Universidad Central de Venezuela

Facultad de Ingeniería

Escuela de Ingeniería Eléctrica

Laboratorio N°3

Periodo 2023-3

29/01/2023

CI 29571461 Ricardo Santana

1. Introducción

Gracias a la matemática es posible representar modelos que describan el comportamiento de algunos de los fenomenos de la naturaleza, no obstante muchos de estos modelos son construidos con funciones no lineales, como las trigonometricas, las logaritmicas, las exponenciales, etc; lo cual dificulta en parte los cálculos necesarios para llegar a cierta hipótesis. Sin embargo gracias los procesadores modernos es posible obtener una aproximación razonable y que, a través de diversos métodos numéricos que pueden ser programados, se puede utilizar con fines práticos.

Uno de los cálculos más realizados en una función es la búsqueda de sus raíces, es decir, cuando la función se aproxima a cero, utilizado por ejemplo en la optimización. En la actualidad exiten diversos métodos númericos para resolver ésta problemática, de los cuales se presentarán los métodos de bisección, de Newton, de la secante y a través de punto fijo.

Por otra parte, así como es posible obtener un sistema de ecuaciones lineales, también puede existir la posibilidad de obtener uno no lineal, entonces surge la necesidad de reponder la siguiente interrogante ¿Es posible calcular una solución razonable?; utilizando métodos numericos sí. Éstos últimos, como el de Newton, aproximan a través de iteraciones un resultado con errores que pueden considerarse despreciables dependiendo el uso que se le vaya a dar.

2. Marco teórico

2.1 Biseccion [1]

El método de bisección se basa en el teorema del valor intermedio, que en una de sus versiones establece:

Teorema 1. Si $f : [a, b] \to \mathbb{R}$ es una función continua y $f(a)f(b) < 0$, entonces existe $c \in (a, b)$ tal que $f(c) = 0$.

Ahora, si $f : [a, b] \to \mathbb{R}$ es una función continua y $f(a)f(b) < 0$, por el teorema del valor intermedio se conoce que existe al menos una solución de la ecuación $f(x) = 0$, y es posible aplicar el siguiente procedimiento, denominado método de bisección, para determinar dicha solución.

1. Definir $a_0 = a$ y $b_0 = b$.
2. Calcular el punto medio c_0 del intervalo $[a_0, b_0]$, es decir, $c_0 = \frac{(a_0+b_0)}{2}$

   a) Si $f(c_0) = 0$. entonces c0 es un cero de la función, y por lo tanto una solución al problema. Terminar el procedimiento. b) Si $f(a_0).\, f(c_0) > 0$, entonces existe un cero de la función en el intervalo (c_0, b0).Seleccionar este intervalo y definir $a_1 = c_0$ y $b_1 = b_0$. c) $Sif(a_0).\, f(c_0) < 0$, existe un cero de la función en el intervalo (a_0, c_0). Seleccionar este intervalo y definir $a_1 = a_0$ y $b_1 = c_0$.

   Nota: observar que ninguno de los tres casos mencionados arriba se puede dar simultáneamente. Además, si la situación es la del caso b), donde se tiene seguridad que hay un cero de la función en el intervalo (c_0, b_0), esto no descarta que pueda existir otra raíz en el intervalo (a_0, c_0).
3. Repetir el procedimiento con el nuevo intervalo $[a_1, b_1]$.

2.2 Metodo de Newton [1]

Este método se basa en elementos del cálculo diferencial y su interpretación gráfica. Es uno de los métodos más eficientes para determinar, con una precisión deseada, una aproximación de la solución de la ecuación $f(x) = 0$.

Se puede interpretar el método de Newton de la siguiente manera:

1. Seleccionar un punto inicial (semilla) p_0.
2. Calcular la ecuación de la recta tangente a la curva $f(x)$ que pasa por el punto $(p_0, f(p_0))$.
3. Determinar el corte con el eje x de la recta tangente y nombrar como p_1.
4. Si $f(p_1) \neq 0$ repetir los pasos 2, 3 y 4 para p_1.

Ahora, la ecuación de la recta tangente a la curva $y = f(x)$ que pasa por el punto $(p_0, f(p_0))$ tiene como pendiente $m = f'(p_0)$ y corresponde a $y = f'(p_0)x + [f(p_0) - f'(p_0)p_0]$.

Para calcular el cero de esta recta se reemplaza y por cero $0 = f'(p_0)x + f(p_0) - f'(p_0)p_0$ y al despejar x, se tiene $x = p_0 - \frac{f(p_0)}{f'(p_0)}$. Dicho valor de x se nombra como p_1 y por tanto:
$p_1 = p_0 - \frac{f(p_0)}{f'(p_0)}$.

Si se repite el proceso, pero ahora con el punto $(p_1, f(p_1))$, se obtiene: $p_2 = p_1 - \frac{f(p_1)}{f'(p_1)}$.

Repitiendo ahora con $(p_2, f(p_2))$ $p_3 = p_2 - \frac{f(p_2)}{f'(p_2)}$, y finalmente, en la n-ésima iteración $p_n = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})}$

2.2.1 Convergencia con el método de Newton [2]

La derivación del método de Newton por medio de la serie de Taylor al inicio de la sección señala la importancia de una aproximación inicial precisa. La suposición crucial es que el término relacionado con $(p - p_0)^2$ es, en comparación con $|p - p_0|$, tan pequeño que se puede eliminar. Claramente esto será falso a menos que p_0 sea una buena aproximación para p. Si p_0 no está suficientemente cerca de la raíz real, existen pocas razones para sospechar que el método de Newton convergerá en la raíz. Sin embargo, en algunos casos, incluso las malas aproximaciones iniciales producirán convergencia.

Teorema 2. Sea $f \in C^2[a, b]$. Si $p \in (a, b)$ es tal que $f(p) = 0$ y $f'(p) = 0$, entonces existe una $\delta > 0$ tal que el método de Newton genera una sucesión $\{pn\}_{n=1}^{\infty}$ que converge a p para cualquier aproximación inicial $p_0 \in [p - \delta, p + \delta]$.

2.3 Metodo de la secante [1]

Aunque el método de Newton-Raphson es uno de los más eficientes para determinar una solución de la ecuación $f(x) = 0$, la necesidad de conocer $f'(x)$ representa una de sus mayores debilidades, pues el cálculo de la derivada requiere y representa más operaciones por realizar. Una manera de evitar el problema de calcular la derivada, es recordar que

$$f'(p_{n-1}) = \lim_{x \to p_{n-1}} \frac{f(x) - f(p_{n-1})}{x - p_{n-1}}$$

y dado que p_{n-2} se encuentra cerca de p_{n-1} en caso de ser convergente el método de Newton-Raphson, entonces es posible aproximar $f'(p_{n-1})$ por

$$f'(p_{n-1}) \approx \frac{f(p_{n-2}) - f(p_{n-1})}{p_{n-2} - p_{n-1}}$$

Al sustituir en la fórmula de iteración de Newton-Raphson, se obtiene

$$p_n = p_{n-1} - \frac{f(p_{n-1}).\,(p_{n-1} - p_{n-2})}{f(p_{n-1}) - f(p_{n-2})}$$

2.4 Metodo de punto fijo [1]

Desde la antigua Babilonia es conocido el siguiente método para calcular $\sqrt{A}$:

1. Seleccionar x_0 cercano a $\sqrt{A}$.
2. Calcular $x_{n+1} = \frac{1}{2}(x_n + \frac{A}{x_n})$ para n > 0.

Este método es eficiente y uno de los mejores ejemplos de la técnica que se desarrolla en esta sección para aproximar soluciones de ecuaciones. Aunque existen muchas técnicas numéricas que dan solución a la ecuación $f(x) = 0$, tal vez una de las más famosas es el llamado método del punto fijo, un procedimiento para aproximar la solución de la ecuación $x = g(x)$. Para comprender este método, es necesario dar algunas definiciones y ejemplos.

Definición 1. Un punto fijo de una función $g(x)$ es un valor c tal que $g(c) = c$.

Teorema 3. Sean $[a, b] \subseteq \mathbb{R}$ y $g : [a, b] \to [a, b]$ una función continua. Entonces g tiene un punto fijo en $[a, b]$.

Ahora, para calcular una aproximación de un punto fijo de una función g es posible aplicar el siguiente procedimiento:

1. Seleccionar un punto inicial p_0.
2. Calcular $g(p_0)$ y nombrarlo como p_1.
3. Si $g(p_1) \neq p_1$, repetir los pasos 2 y 3 para p_1.

Teorema 4. Sean $g : [a, b] \to [a, b]$ una función continua, $s \in [a, b]$ tal que $g(s) = s$ y $|g'(x)| \leq \alpha < 1$, donde $x \in (a, b)$ y α es constante. Sea $p_0 \in [a, b]$. Entonces la sucesión $\{p_n\}_{n=0}^{\infty}$ dada por $p_{n+1} = g(p_n)$ es tal que $p_n \to s$ cuando $n \to \infty$.

2.5 Análisis de error para métodos iterativos [2]

2.5.1 Orden de convergencia

Suponga que $\{p_n\}_{n=0}^{\infty} = 0$ es una sucesión que converge a p, con $p_n \neq p$ para todas las n. Si existen constantes positivas λ y α con

$$\lim_{n \to \infty} \frac{|p_{n+1} - p|}{|p_n - p|^{\alpha}} = \lambda$$

Entonces $\{p_n\}_{n=0}^{\infty} = 0$ converge a p de orden α, con constante de error asintótica λ.

Se dice que una técnica iterativa de la forma $p_n = g(p_{n-1})$ es de orden α si la sucesión $\{p_n\}_{n=0}^{\infty} = 0$ converge a la solución $p = g(p)$ de orden α.

En general, una sucesión con un alto orden converge más rápidamente que una sucesión con un orden más bajo. La constante asintótica afecta la velocidad de convergencia pero no el grado del orden. Se presta atención especial a dos casos:

1. Si $\alpha = 1 (y\ \lambda < 1)$, la sucesión es linealmente convergente.
2. Si $\alpha = 2$, la sucesión es cuadráticamente convergente.

2.6 seudocódigo: Método de Newton para sistemas [2]

Para aproximar la solución del sistema no lineal $F(x) = 0$, dada una aproximación inicial x:

ENTRADA número n de ecuaciones y valores desconocidos; aproximación inicial $x = (x_1, \ldots, x_n)^t$, tolerancia TOL; número máximo de iteraciones N.

SALIDA solución aproximada $x = (x_1, \ldots, x_n)^t$ o un mensaje que indica que se excedió el número de iteraciones.

Paso 1 Determine $k = 1$.

Paso 2 Mientras $(k \leq N)$ haga los pasos 3-7.

Pase 3 Calcule $F(x)$ y $J(x)$, donde $J(x)_{i,j} = \frac{\partial f_i(x)}{\partial x_j}$ para $1 \leq i, j \leq n$.

Paso 4 Resuelva el sistema lineal $n \times n$ $J(x)y = -F(x)$.

Paso 5 Determine $x = x + y$

Paso 6 Si $||y|| < TOL$ entonces SALIDA (x); (*El procedimiento fue exitoso.*) PARE.

Paso 7 Determine $k = k + 1$

Paso 8 SALIDA ('Número máximo de iteraciones excedido'); (*El procedimiento no fue exitoso.*) PARE.

3 Práctica

Herramientas a utilizar:

```
#importando librerias necesarias
import math as mt
import sympy as sp
import numpy as np
from sympy.plotting import plot as  plt

#importando programas diseñados
from cerosDeFunciones import *
from sisEcuNoLin import *
from funciones import *
```

3.1 Problema Test

Se hallará la raíz de la función:

$$f_1(x) = cos(x) - x$$

```
#función dentro del programa funciones definida como trig(x)

#intancia utilizada para el problema test
iteraciones = 50
```

```
tolerancia = 1e-9
aproxInic = 0.5
```

3.1.1 Bisección

```
test.biseccion(-2,2)
raizBiseccion = test.p
print('raíz obtenida por el método de bisección:')
print('x = ', raizBiseccion)
print('f1(x) = ', trig(raizBiseccion))
```

```
     i = 1 , p = 0.000000000000
     i = 2 , p = 1.000000000000
     i = 3 , p = 0.500000000000
     i = 4 , p = 0.750000000000
     i = 5 , p = 0.625000000000
     i = 6 , p = 0.687500000000
     i = 7 , p = 0.718750000000
     i = 8 , p = 0.734375000000
     i = 9 , p = 0.742187500000
     i = 10, p = 0.738281250000
     i = 11, p = 0.740234375000
     i = 12, p = 0.739257812500
     i = 13, p = 0.738769531250
     i = 14, p = 0.739013671875
     i = 15, p = 0.739135742188
     i = 16, p = 0.739074707031
     i = 17, p = 0.739105224609
     i = 18, p = 0.739089965820
     i = 19, p = 0.739082336426
     i = 20, p = 0.739086151123
     i = 21, p = 0.739084243774
     i = 22, p = 0.739085197449
     i = 23, p = 0.739084720612
     i = 24, p = 0.739084959030
     i = 25, p = 0.739085078239
     i = 26, p = 0.739085137844
     i = 27, p = 0.739085108042
     i = 28, p = 0.739085122943
     i = 29, p = 0.739085130394
     i = 30, p = 0.739085134119
     i = 31, p = 0.739085132256
     i = 32, p = 0.739085133187
     raíz obtenida por el método de bisección:
     x =  0.7390851331874728
     f1(x) =  4.633871064640971e-11
```

3.1.2 Newton

```
raizNewton = test.newton()
raizNewton = test.p
print('raíz obtenida por el método de Newton:')
print('x = ', raizNewton)
print('f1(x) = ', trig(raizNewton))
```

```
     i = 1 , p = 0.755222417106
     i = 2 , p = 0.739141666150
     i = 3 , p = 0.739085133921
     i = 4 , p = 0.739085133215
     i = 5 , p = 0.739085133215
     i = 6 , p = 0.739085133215
     i = 7 , p = 0.739085133215
     i = 8 , p = 0.739085133215
     i = 9 , p = 0.739085133215
     i = 10, p = 0.739085133215
     i = 11, p = 0.739085133215
     i = 12, p = 0.739085133215
     i = 13, p = 0.739085133215
     i = 14, p = 0.739085133215
     i = 15, p = 0.739085133215
     i = 16, p = 0.739085133215
     i = 17, p = 0.739085133215
     i = 18, p = 0.739085133215
     i = 19, p = 0.739085133215
     i = 20, p = 0.739085133215
     i = 21, p = 0.739085133215
     i = 22, p = 0.739085133215
     i = 23, p = 0.739085133215
     i = 24, p = 0.739085133215
     i = 25, p = 0.739085133215
     i = 26, p = 0.739085133215
     i = 27, p = 0.739085133215
     i = 28, p = 0.739085133215
     i = 29, p = 0.739085133215
     i = 30, p = 0.739085133215
     i = 31, p = 0.739085133215
     i = 32, p = 0.739085133215
     i = 33, p = 0.739085133215
     i = 34, p = 0.739085133215
     i = 35, p = 0.739085133215
     i = 36, p = 0.739085133215
     i = 37, p = 0.739085133215
     i = 38, p = 0.739085133215
     i = 39, p = 0.739085133215
     i = 40, p = 0.739085133215
     i = 41, p = 0.739085133215
     i = 42, p = 0.739085133215
     i = 43, p = 0.739085133215
     i = 44, p = 0.739085133215
     i = 45, p = 0.739085133215
     i = 46, p = 0.739085133215
     i = 47, p = 0.739085133215
     i = 48, p = 0.739085133215
     i = 49, p = 0.739085133215
```

```
i = 50, p = 0.739085133215
Iteraciones agotadas: Error!
raíz obtenida por el método de Newton:
x =  0.7390851332151607
f1(x) =  0.0
```

3.1.3 Secante

```
test.p0, test.p00 = -5, 5 #modificando aproximaciones iniciales
raizSecante = test.secante()
raizSecante = test.p
print('raíz obtenida por el método de Secante:')
print('x = ', raizSecante)
print('f1(x) = ', trig(raizSecante))
```

```
i = 2 , p = 0.283662185463
i = 3 , p = 0.875203409055
i = 4 , p = 0.722980400685
i = 5 , p = 0.738630908537
i = 6 , p = 0.739086762519
i = 7 , p = 0.739085133052
i = 8 , p = 0.739085133215
raíz obtenida por el método de Secante:
x =  0.7390851332151606
f1(x) =  1.1102230246251565e-16
```

3.1.4 Punto fijo

$$f_1(x) = cos(x) - x = 0$$

Despejando x

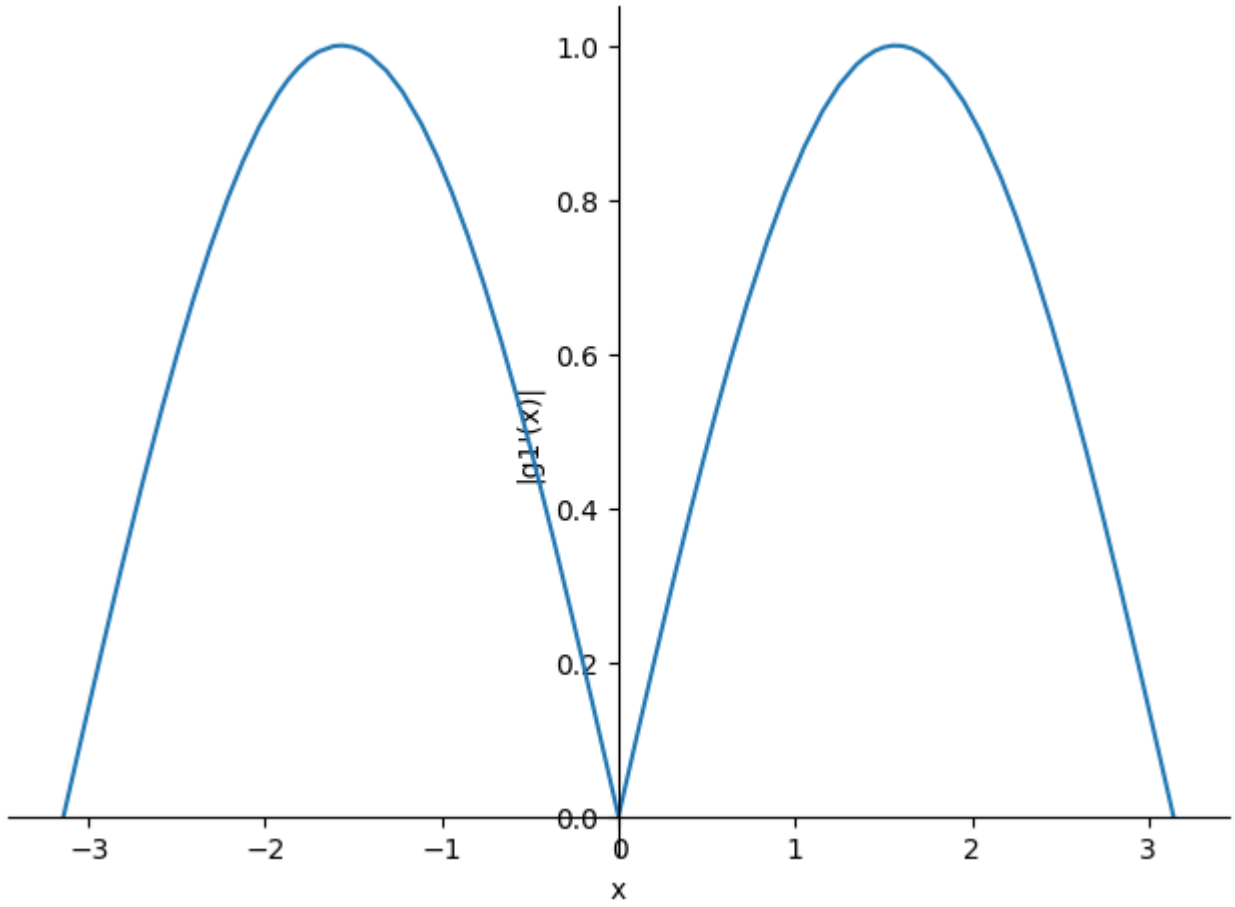
$$x = cos(x)$$

Tomando como función punto fijo

$$g_1(x) = cos(x)$$

Se verifica si la función cumple con las condiciones necesarias para ser punto fijo

```
x = sp.symbols('x')
#graficando el valor absoluto de la derivada de g(x)
plt(sp.Abs(sp.diff(sp.cos(x),x)), (x, -sp.pi, sp.pi), ylabel="|g1'(x)|", xlabel= 'x')
```




```
<sympy.plotting.plot.Plot at 0x9e3fdf0>
```

ya que $|g_1'(x)| < 1$ en un intervalo $[-\pi, \pi]$ se puede tomar $g_1(x) = cos(x)$ como función punto fijo

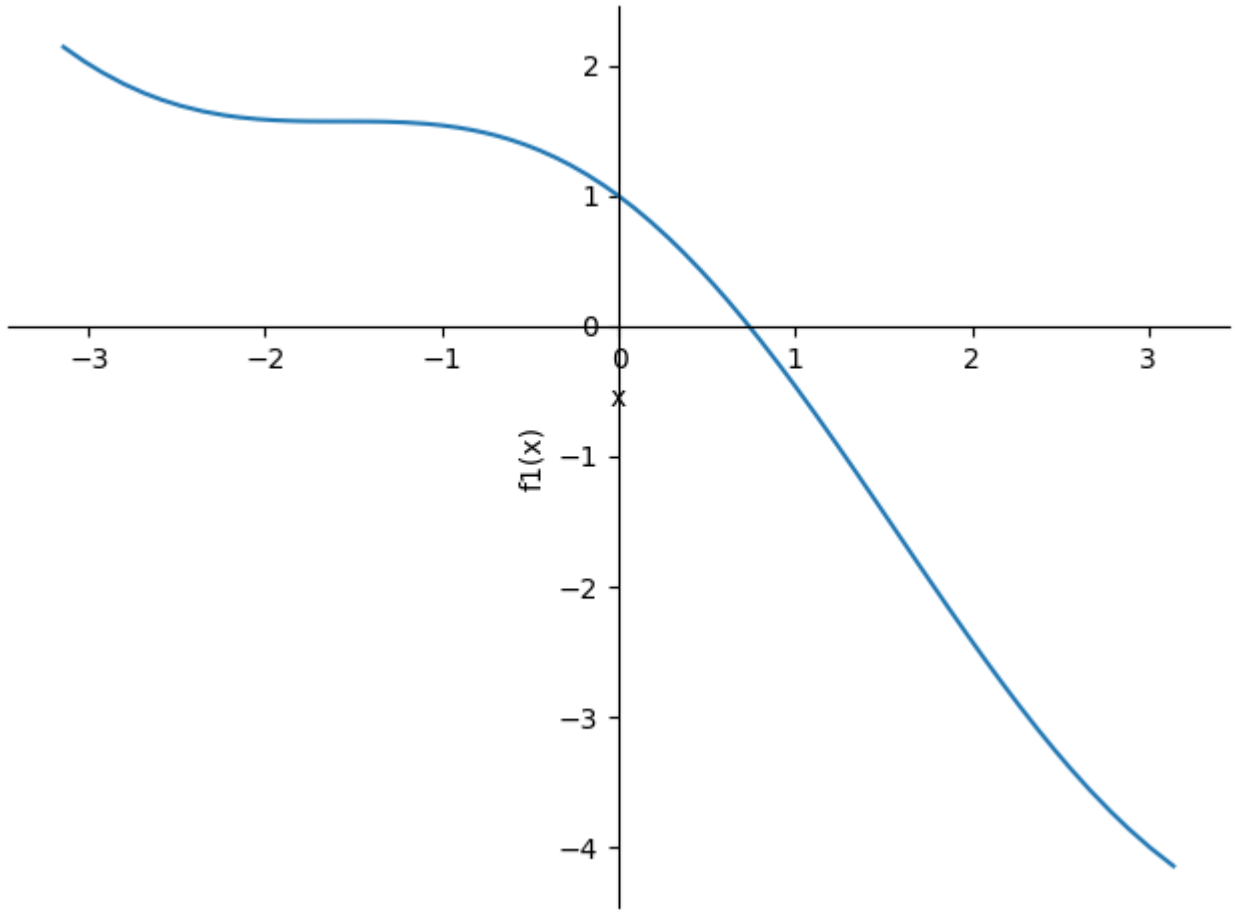
```
test.p0 = -0.5 #modificando aproximacion inicial
raizPuntoFijo = test.puntofijo(trigPuntoFijo, trigPuntoFijoPrima, -mt.pi, mt.pi)
raizPuntoFijo = test.p
print('raíz obtenida por el método de PuntoFijo:')
print('x = ', raizPuntoFijo)
print('f1(x) = ', trig(raizPuntoFijo))
```

```
i = 1 , p = 0.877582561890
i = 2 , p = 0.639012494165
i = 3 , p = 0.802685100682
i = 4 , p = 0.694778026788
i = 5 , p = 0.768195831282
i = 6 , p = 0.719165445942
i = 7 , p = 0.752355759422
i = 8 , p = 0.730081063138
i = 9 , p = 0.745120341351
i = 10, p = 0.735006309015
i = 11, p = 0.741826522643
i = 12, p = 0.737235725442
i = 13, p = 0.740329651878
i = 14, p = 0.738246238332
```

```
i = 15, p = 0.739649962770
i = 16, p = 0.738704539357
i = 17, p = 0.739341452281
i = 18, p = 0.738912449332
i = 19, p = 0.739201444136
i = 20, p = 0.739006779781
i = 21, p = 0.739137910762
i = 22, p = 0.739049580595
i = 23, p = 0.739109081421
i = 24, p = 0.739069001204
i = 25, p = 0.739095999836
i = 26, p = 0.739077813285
i = 27, p = 0.739090063988
i = 28, p = 0.739081811778
i = 29, p = 0.739087370571
i = 30, p = 0.739083626103
i = 31, p = 0.739086148423
i = 32, p = 0.739084449359
i = 33, p = 0.739085593869
i = 34, p = 0.739084822913
i = 35, p = 0.739085342238
i = 36, p = 0.739084992415
i = 37, p = 0.739085228060
i = 38, p = 0.739085069326
i = 39, p = 0.739085176251
i = 40, p = 0.739085104225
i = 41, p = 0.739085152743
i = 42, p = 0.739085120061
i = 43, p = 0.739085142076
i = 44, p = 0.739085127246
i = 45, p = 0.739085137236
i = 46, p = 0.739085130507
i = 47, p = 0.739085135040
i = 48, p = 0.739085131986
i = 49, p = 0.739085134043
i = 50, p = 0.739085132658
Iteraciones agotadas: Error!
raíz obtenida por el método de PuntoFijo:
x =  0.7390851326575361
f1(x) =  9.332472572509687e-10
```

corroborando

```
plt(sp.cos(x)-x, (x, -sp.pi, sp.pi), ylabel="f1(x)", xlabel= 'x')
```




```
<sympy.plotting.plot.Plot at 0xc72b850>
```

3.1.5 Resumen de resultados del problema test

	x_0	$f_1(x_0)$
Bisección	0.7390851331874728	4.633871064640971(10^{-11})
Newton	0.7390851332151607	0.0
Secante	0.7390851332151606	1.110223024625156(10^{-16})
Punto fijo	0.7390851326575361	9.332472572509687(10^{-10})

3.2 Problema 1

Resolver la ecuación:

$$e^x - 2 - x = 0, \ en\ el\ intervalo\ [-2.4; -1.0]$$

define la raíz de la funcion:

$$f_2(x) = e^x - 2 - x$$

```
#función dentro del programa funciones definida como exp(x)
```

```
#intancia utilizada para el problema 1
iteraciones = 50
tolerancia = 1e-9
aproxInic = 0.5
pro1 = zero(exp, expPrima, iteraciones, tolerancia, aproxInic)
```

3.2.1 Bisección

```
pro1.biseccion(-5,5) #prueba 1
raizBiseccion = pro1.p
print('raíz 1 obtenida por el método de bisección:')
print('x = ', raizBiseccion)
print('f2(x) = ', exp(raizBiseccion))
```

```
raíz 1 obtenida por el método de bisección:
x =  -1.8414056609617546
f2(x) =  4.41564562692065e-10
```

```
pro1.biseccion(0,5) #prueba 2
raizBiseccion = pro1.p
print('raíz 2 obtenida por el método de bisección:')
print('x = ', raizBiseccion)
print('f2(x) = ', exp(raizBiseccion))
```

```
raíz 2 obtenida por el método de bisección:
x =  1.1461932206293568
f2(x) =  1.8831158854482055e-11
```

3.2.2 Newton

```
pro1.p0 = 10 #cambiando aproximacion inicial
raizNewton = pro1.newton()
raizNewton = pro1.p
print('raíz 2 obtenida por el método de Newton:')
print('x = ', raizNewton)
print('f2(x) = ', exp(raizNewton))
```

```
Iteraciones agotadas: Error!
raíz 2 obtenida por el método de Newton:
x =  1.1461932206205825
f2(x) =  0.0
```

```
pro1.p0 = -10 #cambiando aproximacion inicial
raizNewton = pro1.newton()
raizNewton = pro1.p
print('raíz 1 obtenida por el método de Newton:'
```

```
Iteraciones agotadas: Error!
raíz 1 obtenida por el método de Newton:
x =  -1.8414056604369606
f2(x) =  0.0
```

3.2.3 Secante

```
#test.p0, test.p00 = -5, 5 #modificando aproximaciones iniciales
raizSecante = pro1.secante()
raizSecante = pro1.p
print('raíz 1 obtenida por el método de Secante:')
print('x = ', raizSecante)
print('f2(x) = ', exp(raizSecante))
```

```
raíz 1 obtenida por el método de Secante:
x =  1.1461932206205827
f2(x) =  2.220446049250313e-16
```

```
pro1.p0 = 0.1 #modificando aproximacion inicial
raizSecante = pro1.secante()
raizSecante = pro1.p
print('raíz 2 obtenida por el método de Secante:')
print('x = ', raizSecante)
print('f2(x) = ', exp(raizSecante))
```

```
raíz 2 obtenida por el método de Secante:
x =  1.1461932206205825
f2(x) =  0.0
```

3.2.4 Punto fijo

$$f_2(x) = e^x - x - 2 = 0$$

Despejando x

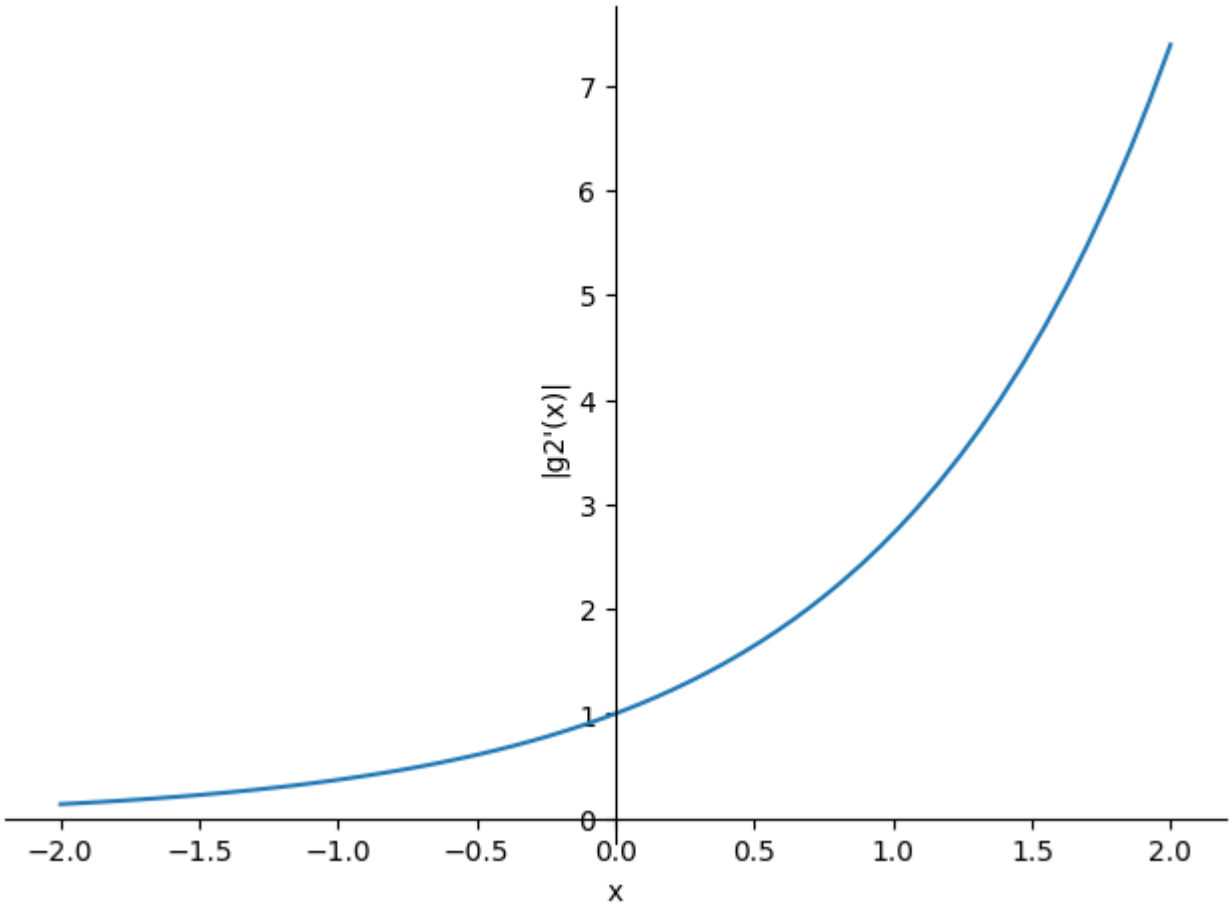
$$x = e^x - 2$$

Tomando como función punto fijo

$$g_2(x) = e^x - 2$$

Se verifica si la función cumple con las condiciones necesarias para ser punto fijo

```
#graficando el valor absoluto de la derivada de g(x)
plt(sp.Abs(sp.diff(sp.exp(x) - 2,x)), (x, -2, 2), ylabel="|g2'(x)|", xlabel= 'x')
```




```
<sympy.plotting.plot.Plot at 0xc7c33f0>
```

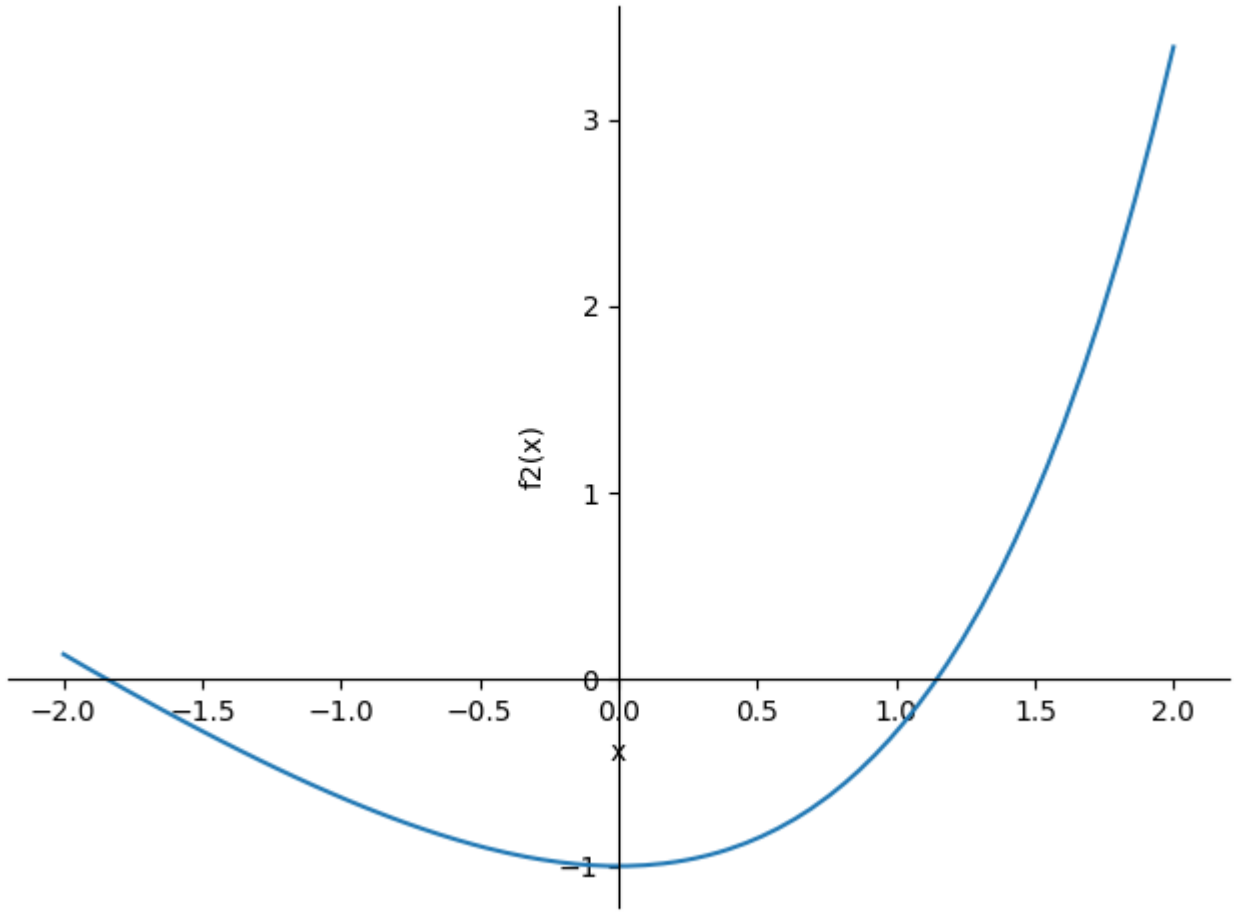
ya que $|g_2'(x)| < 1$ en un intervalo $[-\infty, 0)$ se puede tomar $g_2(x) = e^x - 2$ como función punto fijo. Sin embargo, solo nos permite obtener una sóla raíz

```
pro1.p0 = -0.5 #modificando aproximacion inicial
raizPuntoFijo = pro1.puntofijo(expPuntoFijo, expPuntoFijoPrima, -3, 0)
raizPuntoFijo = pro1.p
print('raíz 1 obtenida por el método de PuntoFijo:')
print('x = ', raizPuntoFijo)
print('f2(x) = ', exp(raizPuntoFijo))
```

```
raíz 1 obtenida por el método de PuntoFijo:
x =  -1.841405660285948
f2(x) =  -1.2706280472229992e-10
```

Corroborando resultados obtenidos

```
plt(sp.exp(x) - x - 2, (x, -2, 2), ylabel="f2(x)", xlabel= 'x')
```




```
<sympy.plotting.plot.Plot at 0xc7d71d0>
```

3.2.5 Resumen de resultados del problema test

	x_1	$f_2(x_1)$	x_2	$f_2(x_2)$
Bisección	-1.8414056608453393	3.436120277200416(10^{-10})	1.1461932206293568	1.8831158854482055(10^{-11})
Newton	-1.8414056604369606	0.0	1.1461932206205825	0.0
Secante	-1.8414056604369606	0.0	0.6859224077218748	-0.7003198815333824
Punto fijo	-1.841405660285948	-1.2706280472229992(10^{-10})	-	-

3.4 problema 3

determinar los valores de las fuentes E_1 y E_2 sabiendo que la potencia asociada a cada una es $W_1 = 2watt$ y $W_2 = 3watt$, considere la siguiente figura e inicie en el iterado $x_0 = [4, 4]^t$, escriba un sistema de ecuaciones con variables E_1 y E_2.

Imagen

Aplicando ley de mallas, se toma I_1 e I_2 como las corrientes que circulan por las baterias E_1 y E_2 respectivamente. I_1 se tomará en sentido horario e I_2 se tomará en sentido antiorario.

Para la malla 1

$$6I_1 + I_2 = E_1$$

sabiendo que en módulo $W_i = E_i I_i \rightarrow I_i = \frac{W_i}{E_i}$

$$\begin{aligned} 6I_1 + I_2 &= E_1 \\ 6\frac{W_1}{E_1} + \frac{W_2}{E_2} &= E_1 \\ 6W_1E_2 + W_2E_1 &= E_1^2E_2 \\ -E_1^2E_2 + 6W_1E_2 + W_2E_1 &= 0 \\ -E_1^2E_2 + 6(2)E_2 + 3E_1 &= 0 \\ -E_1^2E_2 + 3E_1 + 12E_2 &= 0 \qquad (1) \end{aligned}$$

Para la malla 1 (de la misma manera que la malla 1)

$$\begin{aligned} 7I_2 + I_1 &= E_2 \\ 7\frac{W_2}{E_2} + \frac{W_1}{E_1} &= E_2 \\ 7W_2E_1 + W_1E_2 &= E_1E_2^2 \\ -E_1E_2^2 + 7W_2E_1 + W_1E_2 &= 0 \\ -E_1E_2^2 + 7(3)E_1 + 2E_2 &= 0 \\ -E_1E_2^2 + 21E_1 + 2E_2 &= 0 \qquad (2) \end{aligned}$$

Resolviendo, a través del método de Newton, el sistema de ecuaciones no lineales conformado por (1) y (2):

$$\begin{cases} -E_1^2E_2 + 3E_1 + 12E_2 &= 0 \\ -E_1E_2^2 + 21E_1 + 2E_2 &= 0 \end{cases}$$

```
var = sp.symbols('x1 x2')
f1 = -var[0]**2*var[1] + 3*var[0] + 12*var[1]
f2 = -var[0]*var[1]**2 + 21*var[0] + 2*var[1] #seguir agregando funciones de ser necesario
F = [f1, f2]
x0 = [4, 4]
itera = 10
tol = 1e-6

E = SENL(x0, F, itera, tol)
print('E1 = ',E[0])
print('E2 = ',E[1])
```

```
E1 =  3.786864141330596
E2 =  4.854248657521035
```

```
#comprobación
F0 = sp.lambdify([i for i in var], sp.matrices.Matrix(F))
F0(E[0],E[1])
```

```
array([[ 7.10542736e-15],
       [-2.48689958e-14]])
```

3.4.1 Resultados problema 3

$E_1[V]$	3.786864141330596
$E_2[V]$	4.854248657521035

4. Análisis de resultados

En el problema 1 se puede observar que las raíces obtenidas con los métodos de Newton y de la secante fueron las más cercanas a cero al evaluarla en la función de interés, incluso cabe resaltar que el método de Newton después de acabar las respectivas iteraciones establecidas alcanza el cero absoluto definido por el programa. No obstante, las demás aproximaciones también pueden tomarse en cuenta, dependiendo de la aplicación se puede despreciar la diferencia con respecto al cero. Cabe señalar que el método de bisección es considerado el más lento de todas las formas establecidas.

En el problema 2, en la función, es notable que se establacieran dos raíces distintas; debido a ésto ocurrieron algunos eventos partículares que se describirán a continuación. El método de bisección proporcionó una buena aproximación en el orden de 10^{-10} cifras decimales; tomando un intervalo de una longitud de 10 unidades se aproximó hasta el cero más a la izquierda dentro del intervalo, en consecuencia, fue necesario dejar este punto fuera del intervalo de interés para obtener una aproximación del cero más a la derecha; por tanto para múltiples raíces dentro de un intervalo no es recomendable utlizar bisección. El método de Newton sigue siendo igual de efectivo que en el problema 1, siempre y cuando la aproximación inicial esté cerca del cero de interés. Con el método de la secante no fue posible obtener la raíz 2, esto puede deberse a que una de las aproximaciones que se utilizan estaba relativamente cerca, ya que con el cero negativo no hubo problemas e incluso se realizó una excelente aproximación a la raíz. Por último al utilizar el método del punto fijo quedó demostrado que con una sola función de punto fijo no es posible obtener todos los ceros de una función, ésto debido a las condiciones particulares que debe cumplir la función para la convergencia del método en sí.

Al resolver el sistema de ecuaciones no lineales del problema 3 por el método numérico de Newton se estableció que el vector de funciones F era aproximadamente cero por 10^{-14} cifras decimales significativas, además al estar analizando un sistema de potencia el orden de los $picowatts$ puede considerarse despreciable en cuanto a cifras significativas. Por tanto, las tensiones

obtenidas en la última tabla se adecuan perfectamente al sistema, sin embargo estos valores exactos no se pueden considerar en una bateria debido al deterioro de la misma y la incertidumbre que ésta poseé.

5. Conclusión

Se concluye que el método de Newton para hallar los ceros de una funcion es el más efectivo de todos, debido a su aproximación a cero presentada en las tablas de rsultados, el problema se podría presentar si en algunos casos la derivada de la funcion se vuelve complicada de calcular, por lo que seria útil utilizar otro método, además es necesario considerar una cercana aproximación inicial. En cuanto a los sistemas de ecuaciones no lineales se puede decir que despues de ciertas iteraciones llega a converjer a un resultado en el orden 10^{-14} de los decimales, que para aplicaciones de potencia, como es el caso, puede llegar a considerarse cero; se recuerda siempre tomar en cuenta que el numero de ecuaciones e incognitas deben ser iguales para que los sistemas tengan una solución.

6. Referencias