Polinomio de Taylor

Glenda Carranco

2019

1 Polinomio de Taylor

En cálculo, el teorema de Taylor, recibe su nombre del matemático británico, Brook Taylor, quien lo enunció con mayor generalidad en 1712, aunque previamente James Gregory lo había descubierto en 1671.1 Este teorema permite obtener aproximaciones polinómicas de una función en un entorno de cierto punto en que la función sea diferenciable. Además el teorema permite acotar el error obtenido mediante dicha estimación.

El polinomio que aparece en el teorema de Taylor se denomina polinomio de Taylor de orden k.

$$P_k(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(k)}(a)}{k!}(x-a)^k \qquad (1)$$

En la actividad se nos pedia:

Por favor construye un programa en Fortran para calcular y graficar con Gnuplot:

La aproximación de la función Sin(x), de orden:1,3,5,7,9 y 11.

La aproximación de la función ln(1+x), tal como aparece en la tercera figura del artículo de la Wikipedia. Hacer una animación en Gnuplot de la aproximación de Taylor para la función exponencial, utilizando hasta 8 términos, tal como aparece en la sexta figura del mismo artículo de la Wikipedia.

Estos programas use:

```
    program taylor

    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
!==========================
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

=========================================================

Program Taylor

implicit none

real(kind=8):: x, y, h, a, b
integer::j, n, nt

a=0
b=1
n=100
h=(b-a)/n
nt=100

  OPEN(1,FILE="Senox1",STATUS="UNKNOWN")

 do j=0,n
   x=a+j*h
 call Senox(x, y, nt)
 print*,  x, y
     WRITE(1,*)x,y
     WRITE(1,*)
     WRITE(1,*)
     WRITE(1,*)
end do
 close(1)

end program Taylor

=========================================================
subroutine Senox(x,f,nt)

implicit none
 real(kind=8):: x, f, m1, g, factorial
 integer:: i, nt, m
```

```
 f=0

 do i=0,nt-1
 m = (-1)**i
 g= 2*i+1
 m1= x**g/factorial(g)
 f= f+(m*m1)
! print*, i, x, f, m, m1
 end do
end subroutine Senox

=============================================================

function factorial(g)

 implicit none

  integer::i,factorial
  real(kind=8):: g

  do i=1,int(g)

   factorial=factorial*i

  end  do

 end function factorial
```