UNIVERSIDAD TECNICA NACIONAL
INGENIERIA ELECTRONICA

Proyecto

Angie Marchena Mondell

Métodos numéricos

Noviembre de 2021

Análisis de la ecuación

Se tiene la siguiente ecuación

$$x^3 + x^2 + 6x + 55\cos(x) = 10xe^{0.2x} + x^2e^{-0.3x}$$

Primero es necesario igualar a 0 la ecuación.

$$x^3 + x^2 + 6x + 55\cos(x) = 10xe^{0.2x} + x^2e^{-0.3x}$$

$$x^3 + x^2 + 6x + 55\cos(x) - 10xe^{0.2x} - x^2e^{-0.3x} = 0$$

Por lo que se obtiene la función

$$f(x) = x^3 + x^2 + 6x + 55\cos(x) - 10xe^{0.2x} - x^2e^{-0.3x}$$

Con esto podemos recurrir a un graficador para tener una idea de la función y cuales podrían ser sus raíces.

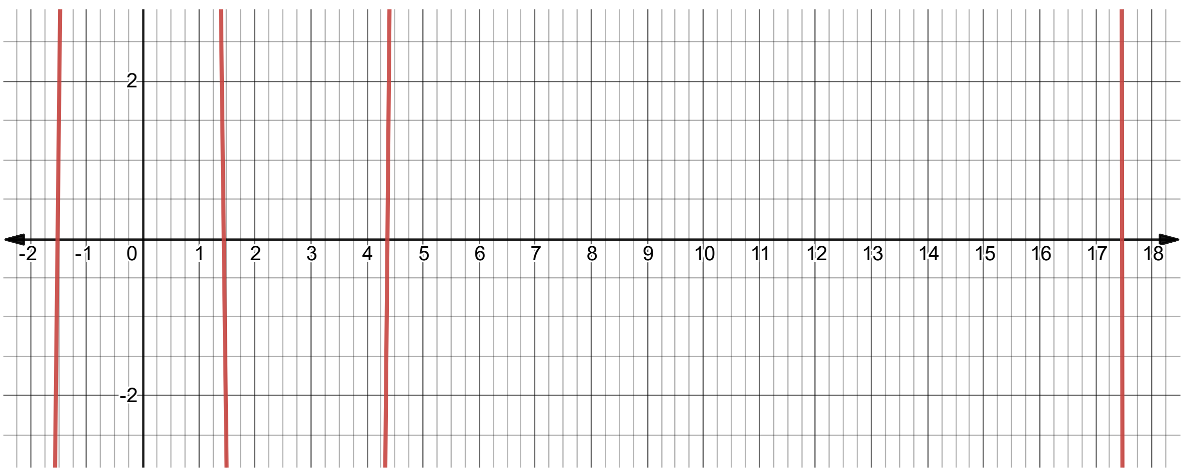


Figura 1: Grafica asociada a la función dada para el proyecto.

Para la ecuación se seleccionó el método de la secante, para esto se necesita 2 valores iniciales, ya que se utiliza la formula tradicional de este método.

Método de la secante

Las soluciones se calculan mediante la siguiente formula.

$$x_{k+1} = x_k - \left(\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}\right) f(x_k)$$

Con valores iniciales x_0 y x_1, los cuales dependen.

La ventaja que se tiene con respecto a otros es que no requiere calcular su derivada, así también que este converge muy rápido, respecto a otros métodos estudiados.

Para los valores iniciales, son necesario 2 por solución, por lo que se proponen los siguientes

- Para la primera solución $x_0 = -2, x_1 = -1$
- Para la segunda solución $x_0 = 1, x_1 = 2$
- Para la tercera solución $x_0 = 4, x_1 = 5$
- Para la cuarta solución $x_0 = 17, x_1 = 18$

El código del método de la secante es el siguiente:

```
#Metodo de la secante
def secante(f1,x0,x1,tol,iterM):
    e=[]
    sol=[]
    k=0
    error=tol+1

    while (k < iterM) and (error>tol):
        xk= x1-((x1-x0)/(f1(x1)-f1(x0)))*f1(x1)
        error=abs(f1(xk))
        k+=1
        x0=x1
        x1=xk
        e=e+[error]
        sol = sol + [xk]

    return [xk, k, error]
```

Figura 2: Código de la secante.

Este método recibe los siguientes parámetros:

- `f1`: Función a calcular las raíces.
- `x0` y `x1`: Valores cercanos a la raíz, tomados como iniciales.
- `tol`: Tolerancia o error máximos permitido.
- `interM`: Iteraciones máximas permitidas.

Además, como salida tenemos un vector, con la solución encontrada, interacciones máximas y valor del error obtenido.

Cálculo de todas las soluciones

Para esto se desarrolló un método el cual permite calcular todas las soluciones de cualquier ecuación, solo es necesario una matriz con los valores iniciales por solución, el cual debe tener la siguiente forma:

$$\text{vector} = \begin{pmatrix} x1_0 & x1_1 \\ x2_0 & x2_1 \\ x3_0 & x3_1 \\ \vdots & \vdots \\ xN_0 & xN_1 \end{pmatrix}$$

El código realizado se encuentra a continuación

```
#funcion que calcula todas las soluciones de la ecuacion
def soluciones(f,vector, tol, iterM):
    resultado = []
    for i in vector:
        valor = secante(f,i[0],i[1],tol,iterM)
        resultado += [valor[0]]
    print("Valores de las soluciones de la ecuacion")

    k=1
    for i in resultado:
        print(f'x{k} = {i}')
        k+=1
```

Figura 3: Método que calcula todas las soluciones.

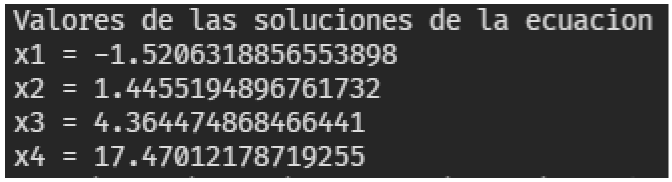
Este método recibe los siguientes parámetros:

- `f`: Función a calcular las raíces.
- `vector`: Valores cercanos a la raíz, tomados como iniciales, los cuales tiene la forma de la matriz anteriormente mencionado.
- `tol`: Tolerancia o error máximos permitido.
- `interM`: Iteraciones máximas permitidas.

Lo que hace el método es llamar a la secante, este calcula la raíz y la guarda en un vector de resultados, finalmente los imprime en orden.

Resultados

Los resultados obtenidos se muestran en la figura 4.



```
Valores de las soluciones de la ecuacion
x1 = -1.5206318856553898
x2 = 1.4455194896761732
x3 = 4.364474868466441
x4 = 17.47012178719255
```

Figura 4: Resultados obtenidos.

Como se ve los resultados obtenidos son los esperados, ya que gráficamente se vio la posición de las raíces de la función.