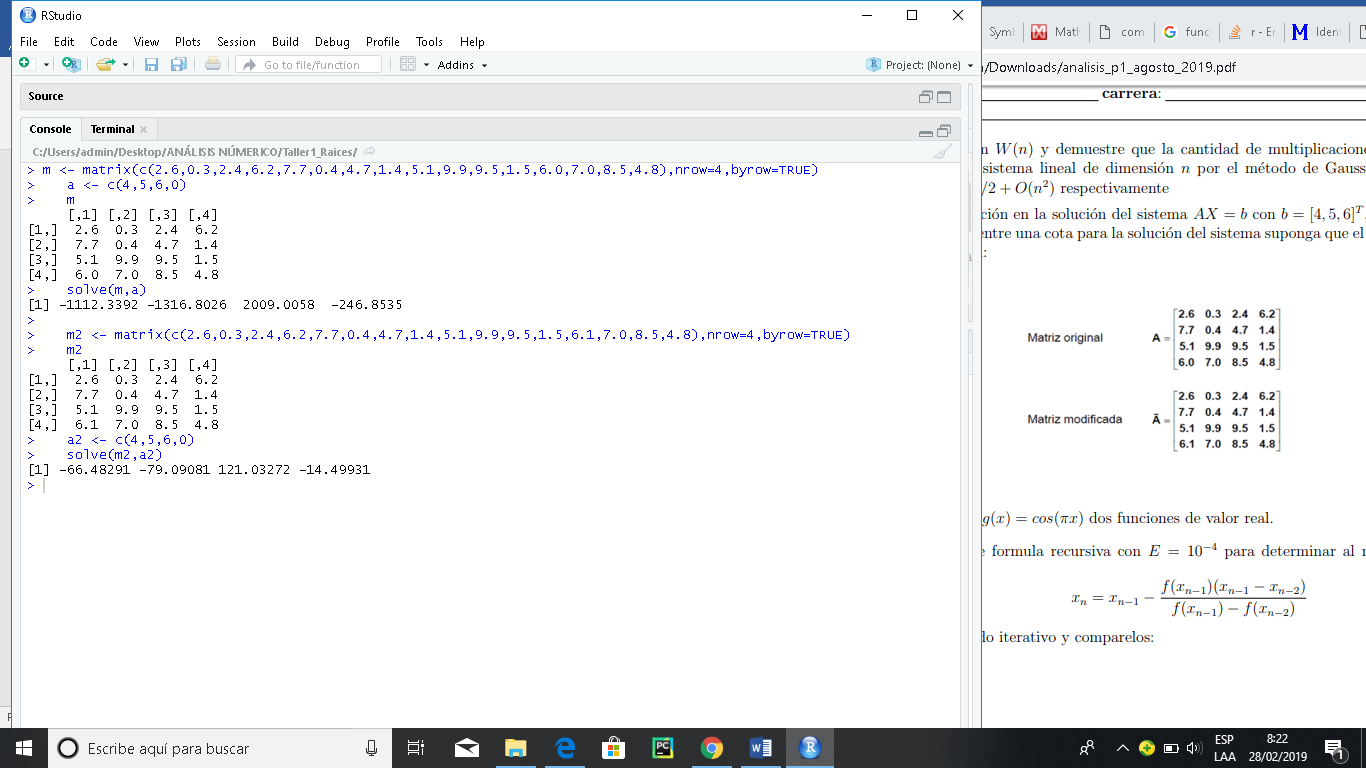
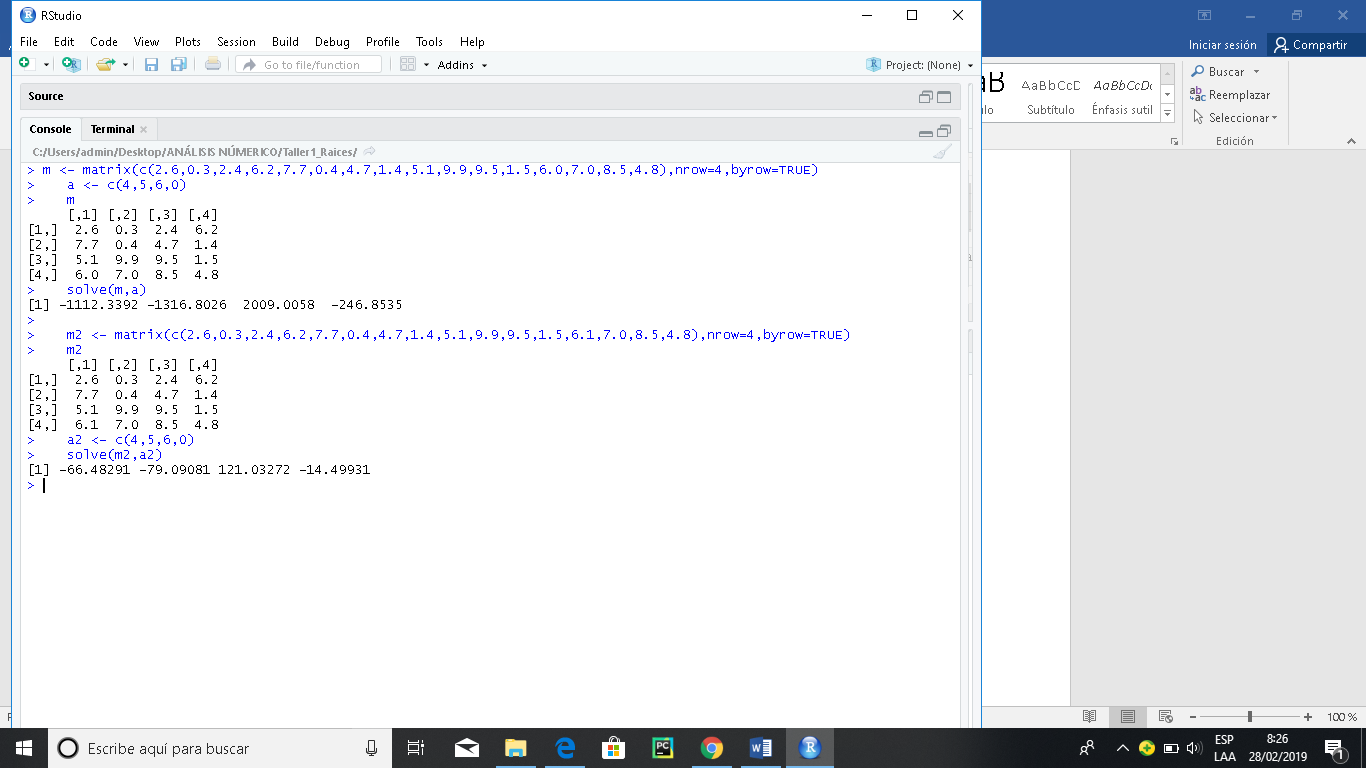
**Parcial 1**

1. **d**
2. **d**

**Resultado Matriz Original:**



**Resultado Matriz Modificada:**



Para poder establecer la cota para la solución del sistema se debe tener en cuenta el número de condición de una matriz para poder cuantificar su nivel de mal condicionamiento definida como:

Sea **AX = B** un sistema de ecuaciones lineales, entonces

**|| A ||** = la norma de la matriz A la cual se necesita para poder el obtener el mayor valor de la suma de las magnitudes de los componentes de cada fila.

**A-1** = La matriz inversa de A

**cond(A) =** || A || \* || A-1 || >= || A\*A-1 || = || I || = 1entonces

**cond(A)** >= 1

Esto quiere decir que un sistema no está mal condicionado o tiene una cota muy alta de error respecto al vector solución y a una variación si supera es menor a uno.

Cabe anotar que, si la matriz tiene filas bastante dependientes de otras filas, su determinante tomara valores muy pequeños y su inversa valores muy grandes, siendo este un inicio de que la matriz está mal condicionada y que una variación mínima en cualquier valor generara errores de cálculo o un cambio bastante significativo respecto al vector solución.

Se probará para la matriz original la cota:

**cond(A) = || A || || A -1 || =** 17879.09

Este resultado indica que se produce un cambio muy significativo en el vector solución si existe una ligera perturbación en la matriz de coeficientes.

También se puede establecer una cota para la sensibilidad o error poniendo un rango de variación en decimales o unidades, indicando igualmente el número mínimo de coeficientes a modificar, y calcular su solución y a partir de esto indicar el nivel de sensibilidad la formula seria:

**cota =**  matriz original + matriz variada / matriz original

1. **Código del Algoritmo de la formula recursiva descrita:**

rm(list=ls())

Fx <- function(t) tan(t\*pi) - cos(t\*pi)

Hx <- function(t,t2) t - Fx(t)\*(t - t2)/Fx(t) - Fx(t2)

metodo2 <- function(x1,x2)

{

t<-x1

t2<-x2

r<-Hx(t,t2)

t2<-t

i<-0

while (Fx(r) != 0 )

{

error<-abs(Fx(t)/Hx(t,t2))

if(error > 1.e-4)

{

t2 <- t

t <- r

}

else {break}

r<-Hx(t,t2)

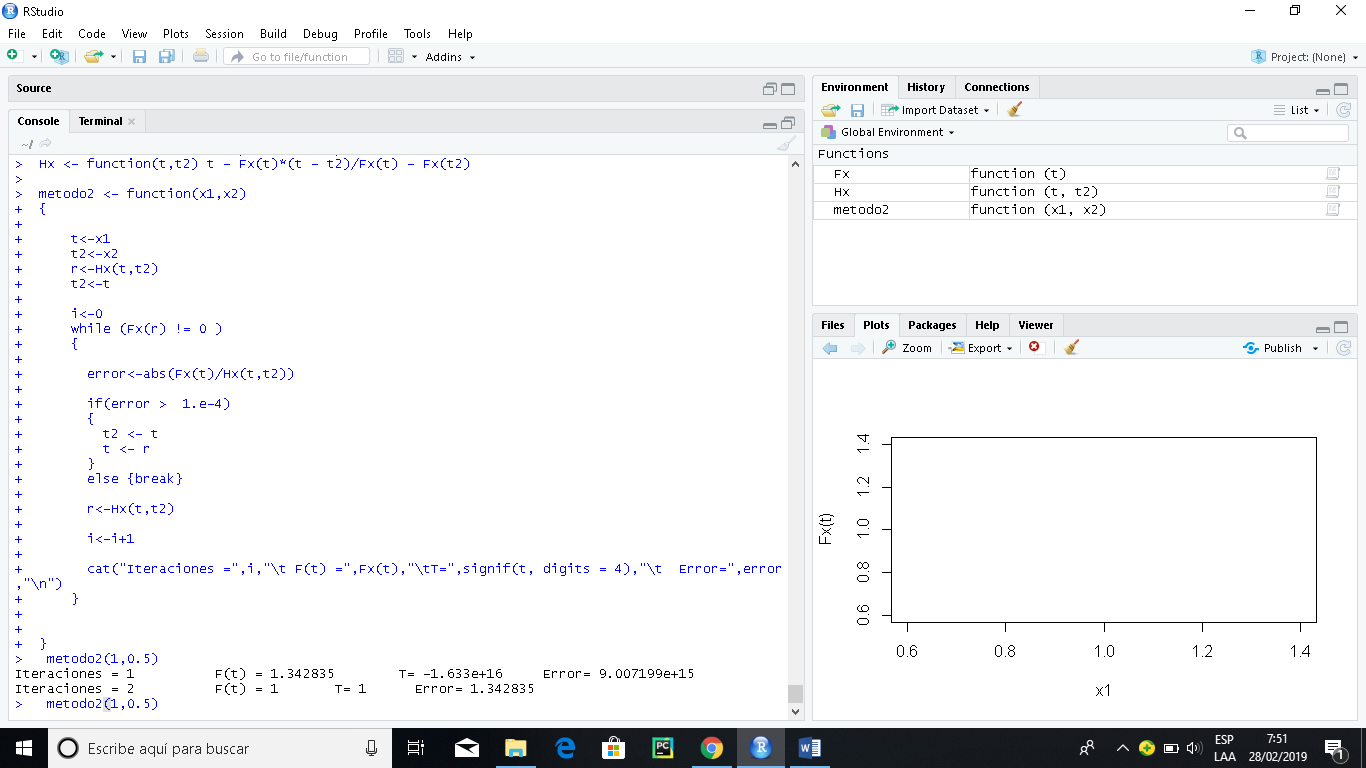
i<-i+1

cat("Iteraciones =",i,"\t F(t) =",Fx(t),"\tT=",signif(t, digits = 4),"\t Error=",error,"\n")

}

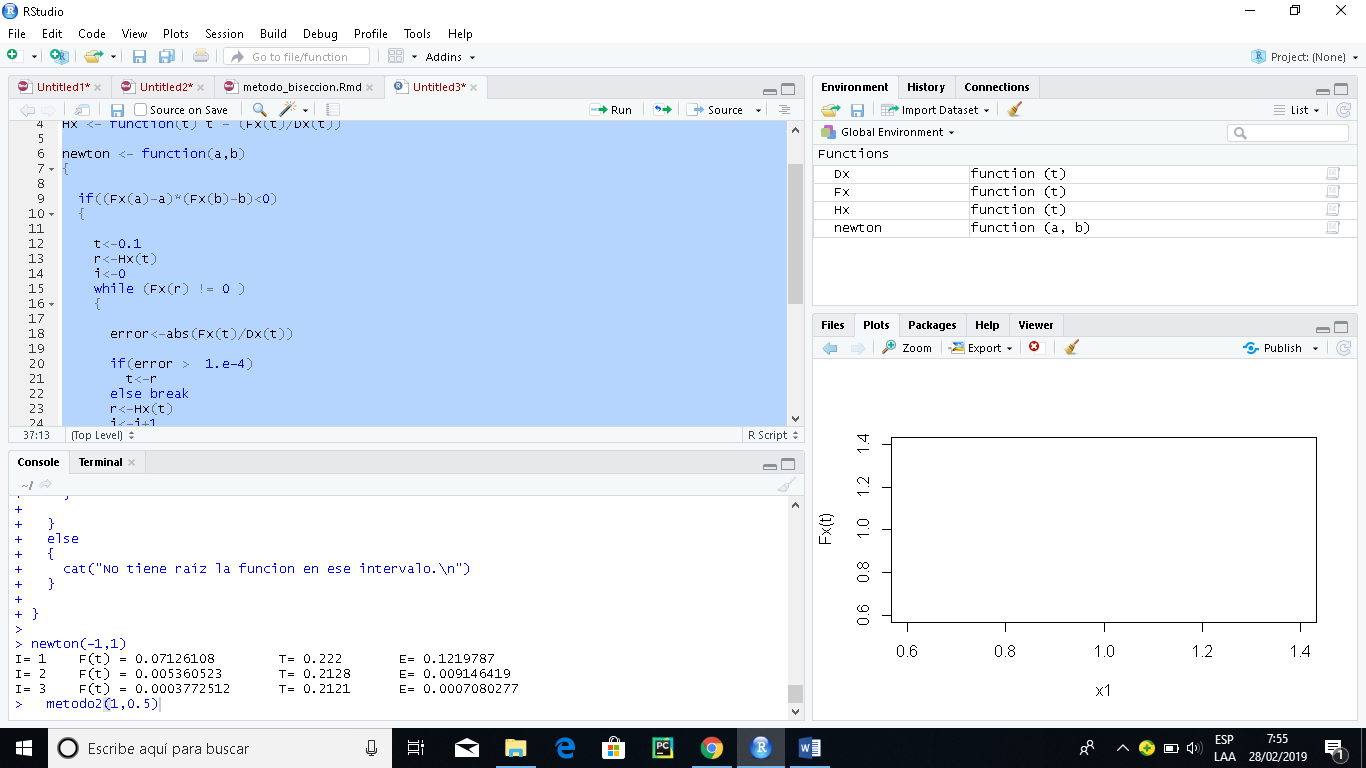
}

metodo2(1,0.5)

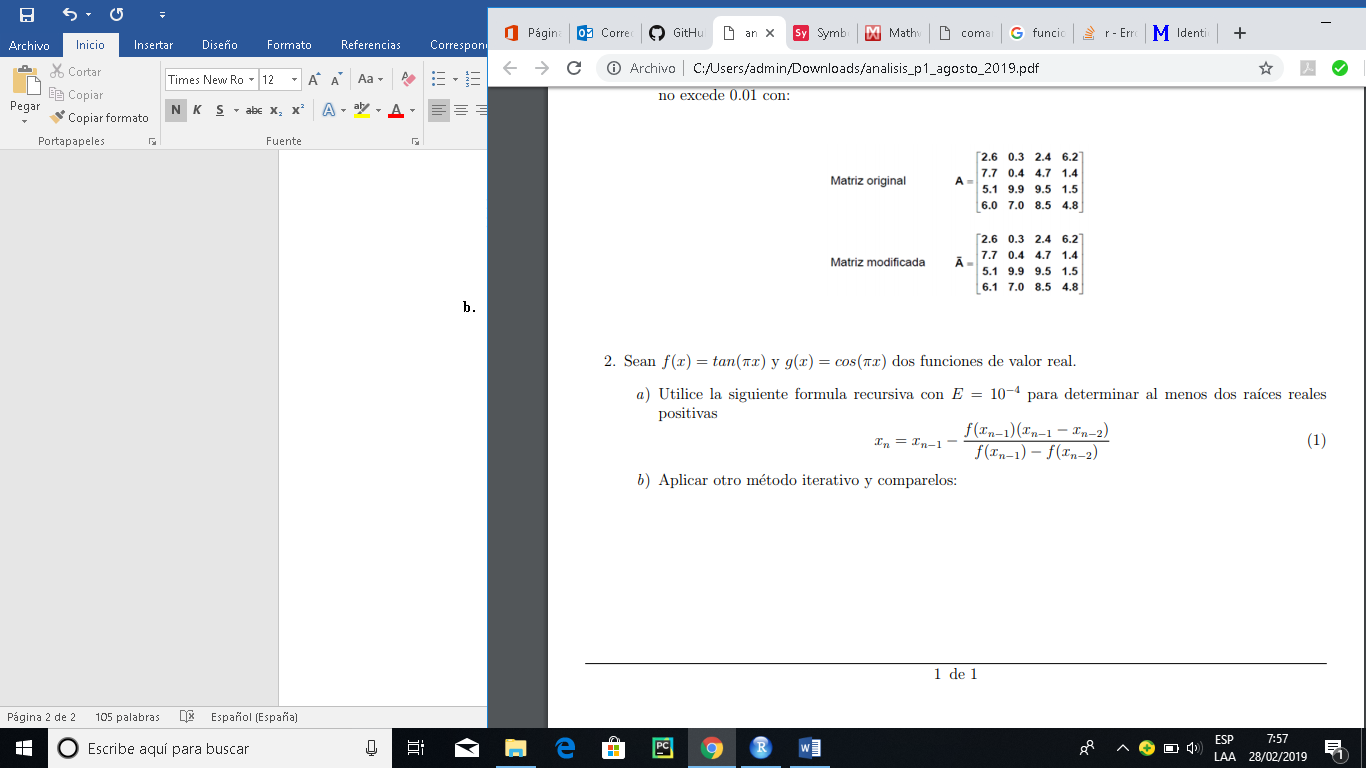
**Tabla**

1. Use el algoritmo de Newton para hallar el punto donde intersectaban las dos funciones mis resultados fueron los siguientes:

**Tabla:**



Se puede concluir que el método de la formula recursiva:



Es más eficiente que el método de Newton ya que es posible observar en las tablas de resultados que realiza menos iteraciones es decir en cada cálculo se aproxima más rápidamente a la a la respuesta a pesar que ambos convergen cuadráticamente dada la definición de su fórmula.