usando las áreas de un número determinado n de trapecios de misma base. En esta práctica se expondrá el algoritmo y código usado para llevar a cabo el método junto a los resultados obtenidos.

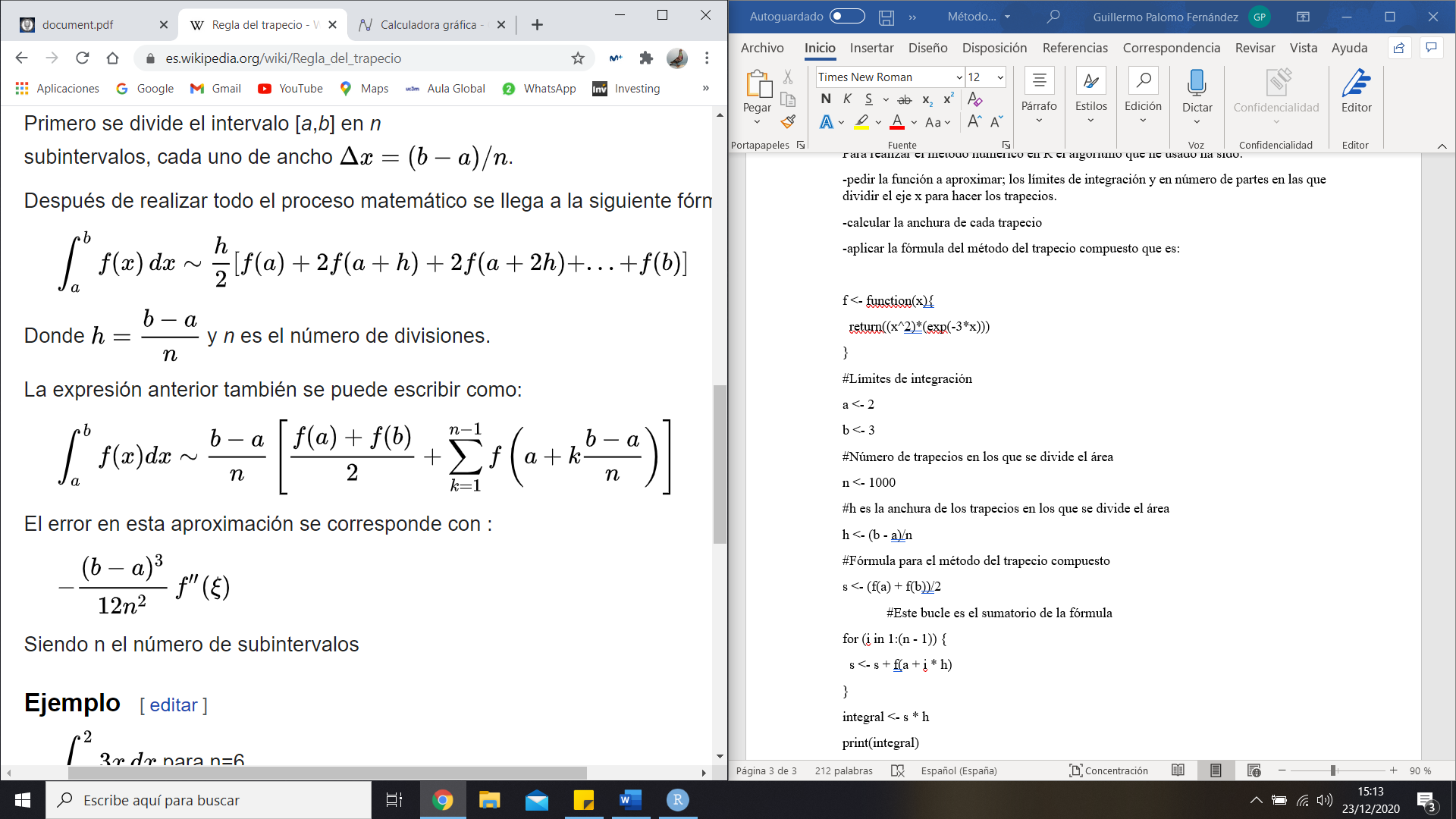
**Algoritmo y código**

Para realizar el método numérico en R el algoritmo que he usado ha sido:

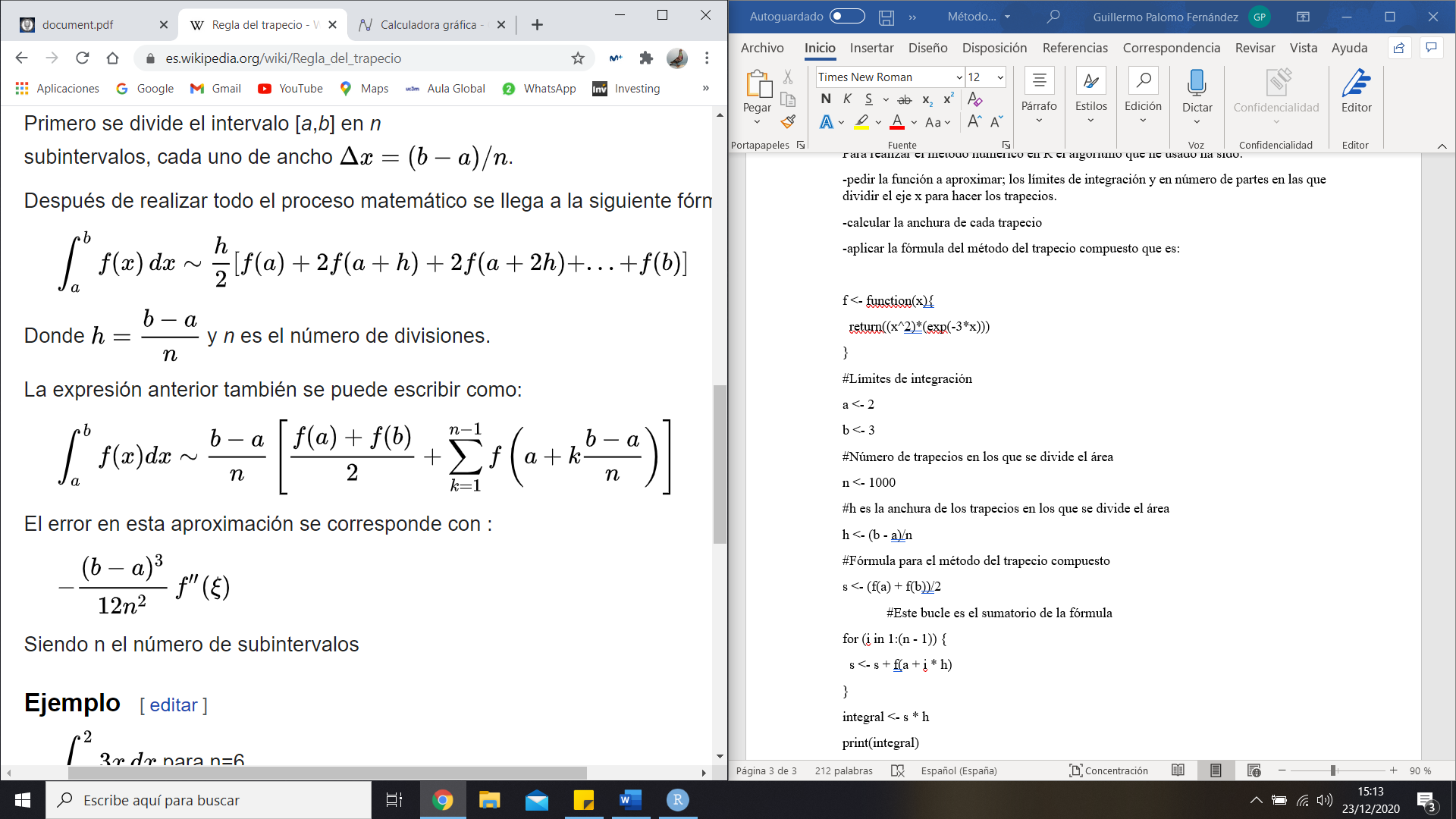
-pedir la función a aproximar; los límites de integración y el número de partes en las que dividir el eje x para hacer los trapecios.

-calcular la anchura de cada trapecio

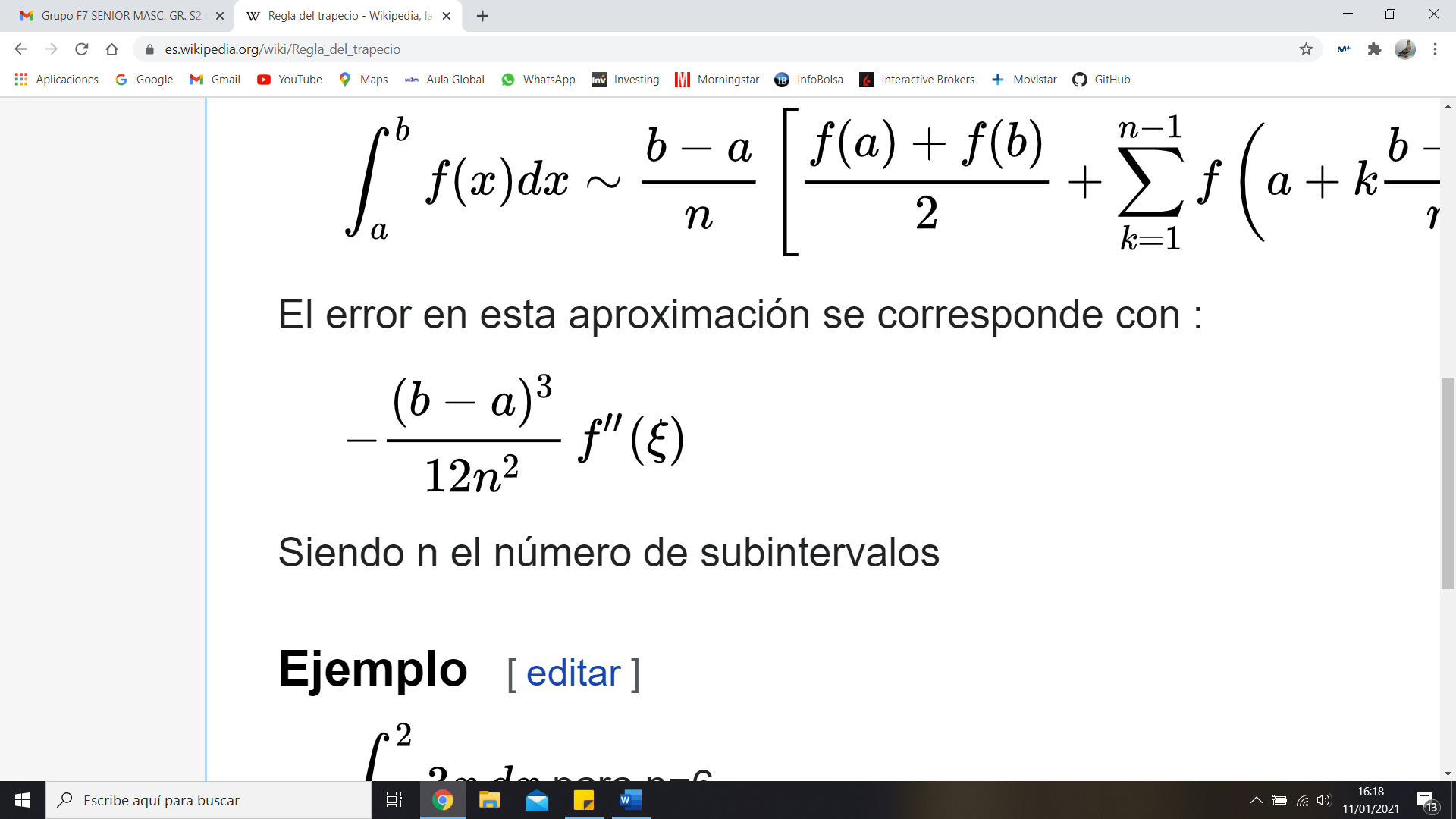
-aplicar la fórmula del método del trapecio compuesto que es:



con



-calcular el error con la fórmula:



Código

f <- function(x){

return(x)

}

#Límites de integración

a <- 1

b <- 3

#Número de trapecios en los que se divide el área

n <- 10

#h es la anchura de los trapecios en los que se divide el área

h <- (b - a)/n

#Fórmula para el método del trapecio compuesto

s <- (f(a) + f(b))/2

#Este bucle es el sumatorio de la fórmula

for (i in 1:(n - 1)) {

s <- s + f(a + i \* h)

}

integral <- s \* h

print(integral)

#Error

##Funcion para la segunda derivada

DD<-function(expr,name,order=1){

if(order<1) stop("Order must be >=1")

if(order==1)D(expr,name)

else DD(D(expr,name),name,order-1)

}

expe <- expression((x^2)\*(exp(-3\*x)))

DD(expe,"x",2)

##Segunda derivada de nuestra función evaluada en 2

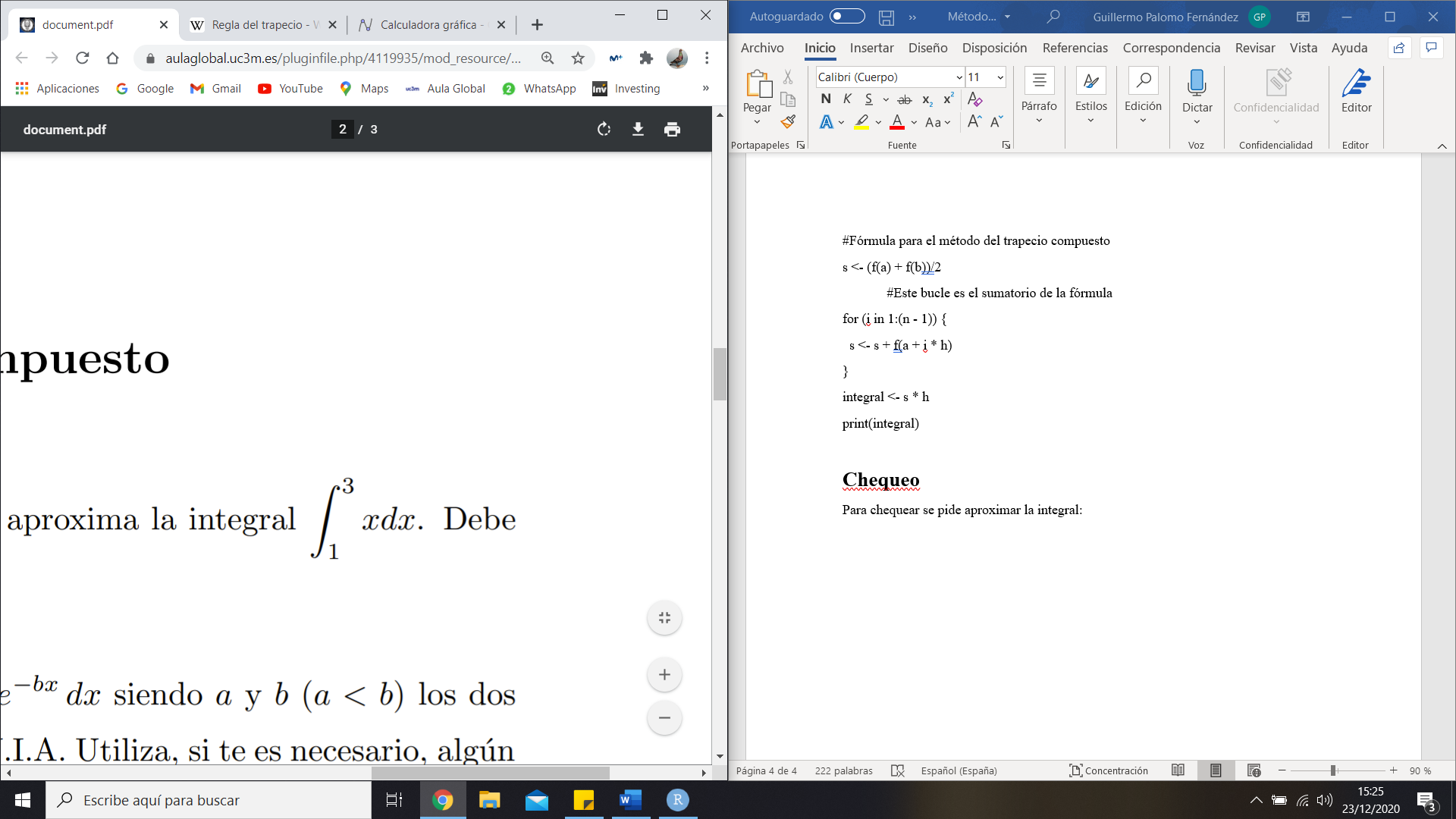
dff <- 2 \* (exp(-3 \* 2)) - 2 \* 2 \* (exp(-3 \* 2) \* 3) - (2 \* 2 \* (exp(-3 \* 2) \* 3) - (2^2) \* (exp(-3 \* 2) \* 3 \* 3))

error <- (((b - a)^3)/((12\*n)^2))\* dff

error

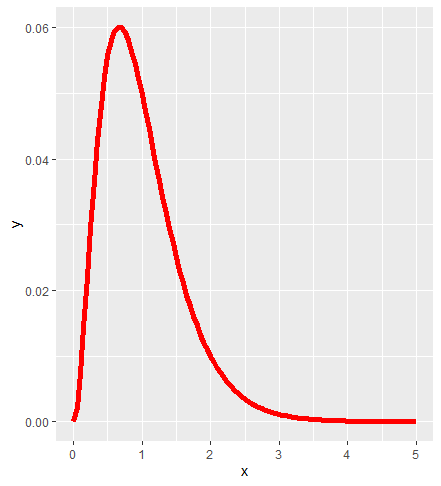
**Chequeo**

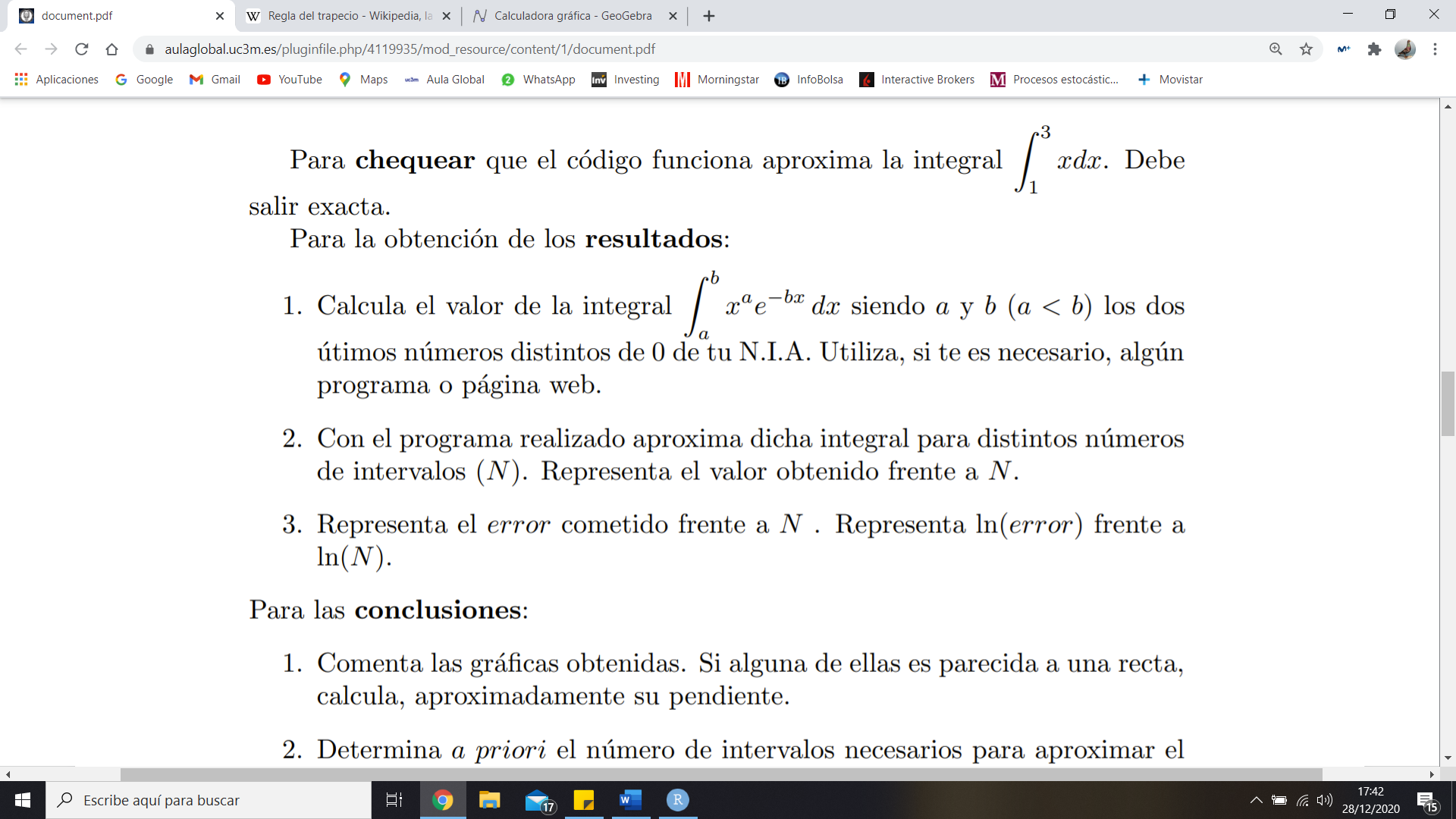
Para chequear se pide aproximar la integral:



con el código anteriormente mostrado la integral se aproxima a 4, saliendo exacta sin importar la n.

**Resultados**

Se pide calcular:

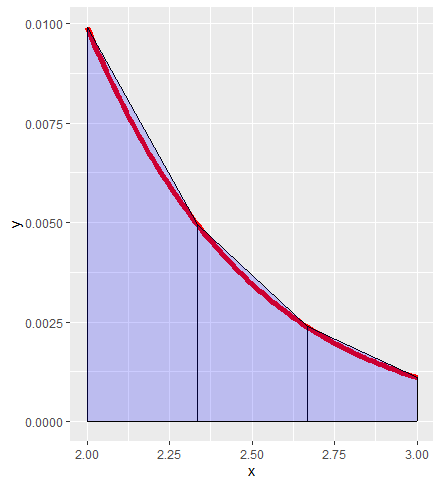


Con a = 2 y b = 3.

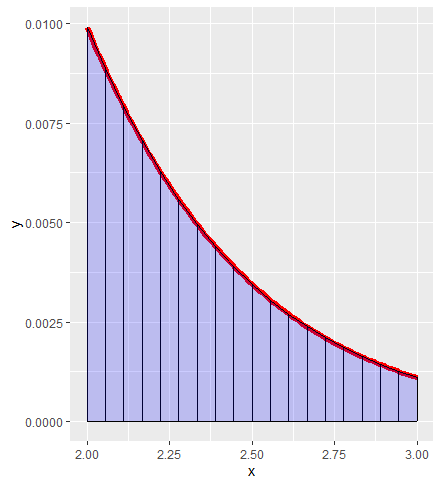
Para *N* cada vez más grandes se aproxima más al valor real de la integral como se ve en la siguiente tabla:

|  |  |
| --- | --- |
| N | Integral |
| 1 | 0.016538545 |
| 2 | 0.004484813 |
| 3 | 0.004287684 |
| 4 | 0.004218243 |
| 5 | 0.004186026 |
| 6 | 0.004168507 |
| 7 | 0.004157936 |
| 8 | 0.004151073 |
| 9 | 0.004146366 |
| 10 | 0.004142999 |
| 11 | 0.004140507 |
| 12 | 0.004138611 |
| 13 | 0.004137136 |
| 14 | 0.004135966 |
| 15 | 0.004135021 |
| 16 | 0.004134249 |
| 17 | 0.004133608 |
| 18 | 0.004133071 |
| 19 | 0.004132617 |
| 20 | 0.004132229 |

Para N = 3:



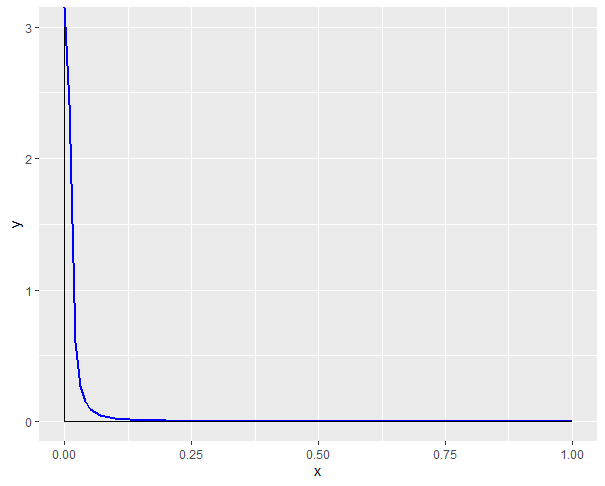
Para N = 18:

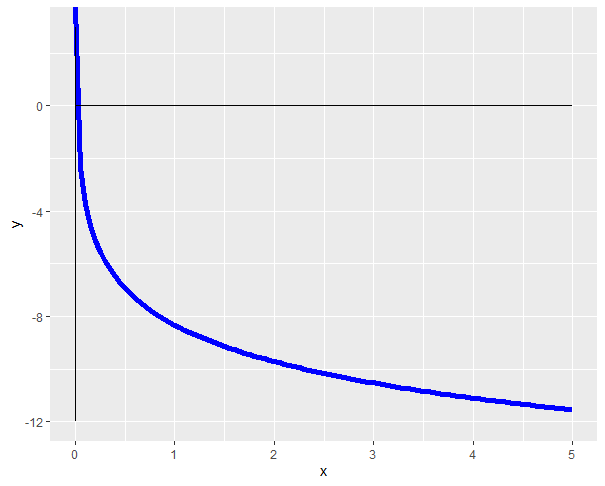


Para representar el valor del error frente a *N* utilizaré dos tablas usando en una de ellas el ln(error) frente al ln(N) y en otra los valores con la fórmula anterior.

|  |  |
| --- | --- |
| N | Error |
| 1 | 2.409898e-04 |
| 2 | 6.024745e-05 |
| 3 | 2.677664e-05 |
| 4 | 1.506186e-05 |
| 5 | 9.639592e-06 |
| 6 | 6.694161e-06 |
| 7 | 4.918159e-06 |
| 8 | 3.765466e-06 |
| 9 | 2.975183e-06 |
| 10 | 2.409898e-06 |

|  |  |
| --- | --- |
| Ln(N) | Ln(error) |
| 0 | -8.330756 |
| 0.6931472 | -9.717050 |
| 1.0986123 | -10.527981 |
| 1.3862944 | -11.103345 |
| 1.6094379 | -11.549632 |
| 1.7917595 | -11.914275 |
| 1.9459101 | -12.222576 |
| 2.0794415 | -12.489639 |
| 2.1972246 | -12.725205 |
| 2.3025851 | -12.935926 |



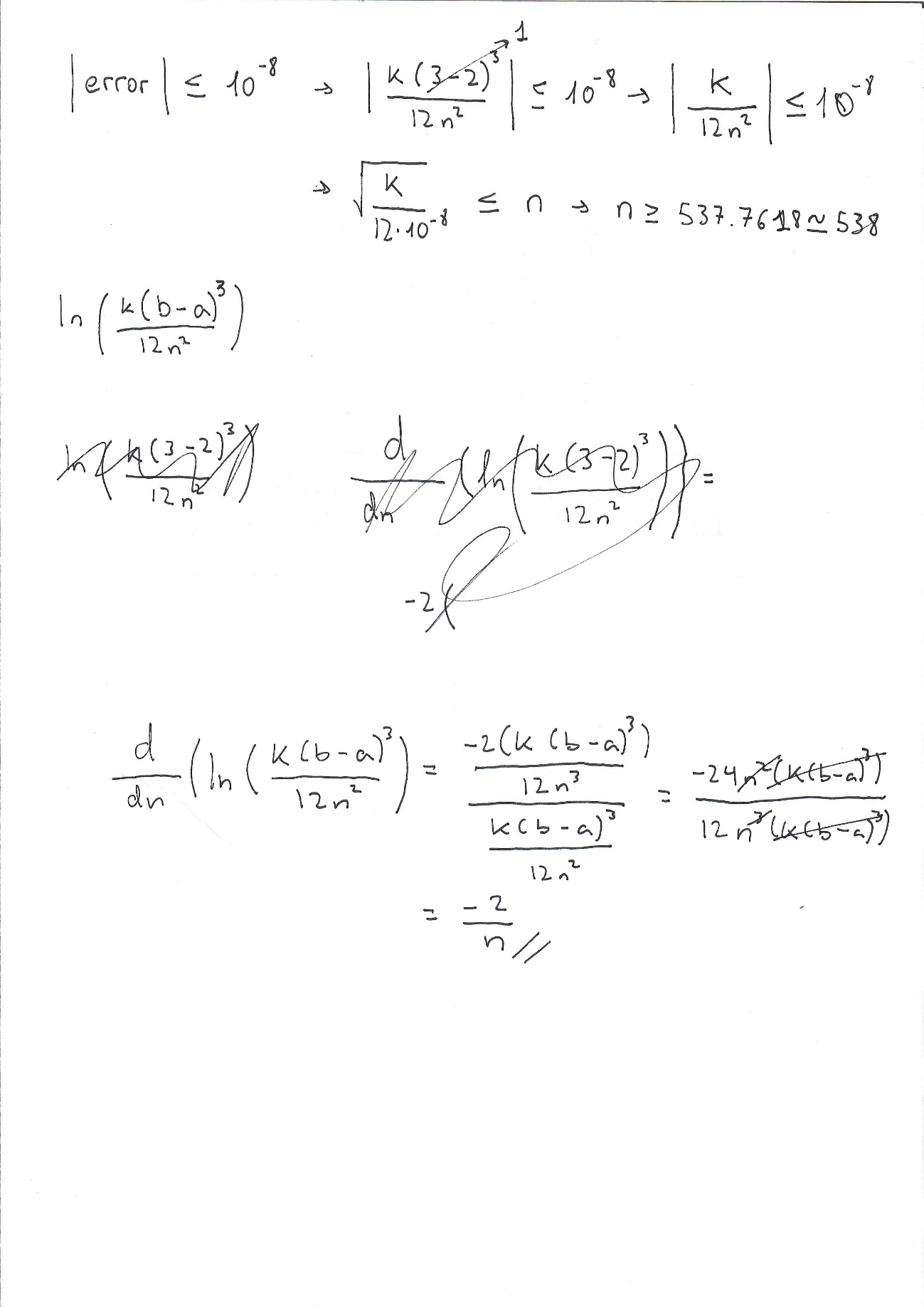


**Conclusiones**

1. Comenta las gráficas obtenidas. Si alguna de ellas es parecida a una recta, calcula, aproximadamente su pendiente.

En las gráficas de las integrales se puede ver que, a mayores *N*, menor es el error y más se aproxima al verdadero valor de la integral.

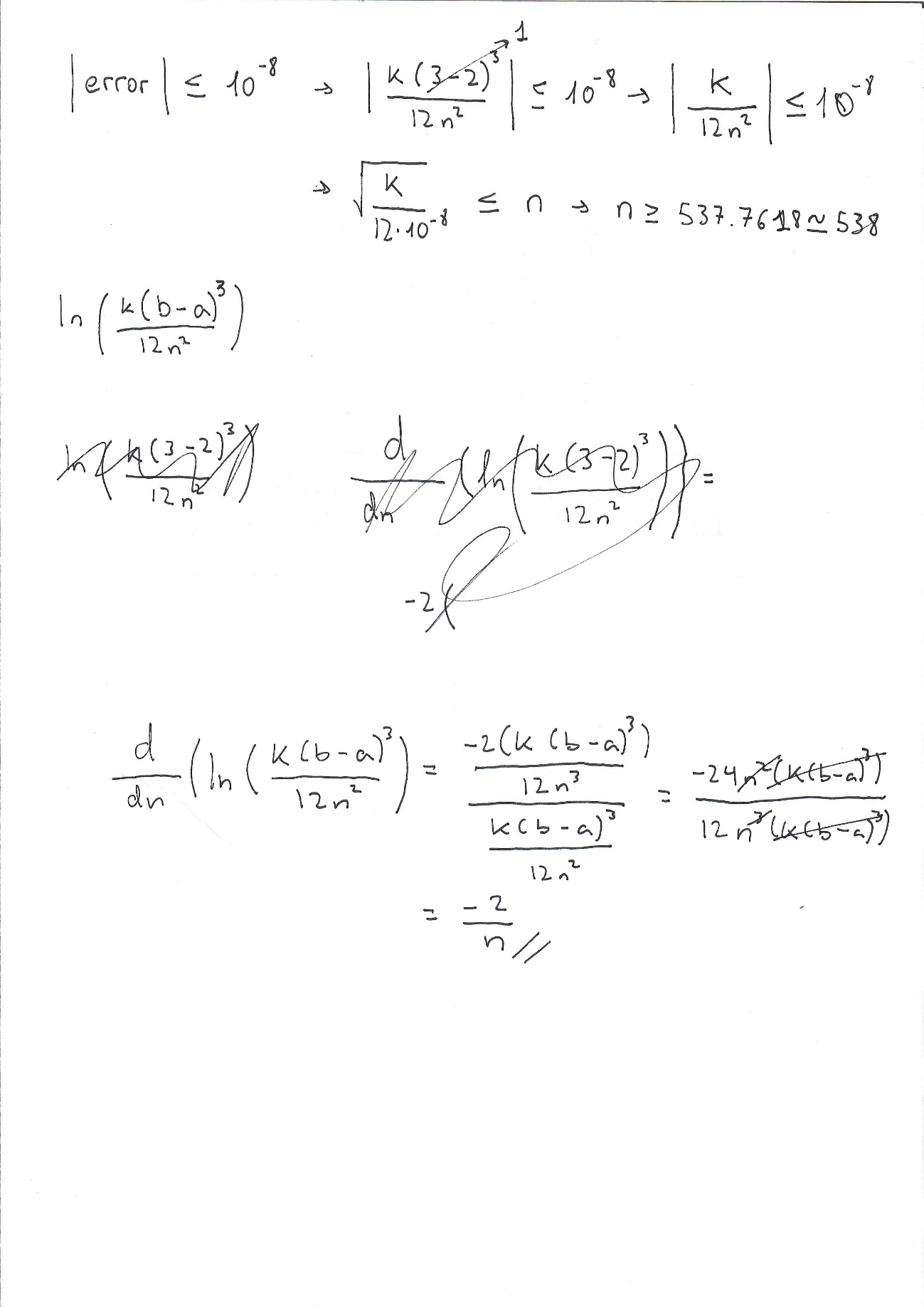
La gráfica más similar a una recta es el ln(error) a partir de alrededor del 2. La pendiente es la derivada de la función del error que es:



1. Determina *a priori* el número de intervalos necesarios para aproximar el valor de la integral con un error menos que 108. Justifica tu respuesta y chequéalo si puedes.

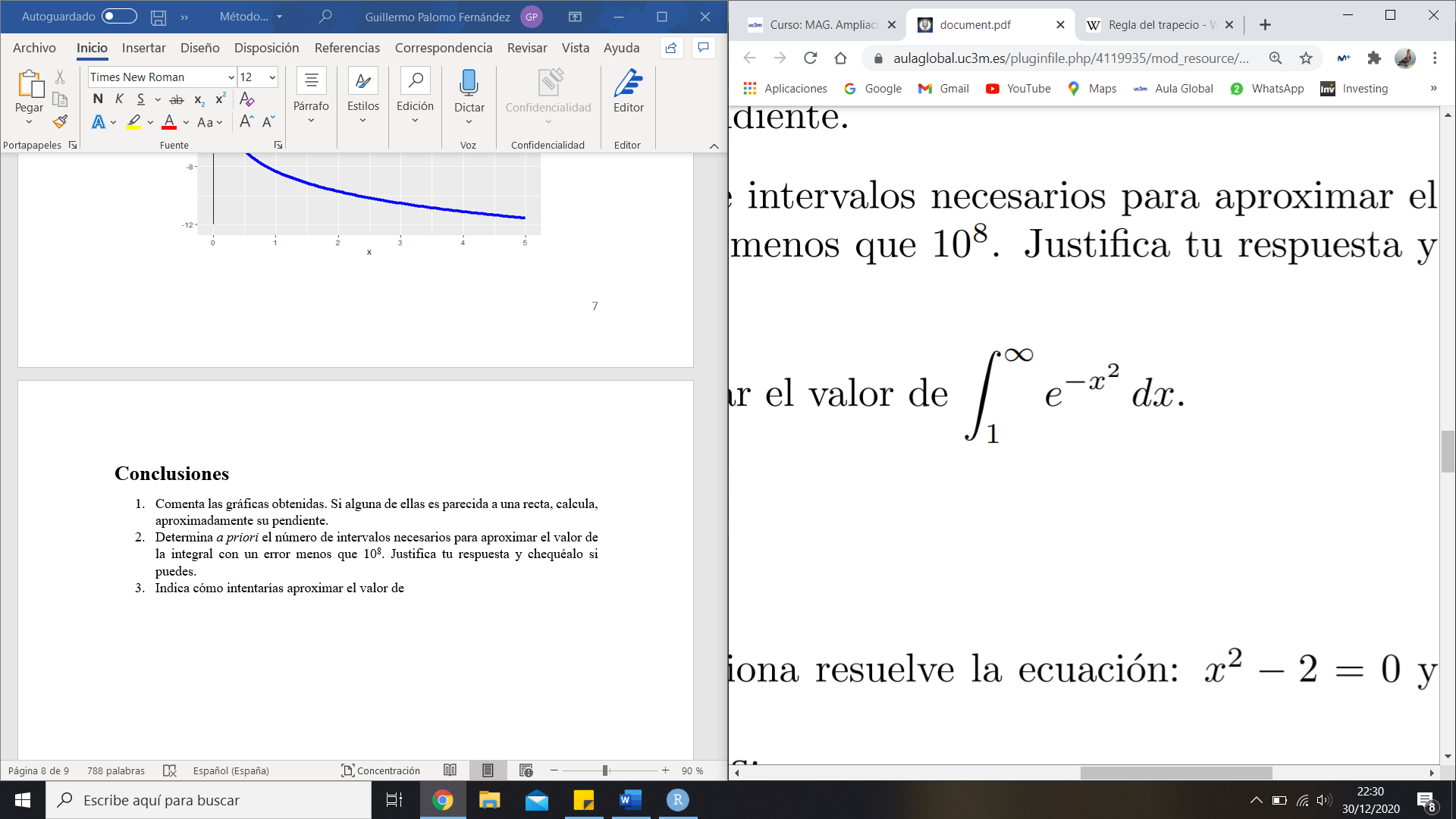
Supongo que se pide un error menor que 0.00000001 en el enunciado. Para ello utilizo la fórmula del error sacando primero la constante *k(dff en código)* para el menor valor del intervalo en el que estudiamos la función, en este caso 2. Para este caso

*k* *=* *0.034702530473339*



Para n = *538*, el resultado de la integral es: *0.004128643*

1. Indica cómo intentarías aproximar el valor de:



NS/NC

**Anexo código de gráficos y tablas(R)**

graph\_integral <- ggplot(data.frame(x = c(a,b)) , mapping = aes(x = x)) +

stat\_function(fun = f, color = "red", lwd = 2) +

geom\_segment(aes(x = b, y = f(b), xend = b, yend = 0)) +

geom\_segment(aes(x = a, y = f(a), xend = a, yend = 0)) +

geom\_segment(aes(x =a, y = 0, xend = b, yend = 0))

h2 <- 0

h3 <- h

for (j in 1:(n)) {

graph\_integral <- graph\_integral +

geom\_segment(aes\_(x = a + h2, y = f(a + h2), xend = a + h3 , yend = f( a + h3))) +

geom\_segment(aes\_(x = a + h2, y = f(a + h2), xend = a + h2 , yend = 0)) +

geom\_polygon(data = data.frame(cbind(c(a + h2, a + h3, a + h3, a + h2), c(f(a + h2), f(a + h3), 0, 0))), aes(x=X1, y=X2), fill = 'blue', alpha = 0.2)

h2 <- h2 + (b - a)/n

h3 <- h3 + (b - a)/n

}

graph\_integral

----------

m <- matrix(nrow = n, ncol = 2)

for (z in 1:n) {

h <- (b - a)/z

s <- (f(a) + f(b))/2

for (i in 1:(z - 1)) {

s <- s + f(a + i \* h)

}

integral <- s \* h

m[z,1] <- z

m[z,2] <- integral

}

m

-----------

graph\_integrales <- ggplot(data.frame(x = c(0,b + 2)) , mapping = aes(x = x)) +

stat\_function(fun = f, color = "red", lwd = 2)

graph\_integrales

----------

graph\_logn <- ggplot(data.frame(x = c(0,5)) , mapping = aes(x = x)) +

stat\_function(fun = function(x){log(x,exp(1))}, color = "blue", lwd = 2) +

geom\_segment(aes(x = 0, y = 0, xend = 5, yend = 0)) +

geom\_segment(aes(x = 0, y = -3, xend = 0, yend = 3))

graph\_logn

graph\_logerror <- ggplot(data.frame(x = c(0,5)) , mapping = aes(x = x)) +

stat\_function(fun = function(x){log((((b - a)^3)/((12\*x)^2))\* dff,exp(1))}, color = "blue", lwd = 2) +

geom\_segment(aes(x = 0, y = 0, xend = 5, yend = 0)) +

geom\_segment(aes(x = 0, y = -12, xend = 0, yend = 3))

graph\_logerror

graph\_error <- ggplot(data.frame(x = c(0,1)) , mapping = aes(x = x)) +

stat\_function(fun = function(x){(((b - a)^3)/((12\*x)^2))\* dff}, color = "blue", lwd = 1) +

geom\_segment(aes(x = 0, y = 0, xend = 1, yend = 0)) +

geom\_segment(aes(x = 0, y = 0, xend = 0, yend = 3))

graph\_error

---------

mx <- matrix(nrow = n, ncol = 2)

for (z in 1:n) {

e <- abs((((b - a)^3)/((12\*z)^2))\* dff

mx[z,1] <- z

mx[z,2] <- e

}

mx

---------

mm <- matrix(nrow = n, ncol = 2)

for (z in 1:n) {

e <- abs((((b - a)^3)/((12\*z)^2))\* dff)

mm[z,1] <- log(z, exp(1))

mm[z,2] <- log(e, exp(1))

}

mm