

Series numericas: Función $\sin(x)$

Jorge Antonio Herrera Alonso, Elizabeth Hernández Martín, Yessica
Sabrina Gómez Buso

12 de mayo de 2014

Facultad de Matemáticas
Universidad de La Laguna

1 Motivación y objetivos

- 1 Motivación y objetivos
- 2 Fundamentos teóricos
 - Teorema de Taylor
 - Polinomio de Taylor

- 1 Motivación y objetivos
- 2 Fundamentos teóricos
 - Teorema de Taylor
 - Polinomio de Taylor
- 3 Procedimiento experimental
 - Descripción de los experimentos
 - Descripción del material
 - Análisis de los resultados

- 1 Motivación y objetivos
- 2 Fundamentos teóricos
 - Teorema de Taylor
 - Polinomio de Taylor
- 3 Procedimiento experimental
 - Descripción de los experimentos
 - Descripción del material
 - Análisis de los resultados
- 4 Algoritmo

- 1 Motivación y objetivos
- 2 Fundamentos teóricos
 - Teorema de Taylor
 - Polinomio de Taylor
- 3 Procedimiento experimental
 - Descripción de los experimentos
 - Descripción del material
 - Análisis de los resultados
- 4 Algoritmo
- 5 Gráfica

- 1 Motivación y objetivos
- 2 Fundamentos teóricos
 - Teorema de Taylor
 - Polinomio de Taylor
- 3 Procedimiento experimental
 - Descripción de los experimentos
 - Descripción del material
 - Análisis de los resultados
- 4 Algoritmo
- 5 Gráfica
- 6 Tablas
 - Experimentos 1 y 2
 - Experimentos 3 y 4

- 1 Motivación y objetivos
- 2 Fundamentos teóricos
 - Teorema de Taylor
 - Polinomio de Taylor
- 3 Procedimiento experimental
 - Descripción de los experimentos
 - Descripción del material
 - Análisis de los resultados
- 4 Algoritmo
- 5 Gráfica
- 6 Tablas
 - Experimentos 1 y 2
 - Experimentos 3 y 4
- 7 Bibliografía

Motivación

Aprendizaje orientado al lenguaje de programación *Python*, el procesador de texto *LaTEX* y el creador de presentaciones *Beamer*.

Objetivos

- **Objetivo principal:** Implementación de *Python* del método de Taylor.

Motivación

Aprendizaje orientado al lenguaje de programación *Python*, el procesador de texto *LaTEX* y el creador de presentaciones *Beamer*.

Objetivos

- **Objetivo principal:** Implementación de *Python* del método de Taylor.
- **Objetivo específico:** Aproximación de una función mediante el método de Taylor.

Teorema de Taylor

Teorema de Taylor:

El Teorema de Taylor 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.

Definición

Sea I un intervalo, c un punto de I , f una función definida en I . Supongamos que f es derivable en todos los puntos hasta el orden $n - 1$ y que existe $f^n(c)$. Entonces:

$$\lim_{x \rightarrow c} \frac{1}{(x - c)^n} [f(x) - f(c) - f'(c)(x - c) - \frac{1}{2!} f''(c)(x - c)^2 - \dots - \frac{1}{n!} f^n(c)(x - c)^n] = 0$$

Definición

Dada una función f derivable n veces en un punto c . se llama polinomio de Taylor en c de orden n al polinomio:

$$f(x) = f(c) + f'(c)(x - c) + \frac{1}{2!}f''(c)(x - c)^2 + + \frac{1}{n!}f^n(c)(x - c)^n$$

Descripción:

- 1 Realización de varios códigos en Lenguaje *Python*.

Descripción:

- 1 Realización de varios códigos en Lenguaje *Python*.
- 2 Repetición de la ejecución del código.

Descripción:

- 1 Realización de varios códigos en Lenguaje *Python*.
- 2 Repetición de la ejecución del código.
- 3 Aproximación $f(x) = \sin(x)$ mediante el método de Taylor.

Descripción:

- 1 Realización de varios códigos en Lenguaje *Python*.
- 2 Repetición de la ejecución del código.
- 3 Aproximación $f(x) = \sin(x)$ mediante el método de Taylor.
- 4 Error estimado.

Descripción:

- 1 Realización de varios códigos en Lenguaje *Python*.
- 2 Repetición de la ejecución del código.
- 3 Aproximación $f(x) = \sin(x)$ mediante el método de Taylor.
- 4 Error estimado.
- 5 Tiempo CPU.

El material requerido para la realización del trabajo ha sido una computadora.

Características de la computadora:

- CPU type: Intel(R) Core(TM) i3-2328M CPU @ 2.50GHz

El material requerido para la realización del trabajo ha sido una computadora.

Características de la computadora:

- CPU type: Intel(R) Core(TM) i3-2328M CPU @ 2.50GHz
- vendor ID GenuineIntel

El material requerido para la realización del trabajo ha sido una computadora.

Características de la computadora:

- CPU type: Intel(R) Core(TM) i3-2328M CPU @ 2.50GHz
- vendor ID GenuineIntel
- CPU speed 1200.000Hz

El material requerido para la realización del trabajo ha sido una computadora.

Características de la computadora:

- CPU type: Intel(R) Core(TM) i3-2328M CPU @ 2.50GHz
- vendor ID GenuineIntel
- CPU speed 1200.000Hz
- cache size 2048 KB

Experimentos realizados:

- 1 Grado del polinomio y punto c , fijos. El punto x cambia.

Experimentos realizados:

- 1 Grado del polinomio y punto c , fijos. El punto x cambia.
- 2 Grado del polinomio y punto x , fijos. El punto c cambia.

Experimentos realizados:

- 1 Grado del polinomio y punto c , fijos. El punto x cambia.
- 2 Grado del polinomio y punto x , fijos. El punto c cambia.
- 3 Grado del polinomio toma el valor 10. Punto c y punto x , igual que el experimento 1.

Experimentos realizados:

- 1 Grado del polinomio y punto c , fijos. El punto x cambia.
- 2 Grado del polinomio y punto x , fijos. El punto c cambia.
- 3 Grado del polinomio toma el valor 10. Punto c y punto x , igual que el experimento 1.
- 4 Grado del polinomio toma el valor de 100. El punto c , fijo. El punto x solo varia en las cantidad de cifras decimales.

Algoritmo principal:

```
#!/src/bin/python
#encoding: UTF-8
import math
from sympy import *
import modulo
import time
import numpy as np
import matplotlib.pyplot as mp

numero = int(raw_input("Introduzca el grado que desea que tenga el Polinomio de Taylor: "))
centro = float(raw_input("Introduzca el punto central donde desea que se evalúe el Polinomio de Taylor: "))
x = float(raw_input("Introduzca el punto x donde desea evaluar el Polinomio de Taylor: "))

modulo.taylor1(x, numero, centro)
y1 = np.linspace(-np.pi, np.pi, 256, endpoint=True)
y1 = np.arange(-8,8,0.1)
y2 = np.sin(y1)

mp.plot(y1,y2, color = 'blue',linewidth=2.5, linestyle="--", label="seno")
mp.legend(loc=0)

# Para poner el simbolo pi en el eje x se usa LaTeX.
mp.xticks([-2*np.pi, -3*np.pi/2, -np.pi, -np.pi/2, 0, np.pi/2, np.pi, 3*np.pi/2, 2*np.pi],
[r'$2\pi$',r'$-\frac{3\pi}{2}$',r'$-\pi$',r'$-\frac{\pi}{2}$',r'$0$',r'$+\frac{\pi}{2}$',r'$+\pi$',
r'$\frac{3\pi}{2}$',r'$2\pi$'])

mp.title("Representacion grafica")

mp.savefig('grafico.png',dpi = 100)

mp.show()
```

Algoritmo del módulo

```
#!/src/bin/python
import math
from sympy import *
import time

def factorial(numero):
    factorial = 1
    while(numero > 1):
        factorial = factorial * numero
        numero = numero - 1
    return factorial

def taylor1(x, numero, centro):
    inicio = time.time()
    c = Symbol('c')
    f = sin(c)
    func = f.evalf(subs={c: centro})
    suma = func
    for i in range(1, numero + 1):
        f_i = diff(f, c)
        v = f_i.evalf(subs={c: centro})
        s = (v / factorial(i)) * ((x - centro) ** i)
        suma = suma + s
        f = f_i
    error = func - suma
    print 'El valor de la funcion original sin(%f) es igual a %f ' % (centro, func)
    print 'El Polinomio de Taylor de grado n=%d en el punto centro c=%f'
    evaluada en el punto x=%f es igual a %f' % (numero, centro, x, suma)
    print 'El Error de la funcion original con el Polinomio de Taylor es: error=%f' % error
    l=[numero, centro, x, suma, error]
    f = open("Taylor.tex", 'a')
```

```
f.write('Grado del Polinomio (n) Punto Central (c) Punto de Evaluacion (x)  
Aproximacion  
Error Tiempo CPU \n ')  
fin = time.time()  
tiempo_total = fin - inicio  
l=l+[tiempo_total]  
f.write(str(l))  
f.write("\n")  
f.close()  
f=open("Taylor.tex","r")  
print(f.read())  
f.close()  
return taylor1
```

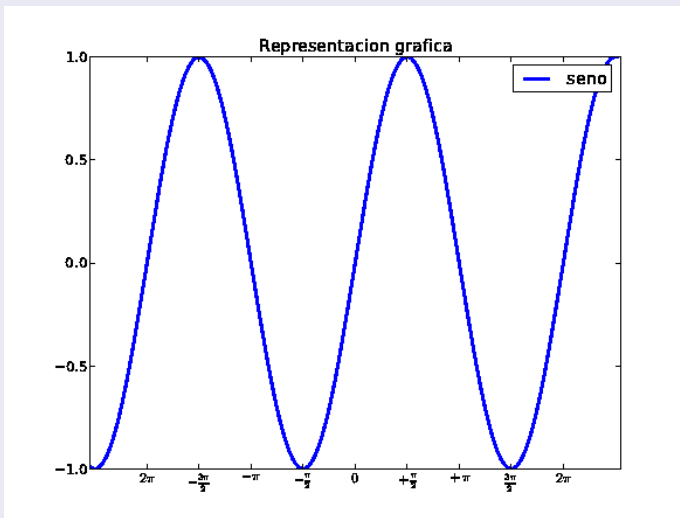


Figure: Gráfica de la función original

Grado	Punto c	Punto x	Aproximacion	Error	Tiempo CPU
4	10.0	-10.0	-4620.52781304922	4619.98379193833	0.00790095329284668
4	10.0	-5.0	-1545.27783908746	1545.73381797657	0.00798487663269043
4	10.0	5.0	-21.1962728645601	20.6522517536707	0.008163928985595703
4	10.0	9.0	0.404548172498635	-0.948569283388005	0.008239030838012695
4	10.0	9.9	-0.457535964436753	-0.086485146452617	0.00795292854309082
4	10.0	10.0	-0.544021110889370	0	0.008179903030395508

Table: Tabla de datos obtenidos. Experimento 1

Grado	Punto c	Punto x	Aproximacion	Error	Tiempo CPU
4	-7.0	2.0	-238.466744388979	237.809757790260	0.008273124694824219
4	-2.0	2.0	-0.559778321379882	-0.349519105445799	0.008683919906616211
4	0.0	2.0	0.666666666666667	-0.666666666666667	0.007463932037353516
4	2.0	2.0	0.909297426825682	0	0.007899999618530273
4	7.0	2.0	21.4904658168047	-20.8334792180859	0.007979869842529297

Table: Tabla de datos obtenidos. Experimento 2

Grado	Punto c	Punto x	Aproximacion	Error	Tiempo CPU
10	10.0	-10.0	2226636.13194962	-2226636.67597073	0.012552976608276367
10	10.0	-5.0	124685.902313545	-124686.446334656	0.011905908584594727
10	10.0	0.0	1920.68755204547	-1921.23157315635	0.011946916580200195
10	10.0	5.0	0.163743739637233	0.707764850526603	0.08893513679504395
10	10.0	9.0	0.412118507257685	-0.956139618147055	0.014183998107910156
10	10.0	9.9	-0.457535893775321	0.0864852171140484	0.012012958526611328
10	10.0	10.0	-0.544021110889370	0	0.014500856399536133

Table: Tabla de datos obtenidos. Experimento 3

Grado	Punto c	Punto x	Aproximacion	Error	Tiempo CPU
100	-1.0	-0.99999	-0.8414655817427645	-5.40306513208133e	0.06290507316589355
100	-1.0	-0.9	-0.783326909627484	-0.0581440751804130	0.06340909004211426

Table: Tabla de datos obtenidos. Experimento 4



Tutorial Beamer.



Apuntes de análisis I.



Comandos latex - Páginas - Fórmulas - Bibliografías.