Universidad de La Laguna

Series de Taylor en la función arcsin(x)

Ana Gómez Pérez, Sara Luis Farrais y Shaila Verona Rodríguez

11 de mayo de 2014

Índice

Índice

Índice

Índice

Índice

Índice

El método de Taylor

El método de Taylor es uno de los algoritmos más antiguos utilizados para aproximar la solución de un problema de valor inicial en una ecuación diferencial ordinaria.

Fórmula del polinomio de taylor

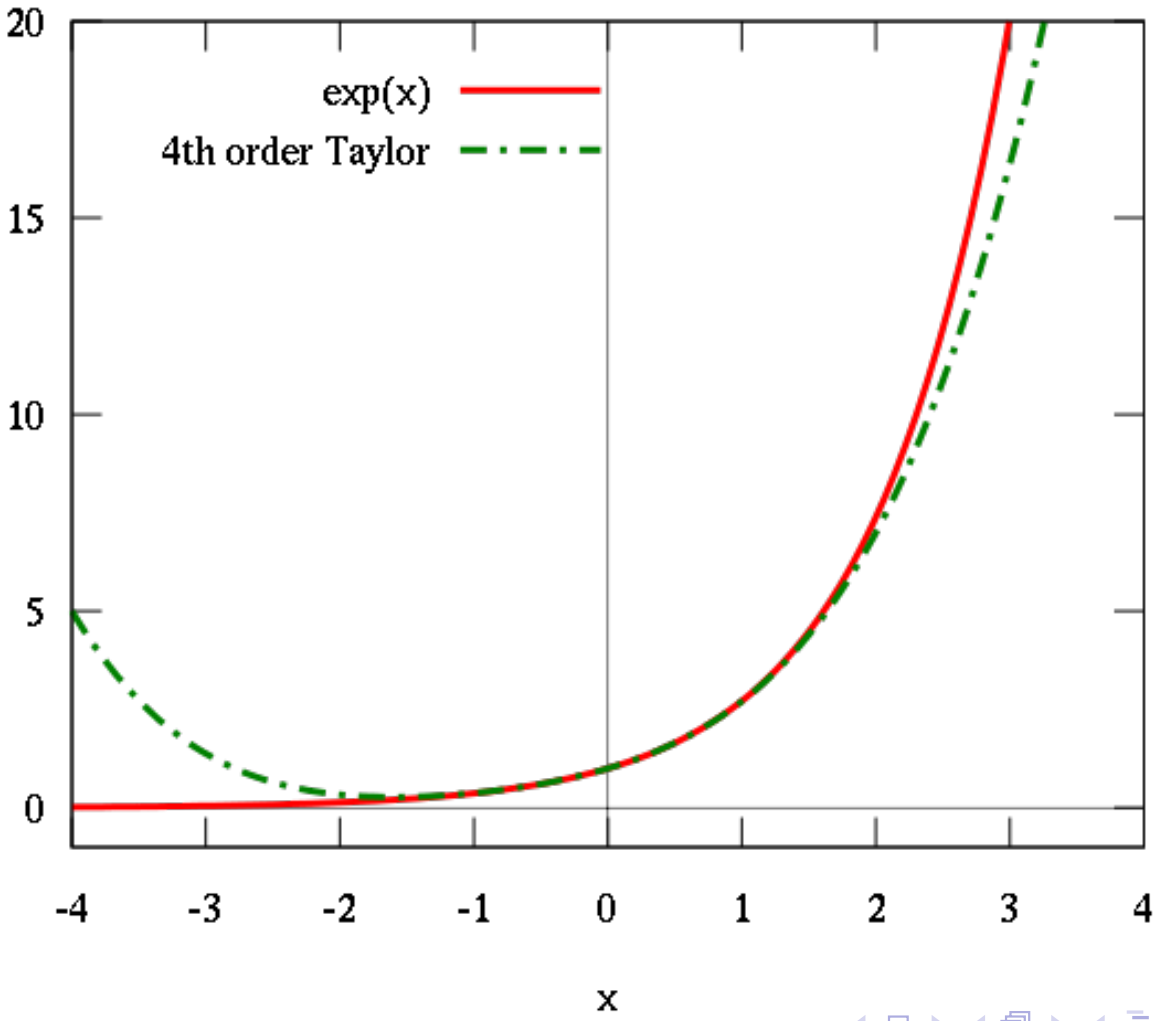
$$p(x) = f(a) + \frac{f'(a)}{1!} * (x-a) + \frac{f''(a)}{2!} * (x-a)^2 + \frac{f'''(a)}{3!} * (x-a)^3 + \ldots + \frac{f^n(a)}{n!}$$

El método de Taylor

El metodo de Taylor es una representación de una función como una infinita suma de términos. Estos términos se calculan a partir de las derivadas de la función para un determinado valor de la variable (respecto de la cual se deriva), lo que involucra un punto específico sobre la función.

El método de Taylor

como podemos ver en esta función exp(x), donde la aproximación se asmeja a dicha función aunque con un margen de error.



Código en Python

A continuación se muestra el código fuente creado en Python para la resolución del problema.

```
#! /src/bin/python
#!encoding: UTF-8

import math
from sympy import *
import time
import matplotlib.pylab as pl
import numpy as np

def factorial(n):
   if n <= 1:
     return 1
   else:
     prod = n*factorial(n-1)
     return prod

def taylor(n,x,a):
  c = Symbol('c')
  funcion = asin(c)
  suma=funcion.evalf(subs={c:a})
  for i in range (1,n+1):
    derv = diff(funcion, c)
    termino = derv.evalf(subs={c:a})
    resultado = (termino/factorial(i))*((x-a)**i)
    suma = suma + resultado
    funcion = derv
  return suma
```

Código en Python

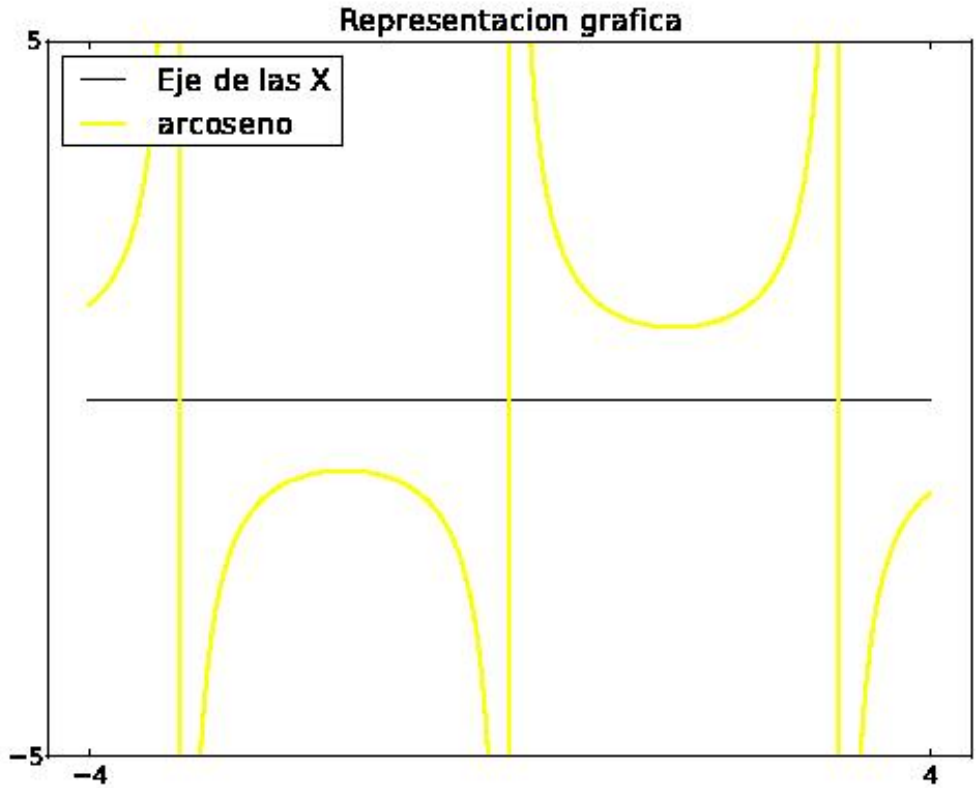
```
if __name__ == "__main__":

  n = int(raw_input("Introduzca el grado del polinomio:"))
  x = float(raw_input("Introduzca el punto donde se evalua el polinomio:"))
  a = float(raw_input("Introduzca el punto central donde se desea evaluar el polinomio:"))
  if (abs(a)>1)or(abs(x)>1):
    print 'Debe introducir valores de a entre [-1,1]'
    a = float(raw_input("Introduzca el punto central donde se desea evaluar el polinomio:"))
    x = float(raw_input("Introduzca el punto donde se evalua el polinomio:"))

start=time.time()
suma = taylor(n,x,a)
finish=time.time()-start
error = abs(asin(x)- suma)
print 'Valor de la aproximacion'
print suma
print 'Valor del error'
print error
print 'Tiempo que tarda el programa en ejecutarse'
print finish
```

Código en Python

```
pl.figure(figsize=(8,6), dpi=80)
pl.subplot(1,1,1)
X = np.linspace(-g, g, 256, endpoint=True)
C = 0*(X)
S = 1/np.sin(X)
pl.plot(X,C, color="black", linewidth=1.0, linestyle="-", label="Eje de las X")
pl.plot(X,S, color="yellow", linewidth=1.5, linestyle="-", label="arcoseno")
pl.legend(loc='upper left')
pl.xlim(X.min()*1.1,X.max()*1.1)
pl.xticks([-g, g])
pl.ylim(C.min()*1.1,C.max()*1.1)
pl.yticks([-h, h])
pl.title("Representacion grafica")
pl.savefig("grafica.eps", dpi=72)
pl.show()
```

Gráfica obtenida

Al aproximar con Taylor vamos a obtener otras ecuaciones según el error.



Experimentos realizados en Python

X	a	tiempo	Error
-1	1	0.0295159816742	4.5184232203082e+220
-0.25	1	0.0272219181061	1.76500872128679e+220
-0.2	1	0.0301878452301	1.6266320376450 3e+220
0	1	0.0254921913147	1.1296055819955 0e+220
0.2	1	0.0249121189117	7.2294757266755 5e+219
0.25	1	0.0299370288849	6.3540314008168 7e+219
1	1	0.0301327705383	0

Cuadro: Experimentos en el algoritmo con polinomios de 3 grado

Experimentos realizados en Python

Obsevando la tabla anterior, podemos apreciar que cuando más se parecen el valor del centro(a) y el puento X de la función, disminuye el valor del error. Por otro lado, cuantos más dispares son dichos valores mayor es el error.
Esto se da en los ejemplos propuestos en la tabla, en donde utilizamos un polinomio de grado 3 para todos los casos. Observamos que en primer caso, el valor del centro es 1 y el de la x es -1, el caso más dispar que puede darse, se obtiene el error más alto. Mientras, en el último caso, donde el valor del centro es 1 y de x es 1 se obtiene un valor 0 del error ya que el valor del centro y el punto X son iguales.

Experimentos realizados en Python

X	a	tiempo	Error
-1	1	0.0167179107666	2.695994666715 06e+67
-0.25	1	0.0157251358032	1.6849966666969 1e+67
-0.2	1	0.0181210041046	1.6175968000290 4e+67
0	1	0.0201098918915	1.3479973333575 3e+67
0.2	1	0.0200808048248	1.0783978666860 3e+67
0.25	1	0.0192070007324	1.0109980000181 5e+67
1	1	0.0163540840149	0

Cuadro: Experimentos en el algoritmo con polinomios de 1 grado

Experimentos realizados en Python

Podemos apreciar semeganza resultado que el experimento anterior. Sin embargo, apreciamos que al tratarse de un polinomio de grado menor se produce un menor error.

Conclusiones

Concluimos de los experimentos y gráficas realizados en Python que al aproximar la función se produce un error de medida. No obstante, aunque se produzcan erros, el método de Taylor eso muy usado en funciónes dificiles de calcular su valor, ya que su error es mínimo. Además, dicho error puede solventarse con el resto de Lagrange.

Bibliografía

- *Apuntes_de_la_asignatura* : *Análisis_Matemático_II*
- *http* : *//es.wikipedia.org/wiki/Serie_de_Taylor*
- *PuntoQ*
- *Análisis_Numérico_con_Aplicaciones.Gerald*Δ*Wheatley*.