Act5

Carlos David Ureña Pérez

December 2019

1 Introducción

En esta actividad daremos un salto de dificultad. Ahora trabajaremos con funciones o subrutinas. En este caso elegí trabajar con funciones ya que a mi parecer son menos complicadas de manejar y ya tenia conocimiento previo sobre esto. La actividad trata de replicar la grafica de wikipedia del seno utilizando la serie de tylor, para despues hacerla con el ln(1+x)

2 Programación Seno:

Para empezar con el programa primero, analicé el programa de ejemplo brindado con el profesor, intenté modificarlo con el sin, sin grandes cambios y como resultado me daba algo cercano a la respuesta, pero no lo suficientemente cerca para concluir con el programa, por lo que investigué más profundamente sobre el sin mediante serie de tylor y encontré que mediante el uso de sumatorias, todo se volvia más sencillo// El programa queda de la sig manera:

```
 implicit none

    real (kind=8) :: x, y

    real (kind=8), external :: sintaylor

    integer :: n, i,k,pares,a

!Abrimos un un archivo de texto para apuntar los valores de x en la función

    OPEN(unit=2,FILE="senodetaylor.dat")

                DO k=1,6, 1

                a=a+1

                pares=mod(k,2)
```

```
                                IF(k==1)THEN
                    n=1
                                ELSE
                                  n=n+2
                                END IF
                DO i=-100, 100
                  x=0.2*i
                  y=sintaylor(x,n)
                  IF(pares==1.AND.k==1)THEN
                  y=x
                  ELSE iF(pares==1.AND.k>1)THEN
                   y=(-1)*y
                  END IF
                  write(2,*) x,y,a
                END DO
          write(2,*)
    END DO
close(2)
END PROGRAM
```

Para continuar con la funcion se escribió lo antes mencionado de sumatorias:

```
    function sintaylor(x,n)
!=========================
    implicit none
```

```
! function arguments:
real (kind=8), intent(in) :: x
integer, intent(in) :: n
real (kind=8) :: sintaylor

! local variables:
real (kind=8) :: term, partial_sum,a,c,b,d
integer :: j

partial_sum = 0
!DO para comenzar la suma desde grado 0 hasta n
DO j=0,n,1
 a=(-1.0)**j
 b=2*j+1
            d=b
 c=x**b
  DO
              d=d-1
              b=b*d
              IF(d==1)EXIT
              IF(d==0)THEN
              b=1
```

```
                EXIT

                END IF

     END DO

    term=a/b*c

    partial_sum=partial_sum+term

    END DO

   sintaylor=partial_sum


end function sintaylor
```

Al terminar y graficar en gnuplot nos dará estos datos si usamos el zoom correctamente:

3 Programación ln(1+x)

```
    PROGRAM         logaritmo

IMPLICIT NONE

REAL,external::ln

INTEGER::i,g,n

REAL::x,y


OPEN(unit=1,file="ln.dat",access="Append")

DO i=1,4,1

                IF(i==1)THEN

                g=4

                else if(i==2)then
```

```
                g=7
                else if(i==3)then
                g=11
                else if(i==4)then
                g=16
                END IF

                DO n=-1000,1000,1
          x=0.01*n
          y=ln(x,g)
           WRITE(1,*) x,y
    END DO
        WRITE(1,*) ' '
    END DO
   CLOSE (1)
END PROGRAM



FUNCTION ln(x,g)
IMPLICIT NONE
!
REAL,intent(in)::x
INTEGER,intent(in)::g
```

```
REAL::ln

REAL::termino,suma,a,b,c

INTEGER:: m


suma=0

DO m=1,g,1

     a=(-1.0)**(m+1)

     b=x**m

     c=m


      termino=a*(b/c)

       termino=suma+termino

   END DO


    Ln=suma


END FUNCTION Ln
```

4 Conclusión

Creo que fue bastante sencillo programar este programa a pesar de todos los problemas del inicio, lo más complicado fue entender la sumatoria y pasarla al lenguaje Fortran en forma de function. Sin embargo creo que una de las lineas no salió bien y no pude identificar el por qué de este error
La parte del ln fue muy parecida al del seno, solo que como pedía menos cosas fue menos complicado ya que se logro obtener el primer programa.