Printout

Wednesday, September 25, 2024 8:42 AM

Section 1

MANE 3351

Lecture 9

Classroom Management

Agenda

- Part 2: Bisection Search Error Analysis
- False Position Method
- Test 1
- Laboratory Session 9 covers Lab 4

Resources

Handouts

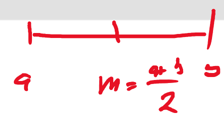
- Lecture 9 Slides
- Lecture 9 Marked Slides
- Test 1 from 2023

Assignments

- Homework 2 (assigned 9/18/2024, due 9/25/2024 (before 11:59 pm - no late submissions)
- Lab 4 (assigned 9/25/2024, due 10/2/2024 (before 2:00 pm))
- Read textbook pages 71 - 74

Schedule

Lecture/Lab	Date	Topic
10	9/27	Bisection Method Error Analysis, False Position (not on Test 1), lab session
11	10/2	Test 1 (lectures 1-7); no lab



Bisection Method: Error Analysis

Bisection Method Theorem

Cheney and Kincaid (2004)[^1] provide a definition of the bisection method

> *If the bisection algorithm is applied to a continuous function on an interval $[a, b]$, where $f(a)f(b) < 0$, then, after n steps, an approximate root will have been computed with error at most $(b - a)/2^{n+1}$.*

This definition provides the following useful results:

- $|e_n| \leq \frac{1}{2^{n+1}}(b - a)$
- $n > \frac{\log(b-a) - \log 2\varepsilon}{\log 2}$

ε - epsilon - tolerance

lg = log₁₀

Percent Relative Error

Chapra and Canale (2015) [^2] provide a formula for the approximate percent relative error

- $\varepsilon_a = \left| \frac{x_r^{new} - x_r^{old}}{x_r^{new}} \right| \times 100\%$ where x_r^{new} is the root for the present iteration and x_r^{old} is the root from the previous iteration

False Position

- Attempts to converge faster than bisection method by using functional information
- Referred to as the bracketed secant method in textbook
- The new root is found by

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

- Figure 5.12 (Chapra and Canale(2015)[^2] illustrates the false position method

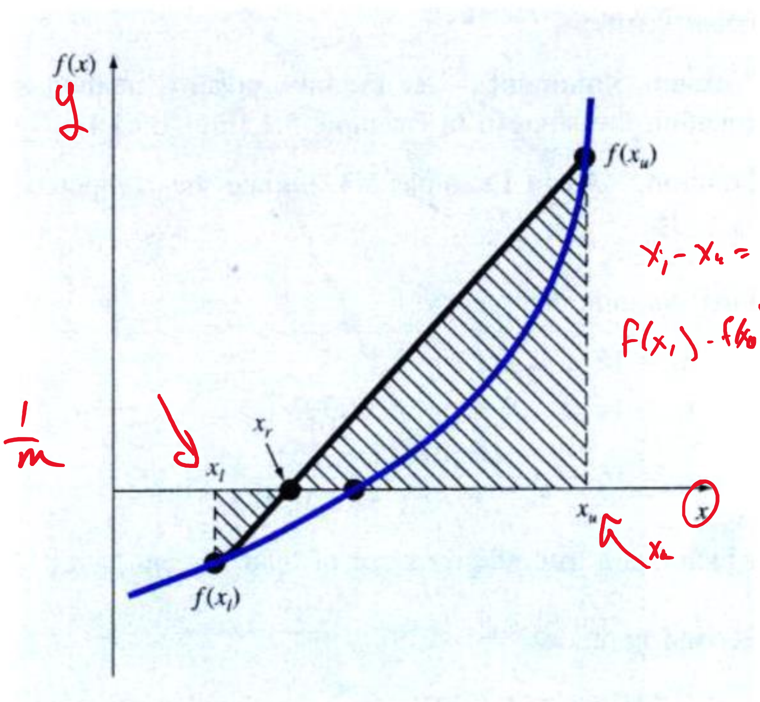


FIGURE 5.12
A graphical depiction of the method of false position. Similar triangles used to derive the formula for the method are shaded.

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

$$= x_u - f(x_u)\left(\frac{\Delta x}{\Delta y}\right) = \frac{1}{m}$$

$$\text{slope} = \frac{\Delta \text{rise}}{\Delta \text{run}} \rightarrow \frac{\Delta y}{\Delta x}$$

False Position Code

The pseudocode, shown below was taken from [^3]

Complete Pseudocode for Regula Falsi or False Position Method

```
1. Start

2. Define function f(x)

3. Input
      a. Lower and Upper guesses x0 and x1
      b. tolerable error e

4. If f(x0)*f(x1) > 0
      print "Incorrect initial guesses"
      goto 3
   End If

5. Do

      x2 = x0 - ((x0-x1) * f(x0))/(f(x0) - f(x1))

      If f(x0)*f(x2) < 0
              x1 = x2
      Else
              x0 = x2
      End If

   While abs(f(x2) > e
```

False Position Error Analysis

- Much harder to analyze than bisection method
- Convergence to root depends on shape of $f(x)$
- At times, false position may converge slowly!, Example from textbook[^4], shown below

Figure 2.7.1: A troublesome function for the bracketed secant method.

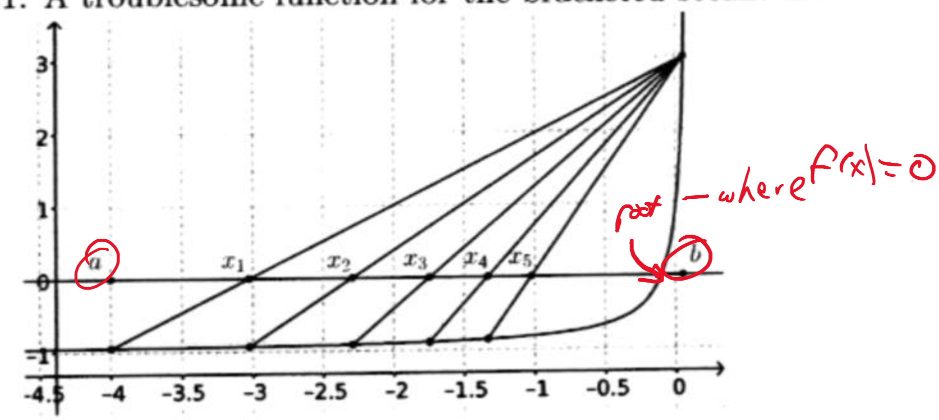


Figure 2: False Position troublesome function

Loops in Python

- Python only supports two types of loops: for and while
- Pseudocode used do/while loop

Repeat/until Loop

Python Code for False Position

```
# False Position
from scipy import stats
import math
def f(x):
    return stats.norm.cdf(x)-.25
# input a & b
while True:
    a=float(input("enter the lower bound: "))
    b=float(input("enter the upper bound: "))
    if f(a)*f(b)<0.0:
        break
    print("incorrect bounds, please re-enter")
print("the lower bound is {}".format(a))
print("the upper bound is {}".format(b))
counter=0
# process loop
while True:
```

Test One

- You are allowed one 4 inch by 6 inch notecard containing handwritten notes
- Calculator is needed; no programmable calculators that can calculate derivatives
- Covers material from Lectures 1 - 7
- Review Homework 1 and 2 and solutions
- Review Test 1
- Resistor information will be provided

References

[1]: Cheney, W. and Kincaid, D. (2004), *Numerical Mathematics and Computing, 5th edition*
[2]: Chapra, S. and Canale, R. (2015), *Numerical Methods for Engineering, 7th edition*
[4]: Brin, L., (2020), *Tea Time Numerical Analysis: Experiences in Mathematics, 3rd edition*