



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

1 El método de Taylor

- 1 El método de Taylor
- 2 Código en Python

- 1 El método de Taylor
- 2 Código en Python
- 3 Experimentos realizados

- 1 El método de Taylor
- 2 Código en Python
- 3 Experimentos realizados
- 4 La Bibliografía

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

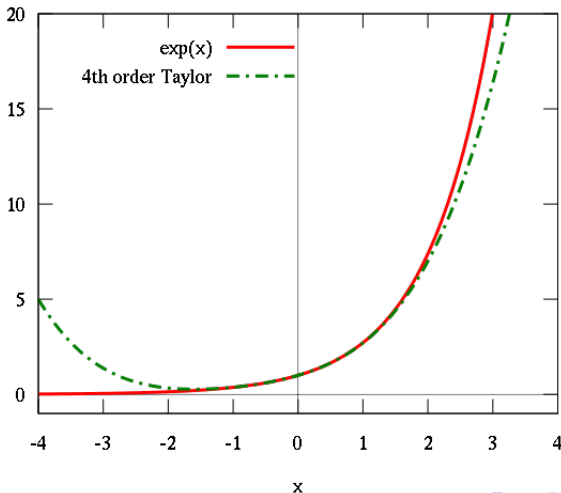
$$p(x) = f(a) + \frac{f'(a)}{1!} * (x - a) + \frac{f''(a)}{2!} * (x - a)^2 + \frac{f'''(a)}{3!} * (x - a)^3 + \dots + \frac{f^n(a)}{n!}$$

El método de Taylor

El método 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 asemeja 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.

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

import math
from sympy import *
import time
import matplotlib.pyplot as plt
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

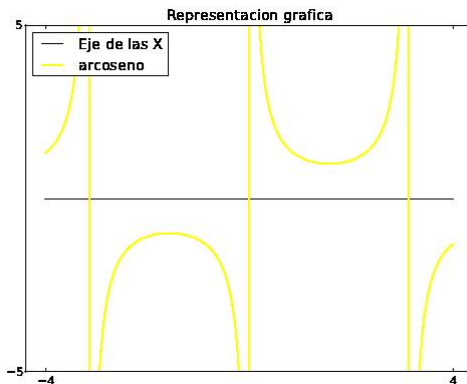
```
if __name__ == "__main__":  
    n = int(raw_input("Introduzca el grado del polinomio:"))  
    x = float(raw_input("Introduzca el punto donde se evalua el polinomio:"))  
    a = float(raw_input("Introduzca el punto central donde se desea evaluar el polinomio:"))  
    if (abs(a)>1)or(abs(x)>1):  
        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	4.51842232203082e+220
-0.25	1	0.0272219181061	1.76500872128679e+220
-0.2	1	0.0301878452301	1.62663203764503e+220
0	1	0.0254921913147	1.12960558199550e+220
0.2	1	0.0249121189117	7.22947572667555e+219
0.25	1	0.0299370288849	6.35403140081687e+219
1	1	0.0301327705383	0

Cuadro: Experimentos en el algoritmo

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.

Apuntes_de_la_asignatura : Análisis_Matemático_II

[http : //es.wikipedia.org/wiki/Serie_de_Taylor](http://es.wikipedia.org/wiki/Serie_de_Taylor)

PuntoQ

Análisis_Numérico_con_Aplicaciones.Gerald Δ Wheatley.