UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE INGENIERÍA

Universidad Autónoma de Querétaro

Facultad de Ingeniería

Métodos cerrados

Análisis numérico

Autor:
David Gómez Torres

Agosto 2021

ÍNDICE GENERAL

1. Introducción **1**

1.1. Métodos gráficos 1
1.2. Método de la bisección 2
1.3. Criterios de paro y estimaciones de error 2
1.4. Método de la falsa posición 3
1.4.1. Falsa posición modificada 3

2. Metodología **4**

2.1. Raíces de polinomio de 2 grado 4
2.2. Raíz de polinomio de 5to grado 7
2.3. Raíz de polinomio de 1er grado 9
2.4. Raíz de una incógnita elevada a una potencia racional 11
2.5. Velocidad de un paracaidista 13

3. Resultados **15**

4. Anexos **18**

4.1. Anexo A: Evidencia del funcionamiento de los código reportados 18

5. Bibliografía **20**

CAPÍTULO 1

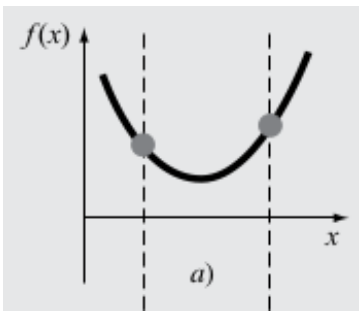
INTRODUCCIÓN

Antes de presentar un serie de ejercicios resueltos y reportar el código con el que fueron resueltos, a continuación se hace un resumen de las páginas 96-109 del libro [**1**].

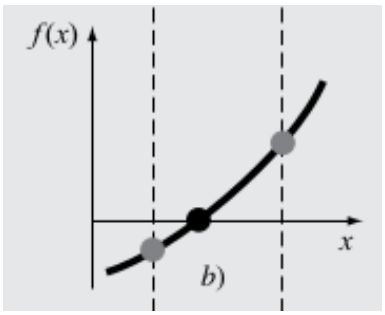
1.1. Métodos gráficos

Un método simple para obtener una aproximación a la raíz de la ecuación $f(x) = 0$ consiste en graficar la función y observar dónde cruza el eje x. Este punto, que representa el valor de x para el cual $f(x) = 0$, ofrece una aproximación inicial de la raíz. Las técnicas gráficas tienen un valor práctico limitado, ya que no son precisas. Sin embargo, los métodos gráficos se utilizan para obtener aproximaciones de la raíz.

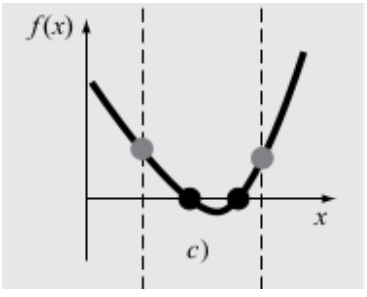
La siguiente figura muestra algunas de las formas en las que la raíz puede encontrarse (o no encontrarse) en un intervalo definido por un límite inferior x_l y un límite superior x_u.



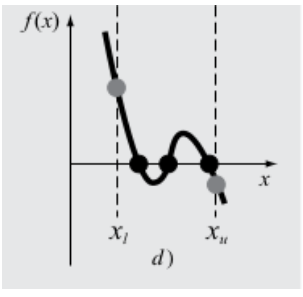
(a) Sin ninguna raíz real



(b) 1 solución real



(c) Número par de raíces



(d) Número impar de raíces

Figura 1.1: Generalizaciones para encontrar raíces

1.2. Método de la bisección

El método de bisección, conocido también de partición de intervalos o de Bolzano, es un tipo de búsqueda incremental en el que el intervalo se divide siempre a la mitad. Si la función cambia de signo sobre un intervalo, se evalúa el valor de la función en el punto medio. La posición de la raíz se determina situándola en el punto medio del subintervalo, dentro del cual ocurre un cambio de signo. El proceso se repite hasta obtener una mejor aproximación. Ver figura (1.2).

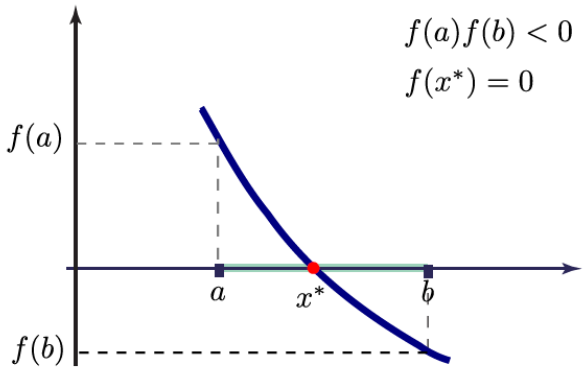


Figura 1.2: Representación gráfica del método de la bisección

1.3. Criterios de paro y estimaciones de error

Una sugerencia inicial sería finalizar el cálculo cuando el error verdadero se encuentre por debajo de algún nivel prefijad, sin embargo, dicha estrategia es inconveniente, ya que la estimación del error se basa en el conocimiento del valor verdadero de la raíz de la función. Por lo tanto, necesitamos estimar el error de forma tal que no se necesite el conocimiento previo de la raíz y entonces definimos el error relativo porcentual e_a:

$$e_a = \frac{x_r^{nuevo} - x_r^{anterior}}{x_r^{nuevo}} 100\,\% \tag{1.1}$$

sí cuando e_a es menor que un valor previamente fijado e_s, termina el cálculo.

De la misma manera, debido a que en cada iteración se reduce el error a la mitad, la fórmula general que relaciona el error y el número de iteraciones, n, es

$$E_a^n = \frac{\Delta x^0}{2} \tag{1.2}$$

Si E_a^d es el error deseado, en esta ecuación se despeja

$$\begin{aligned} n &= \frac{\log(\Delta x^0 / E_{a,d})}{\log 2} \\ &= \log 2 \left(\frac{\Delta x^0}{E_{a,d}} \right) \end{aligned} \tag{1.3}$$

La ecuación (1.3) nos da la relación para el número de iteraciones necesarias dada un error deseable.

1.4. Método de la falsa posición

Aun cuando la bisección es una técnica perfectamente válida para determinar raíces, su método de aproximación por "fuerza bruta" es relativamente ineficiente. La falsa posición es una alternativa basada en una visualización gráfica.
Un inconveniente del método de bisección es que al dividir el intervalo de x_l a x_u en intervalos iguales, no se toman en consideración las magnitudes de $f(x_l)$ y $f(x_u)$.

Un método alternativo que aprovecha esta visualización gráfica consiste en unir $f(x_l)$ y $f(x_u)$ con una línea recta. La intersección de esta línea con el eje de las x representa una mejor aproximación de la raíz. El hecho de que se reemplace la curva por una línea recta da una *"falsa posición"* de la raíz; también conocido como *"interpolación lineal"*. Ver figura (1.3).

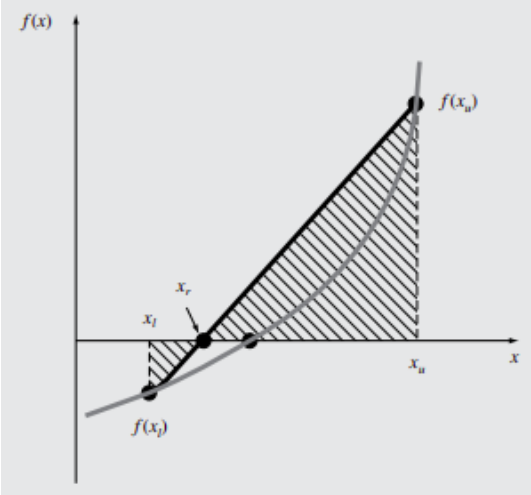


Figura 1.3: Representación gráfica del método de la falsa posición

Usando triángulos semejantes,la intersección de la línea recta con el eje de las x se estima mediante

$$x_r = x_u - \frac{f(x_u)(x_l - x_u)}{f(x_l) - f(x_u)} \tag{1.4}$$

1.4.1. Falsa posición modificada

Una forma de disminuir la naturaleza unilateral de la falsa posición consiste en obtener un algoritmo que detecte cuando se "estanca" uno de los límites del intervalo. Si ocurre esto, se divide a la mitad el valor de la función en el punto de "estancamiento". A este método se le llama *método de la falsa posición modificado.*

En este método se han usado contadores para determinar si uno de los límites del intervalo permanece fijo "estancado" durante dos iteraciones. Si ocurre así, el valor de la función en este valor de "estancamiento" se divide a la mitad.

CAPÍTULO 2

METODOLOGÍA

2.1. Raíces de polinomio de 2 grado

Problema 5.1

Determine las raíces reales de $f(x) = 0{,}5x^2 + 2{,}5x + 4{,}5$:

1. Gráficamente.
2. Empleando la fórmula cuadrática.
3. Usando el método de bisección con tres iteraciones para determinar la raíz más grande. Emplee como valores iniciales $x_l = 5$ y $x_u = 10$. Calcule el error estimado e_a y el error verdadero e_t para cada iteración.

1. Gráficamente

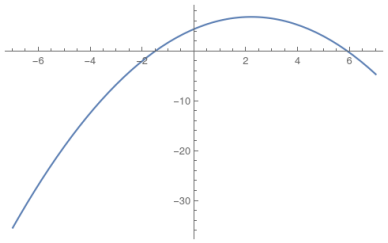
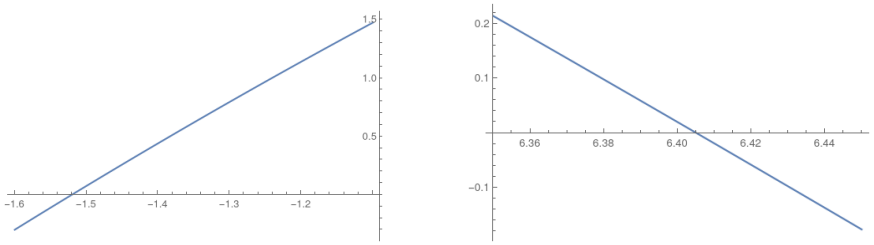


Figura 2.1: Panorama general de las raíces



(a) Raíz menor (b) Raíz mayor

Figura 2.2: Raíces por el método gráfico

2. Mediante la formula cuadrática

```
#programa para calcular las raices de un polinomio cuadratico

import math

#pedir variables
a = float(input("Ingrese el valor del termino cuadratico: "))
b = float(input("Ingrese el valor del termino lineal: "))
c = float(input("Ingrese el valor de la constante: "))

#operaciones

discri = b**2 - 4 * a * c


#bloque if

if (discri > 0 ):
    x1 = ( -b + math.sqrt(discri) ) / (2 * a )
    x2 = ( -b - math.sqrt(discri) ) / (2 * a )

    print ("La primer solucion real X1 es: ") + str(x1)
    print ("La segunda solucion real X2 es: ") + str(x2)

elif (discri < 0):
    real = (-b ) /  ( 2 * a )
    imag = math.sqrt( abs(discri) ) / (2 * a )

    print ("La primera solucion imaginaria es: ") + str(real) +
    " " +  str(imag) + "+i"
    print ("La segunda solucion imaginaria es: ") + str(real) +
    " " +  str(imag) + "-i"
elif (discri == 0) :
    x3 = (-b) / (2 * a)
    print ("Solo existe una solucion: ") + str(x3)
```

3. Mediante el método de bisección

```
def f(x):
    return 0.5*x**2+2.5*x+4.5

def Metodo_Biseccion(f,xl,xu,ea):

    m1 = xl
    xr = xu
    iter = 0

    while(abs(m1-m)>ea):
        m1 = m
        xr = (xl+xu)/2
        if(f(xl)*f(xr)<0):
            xu = xr
        if(f(xr)*f(xu)<0):
            xl = xr
        iter = iter + 1
        es = abs(m1-xr)
        print("Iteracion",iter," Raiz:",xr," Error aprox:",es)

Metodo_Biseccion(f,5,10,1e-6)
```

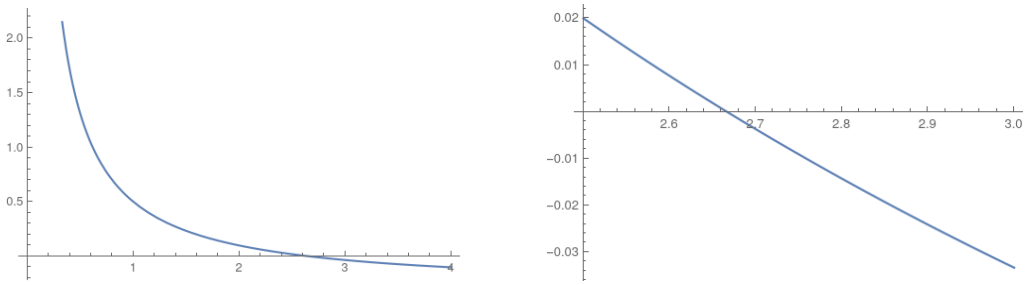
2.2. Raíz de polinomio de 5to grado

Problema 5.3

Determine las raíces reales de $f(x) = 25 + 82x - 90x^2 + 44x^3 - 8x^4 + 0{,}7x^5$:

1. Gráficamente.
2. Usando el método de bisección para localizar la raíz más grande con $e_s = 10\,\%$. Utilice como valores iniciales $x_l = 0.5$ y $x_u = 1.0$.
3. Realice el mismo cálculo que en 2), pero con el método de la falsa posición y $e_s = 0.2\,\%$.

1. Graficamente



(a) Panorama de la función (b) Ampliación

Figura 2.3: Raíces por el método gráfico

2.

```
def f(x):
    return -25+82*x-90*x**2+44*x**3-8*x**4+0.7*x**5

def Metodo_Biseccion(f,xl,xu,ea):

    m1 = xl
    xr = xu
    iter = 0

    while(abs(m1-m)>ea):
        m1 = m
        xr = (xl+xu)/2
        if(f(xl)*f(xr)<0):
            xu = xr
        if(f(xr)*f(xu)<0):
            xl = xr
        iter = iter + 1
        es = abs(m1-xr)
        print("Iteracion",iter," Raiz:",xr," Error aprox:",es)

Metodo_Biseccion(f,0.5,1,10)
```

3.

```
import numpy as np

def f(x):
    return -25+82*x-90*x**2+44*x**3-8*x**4+0.7*x**5


def Metodo_Falsa(f,xl,xu,es,imax):
    iter = 0
    fl = f(xl)
    fu = f(xu)
    ea = 0
    xr = 0
    il = 0
    while (ea < es or iter >= imax):
        xrold = xr
        xr = xu - fu * (xl - xu) / (fl - fu)
        fr = f(xr)
        iter = iter + 1
        if (xr < 0 or xr > 0):
            ea = abs((xr - xrold) / xr) * 100

        test = fl * fr
        if(test < 0):
            xu = xr
            fu = f(xu)
            iu = 0
            il = il + 1
            if (il >= 2):
                fl = fl / 2
            elif (test > 0):
                xl = xr
                fl = f(xl)
                il = 0
                iu = iu + 1
                if (iui >= 2):
                    fu = fu / 2
            else:
                ea = 0
        print("Iteracion",iter," Raiz:",xr," Error:",ea)
    Metodo_Falsa = xr
Metodo_Falsa(f,0.5,1,0.2,100)
```

2.3. Raíz de polinomio de 1er grado

Problema 5.7

Determine la raíz real de $f(x) = (0{,}8 - 0{,}3x)/x$:

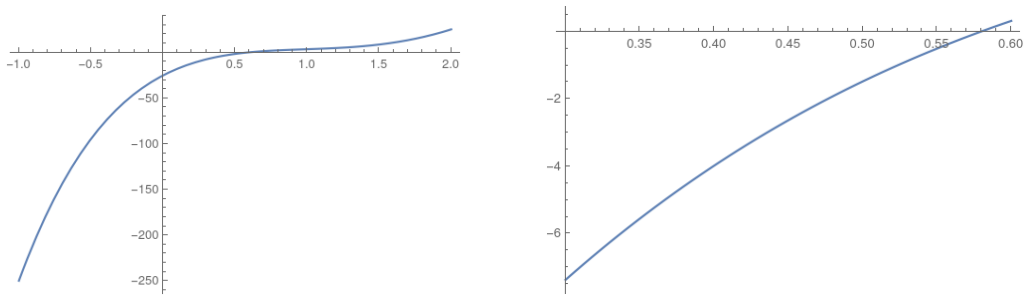
1. Analíticamente.
2. Gráficamente.
3. Empleando tres iteraciones en el método de la falsa posición, con valores iniciales de 1 a 3, calcule el error aproximado e_a y el error verdadero e_t en cada iteración. ¿Hay algún problema con el resultado?

1. Para encontrar la raíz de la función, hemos de notar que se trata de aquellos puntos donde la función $f(x) = 0$ y la manera determinarlos será igualando a cero el numerador de la función

$$\begin{aligned} 0{,}08 - 0{,}3x &= 0 \\ -0{,}3x &= -0{,}08 \\ x &= \frac{-0{,}08}{-0{,}3} \end{aligned}$$

   Lo cual es, aproximadamente, $x = 2.\bar{6}$

2. Graficamente



(a) Panorama de la función (b) Ampliación

Figura 2.4: Raíces por el método gráfico

3.

```
import numpy as np

def f(x):
    return 0.8*x-0.3


def Metodo_Falsa(f,xl,xu,es,imax):
    iter = 0
    fl = f(xl)
    fu = f(xu)
    ea = 0
    xr = 0
    il = 0
    while (ea < es or iter >= imax):
        xrold = xr
        xr = xu - fu * (xl - xu) / (fl - fu)
        fr = f(xr)
        iter = iter + 1
        if (xr < 0 or xr > 0):
            ea = abs((xr - xrold) / xr) * 100
            et = 0.375 - xr
        test = fl * fr
        if(test < 0):
            xu = xr
            fu = f(xu)
            iu = 0
            il = il + 1
            if (il >= 2):
                fl = fl / 2
            elif (test > 0):
                xl = xr
                fl = f(xl)
                il = 0
                iu = iu + 1
                if (iui >= 2):
                    fu = fu / 2
            else:
                ea = 0
        print("Iteracion",iter," Raiz:",xr," Error aprox:",ea,
        " Error verdad:",et)
    Metodo_Falsa = xr

Metodo_Falsa(f,1,3,0.2,3)
```

2.4. Raíz de una incógnita elevada a una potencia racional

Problema 5.11

Determine la raíz real de $x^{3,5} = 80$:

1. En forma analítica.
2. Con el método de la falsa posición dentro de $e_s = 2.5\,\%$. Haga elecciones iniciales de 2.0 a 5.0.

1. Para encontrar la raíz de la ecuación

$$x^{3,5} = 80$$

debemos despejar la incógnita, y para ello, utilizamos las propiedades de los logaritmos

$$\begin{aligned} &=> \quad \log_{3,5}(x^{3,5}) = \log_{3,5}(80) \\ &=> \quad x = \log_{3,5}(80) \end{aligned}$$

Lo cual nos da como resultado, aproximadamente, $x = 3{,}497889$

2.

```
import numpy as np

def f(x):
    return x**(3.5)-80


def Metodo_Falsa(f,xl,xu,es,imax):
    iter = 0
    fl = f(xl)
    fu = f(xu)
    ea = 0
    xr = 0
    il = 0
    while (ea < es or iter >= imax):
        xrold = xr
        xr = xu - fu * (xl - xu) / (fl - fu)
        fr = f(xr)
        iter = iter + 1
        if (xr < 0 or xr > 0):
            ea = abs((xr - xrold) / xr) * 100
            et = 0.375 - xr
        test = fl * fr
        if(test < 0):
            xu = xr
            fu = f(xu)
```

```
                iu = 0
                il = il + 1
                if (il >= 2):
                    fl = fl / 2
                elif (test > 0):
                    xl = xr
                    fl = f(xl)
                    il = 0
                    iu = iu + 1
                    if (iui >= 2):
                        fu = fu / 2
                else:
                    ea = 0
            print("Iteracion",iter," Raiz:",xr," Error aprox:",ea,
            " Error verdad:",et)
    Metodo_Falsa = xr

Metodo_Falsa(f,2,5,2.5,100)
```

2.5. Velocidad de un paracaidista

Problema 5.13

La velocidad u de un paracaidista que cae está dada por

$$v = \frac{gm}{c}(1 - e - (c/m)t)$$

donde g = 9.81 m/s 2 . Para un paracaidista con coeficiente de resistencia de c = 15 kg/s, calcule la masa m de modo que la velocidad sea y = 36 m/s en t = 10 s. Utilice el método de la falsa posición para determinar m a un nivel de $e_s = 0.1\,\%$.

```
import numpy as np

def f(x):

    #return (9.8*x/15)*(1-np.exp(150/x))-35

def Metodo_Falsa(f,xl,xu,es,imax):
    iter = 0
    fl = f(xl)
    fu = f(xu)
    ea = 0
    xr = 0
    il = 0
    while (ea < es or iter >= imax):
        xrold = xr
        xr = xu - fu * (xl - xu) / (fl - fu)
        fr = f(xr)
        iter = iter + 1
        if (xr < 0 or xr > 0):
            ea = abs((xr - xrold) / xr) * 100
            et = 0.375 - xr
        test = fl * fr
        if(test < 0):
            xu = xr
            fu = f(xu)
            iu = 0
            il = il + 1
            if (il >= 2):
                fl = fl / 2
            elif (test > 0):
                xl = xr
                fl = f(xl)
                il = 0
                iu = iu + 1
                if (iui >= 2):
                    fu = fu / 2
```

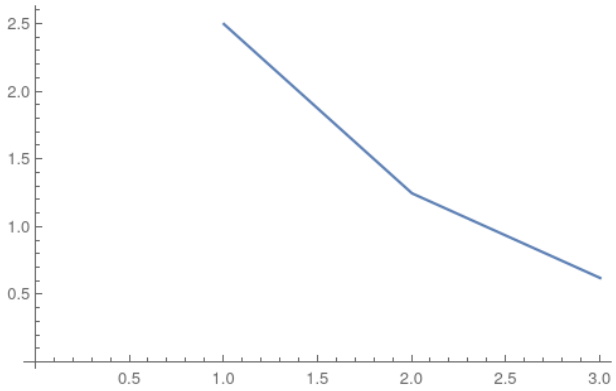
```
37                else:
38                    ea = 0
39            print("Iteracion",iter," Raiz:",xr," Error aprox:",ea)
40        Metodo_Falsa = xr
41
42    Metodo_Falsa(f,59,60,0.1,100)
```

CAPÍTULO 3

RESULTADOS

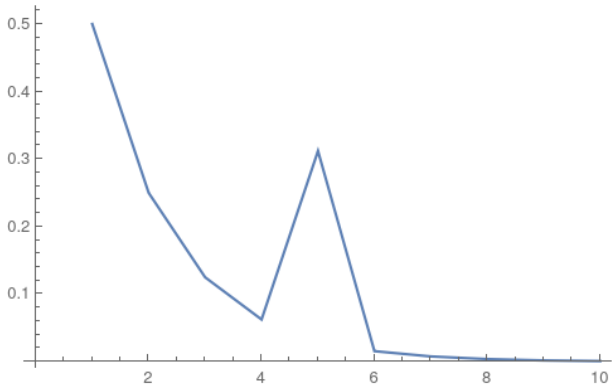
Ejercicio 5.1

Iteración	Error relativo porcentual
1	2.5
2	1.25
3	0.625



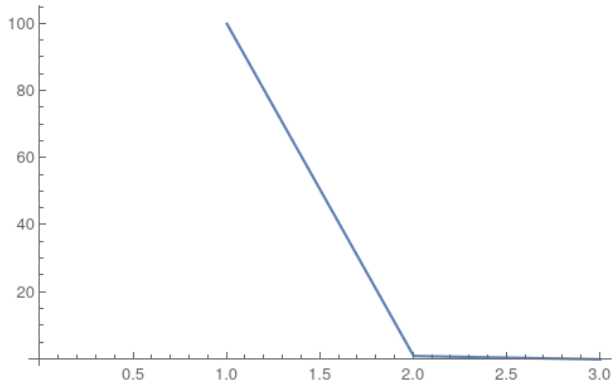
Ejercicio 5.3

Iteración	Error relativo porcentual
1	0.5
2	0.25
3	0.125
4	0.0625
5	0.0312
6	0.0156
7	0.0078
8	0.0039
9	0.0019
10	0.0009



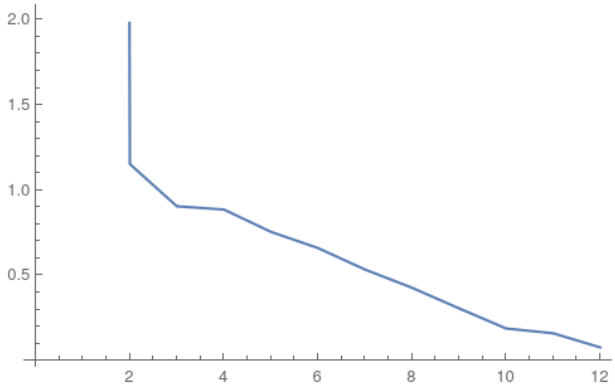
Ejercicio 5.7

Iteración	Error relativo porcentual
1	100
2	1.002
3	0.006



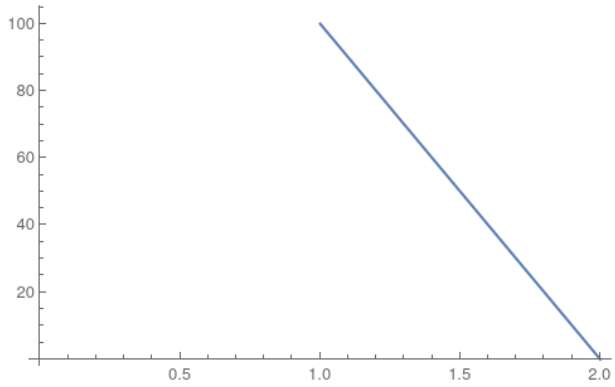
Ejercicio 5.11

Iteración	Error relativo porcentual
1	100
2	1.154
3	0.906
4	0.887
5	0.756
6	0.661
7	0.535
8	0.428
9	0.307
11	0.189
12	0.161
13	0.078



Ejercicio 5.13

Iteración	Error relativo porcentual
1	100
2	0.0148



CAPÍTULO 4

ANEXOS

4.1. Anexo A: Evidencia del funcionamiento de los código reportados

Ejercicio 5.1

Figura 4.1: Evidencia del funcionamiento

Ejercicio 5.3

Figura 4.2: Evidencia del funcionamiento

Ejercicio 5.7

Figura 4.3: Evidencia del funcionamiento

```
ezequiel@ezequiel-HP-Pavilion-15-Notebook-PC:~/Ingenieria Fisica/4 semestre/Metodos Numericos/Tarea 2$ python3 biseccion.py
Iteracion 1  Raiz: 0.5  Error aprox: 0.5
Iteracion 2  Raiz: 0.75  Error aprox: 0.25
Iteracion 3  Raiz: 0.625  Error aprox: 0.125
Iteracion 4  Raiz: 0.5625  Error aprox: 0.0625
Iteracion 5  Raiz: 0.59375  Error aprox: 0.03125
Iteracion 6  Raiz: 0.578125  Error aprox: 0.015625
Iteracion 7  Raiz: 0.5859375  Error aprox: 0.0078125
Iteracion 8  Raiz: 0.58203125  Error aprox: 0.00390625
Iteracion 9  Raiz: 0.580078125  Error aprox: 0.001953125
Iteracion 10  Raiz: 0.5791015625  Error aprox: 0.0009765625
Iteracion 11  Raiz: 0.57958984375  Error aprox: 0.00048828125
Iteracion 12  Raiz: 0.579345703125  Error aprox: 0.000244140625
Iteracion 13  Raiz: 0.5794677734375  Error aprox: 0.0001220703125
Iteracion 14  Raiz: 0.5794067382812 5  Error aprox: 6.1035156250e-05
Iteracion 15  Raiz: 0.5794372558593 75  Error aprox: 3.0517578125e-05
Iteracion 16  Raiz: 0.5794219970703125  Error aprox: 1.525878906250e-05
Iteracion 17  Raiz: 0.5794143676757812  Error aprox: 7.629394531250e-06
Iteracion 18  Raiz: 0.5794105529785156  Error aprox: 3.814697265625e-06
Iteracion 19  Raiz: 0.5794086456298828  Error aprox: 1.907348632812 5e-06
Iteracion 20  Raiz: 0.5794095993041992  Error aprox: 9.536743164062 5e-07
ezequiel@ezequiel-HP-Pavilion-15-Notebook-PC:~/Ingenieria Fisica/4 semestre/Metodos Numericos/Tarea 2$
```

Ejercicio 5.11

Figura 4.4: Evidencia del funcionamiento

```
ezequiel@ezequiel-HP-Pavilion-15-Notebook-PC:~/Ingenieria Fisica/4 semestre/Metodos Numericos/Tarea 2$ python3 falsa_pos.py
Iteracion 1  Raiz: 2.768317965873622  Error aprox: 100.0  Error verdad: -2.393317965873622
ezequiel@ezequiel-HP-Pavilion-15-Notebook-PC:~/Ingenieria Fisica/4 semestre/Metodos Numericos/Tarea 2$
```

Ejercicio 5.13

Figura 4.5: Evidencia del funcionamiento

```
ezequiel@ezequiel-HP-Pavilion-15-Notebook-PC:~/Ingenieria Fisica/4 semestre/Metodos Numericos/Tarea 2$ python3 falsa_pos.py
Iteracion 1  Raiz: 0.375  Error aprox: 100.0  Error verdad: 0.0
ezequiel@ezequiel-HP-Pavilion-15-Notebook-PC:~/Ingenieria Fisica/4 semestre/Metodos Numericos/Tarea 2$
```

CAPÍTULO 5

BIBLIOGRAFÍA

1. Chapra, S. C., Canale, R. P. (2011). *Numerical methods for engineers* (Vol. 1221). New York: Mcgraw-hill.