4. Bisection Method

Cheng Peng

Lecture Note for MAT325 Numerical Analysis

Contents

1 Introduction

There are different methods for root finding. The bisection method discussed in this note is useful for finding a root of single variable functions that satisfy certain assumptions.

2 The Question

The general question to answer: let $f(x)$ be a single variable continuous function defined on D, a sub set of R. Assume there is at least one root for equation $f(x) = 0$ on the closed interval [a, b].

Goal: using a numerical method to locate the root.

Several methods can be used to find the root of a given single variable equation satisfying certain regular conditions in the next few lectures. Most of these methods are direct applications of some concepts we have already covered in elementary calculus courses.

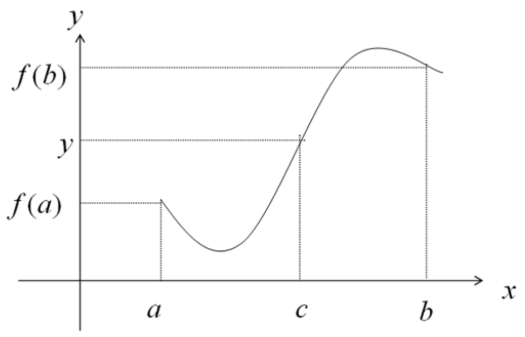
We follow the same process throughout the semester to do numerical analysis:

- doing some mathematics
- developing an algorithm
- performing error analysis
- implementing the algorithm using a programming language

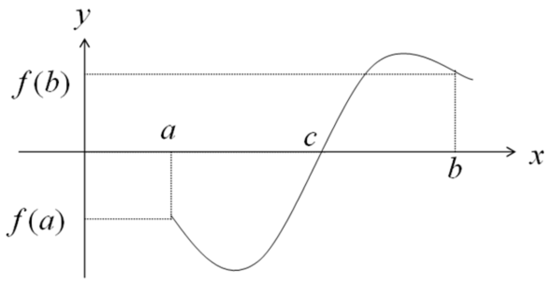
3 Bisection Method

The bisection method is developed based on the **intermediate value theorem (IVT)**. If a function $y = f(x)$ is continuous for all x in the closed interval $[a, b]$, and y_0 is a number between $f(a)$ and $f(b)$, then there is a number $x = c$ in (a, b) that satisfies $f(c) = y_0$.

This definition can be graphically explained in the following figure



As a special case, if we have two distinct values $x = a$ and $x = b$ that satisfy $f(a)f(b) < 0$. Then by the intermediate value theorem, there exists a $c \in [a, b]$ such that $f(c) = 0$. This means, $f(x) = 0$ has root c in $[a, b]$.



3.1 The Logic of Bisection Method

The method calls for a repeated halving of sub-intervals of $[a, b]$ and, at each step, locating the half interval that contains the root r (i.e., $f(r) = 0$).

https://github.com/pengdsci/MAT325/raw/main/w03/img/w03-bisectionMethod.gif

3.2 Bisection Algorithm

Based on the logic of the bisection method, we develop the following pseudo-code to find the root of a given equation $f(x) = 0$ on the interval $[a, b]$ assuming that $f(a)f(b) < 0$.

```
INPUT: ending values: a, b;
           tolerance: TOL
      max. iteration: N
OUTPUT:  approximate root and errors
         error/warning messages
         intermediate outputs

STEP 1.  SET n = 0
         ERR = |b - a|
STEP 2.  WHILE ERR > TOL DO:
         STEP 3  n = n + 1               (updating iteration index)
```

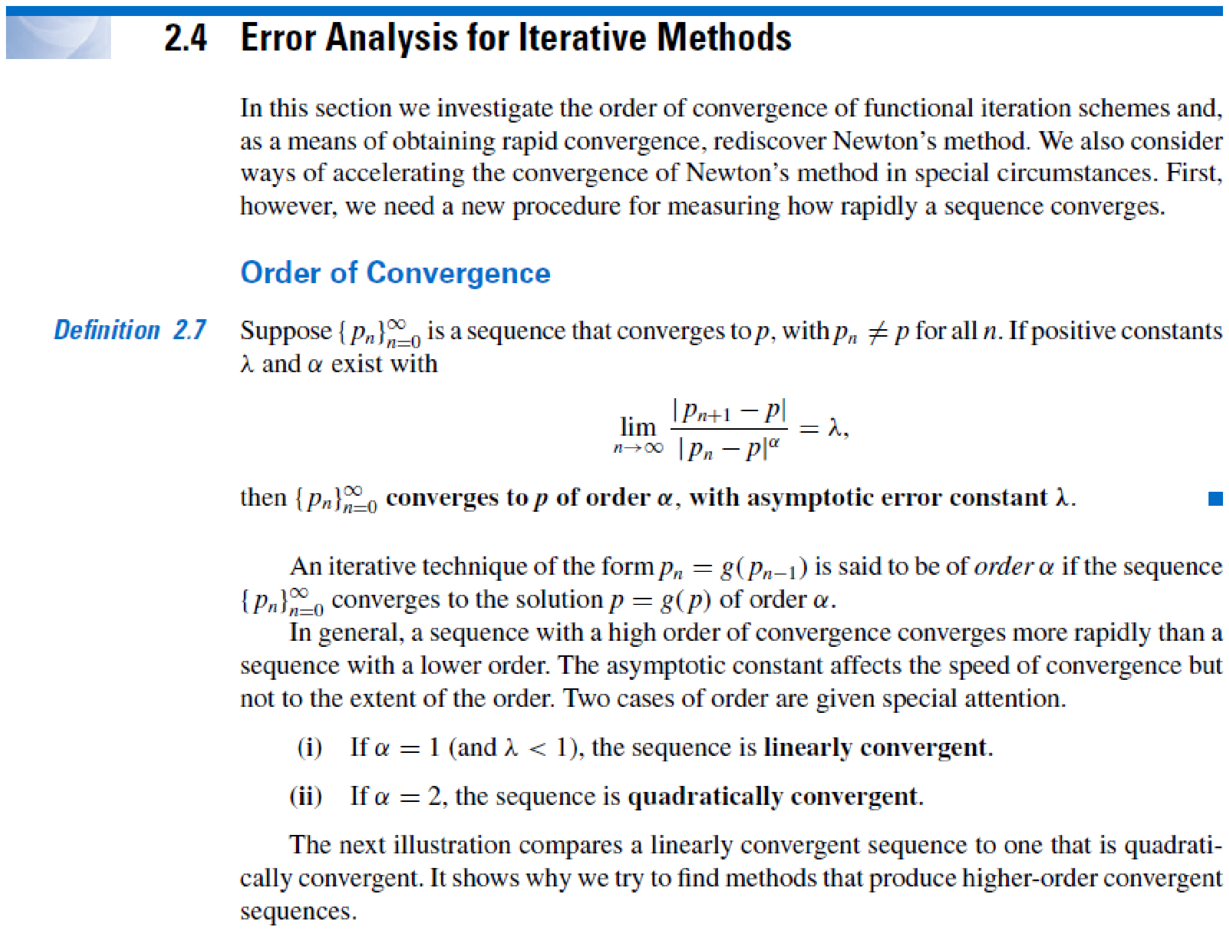
```
        SET m = a + (b - a)/2   (mid-point)
STEP 4. IF f(m)f(b) < 0 DO:
            a = m               (updating ending value)
            b = b               (not necessary...)
        ENDIF
STEP 5. IF f(a)f(b) > 0 DO:
           a = a
           b = m
        ENDIF
STEP 6. ERR = |b - a|
        IF ERR < TOL DO:
            OUTPUT (p = m, other information)
            STOP
        ENDIF
        IF ERR > TOL DO:
           OUTPUT (intermediate info)
        ENDIF
STEP 7. IF n = N DO:
           OUTPUT (message on iteration limit)
           STOP
        ENDIF
ENDWHILE
```

3.3 Error Analysis

Given that we have an initial bound on the problem $[a, b]$, then the maximum error of using either a or b as our approximation is $h = b - a$. Because we halve the width of the interval with each iteration, the error is reduced by a factor of 2, and thus, the error after n iterations will be $h/2^n$. Let r be the true root of $f(x) = 0$ and p_n is the approximate root at n-th iteration. The absolute error of the bisection method is given by

$$|r - p_n| = \frac{b-a}{2^n}.$$

Next, we introduce the order of convergence that is defined in Section 2.4 of the textbook.



2.4 Error Analysis for Iterative Methods

In this section we investigate the order of convergence of functional iteration schemes and, as a means of obtaining rapid convergence, rediscover Newton's method. We also consider ways of accelerating the convergence of Newton's method in special circumstances. First, however, we need a new procedure for measuring how rapidly a sequence converges.

Order of Convergence

Definition 2.7 Suppose $\{p_n\}_{n=0}^{\infty}$ is a sequence that converges to p, with $p_n \neq p$ for all n. If positive constants λ and α exist with

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

then $\{p_n\}_{n=0}^{\infty}$ **converges to p of order α, with asymptotic error constant λ.** ■

An iterative technique of the form $p_n = g(p_{n-1})$ is said to be of *order* α if the sequence $\{p_n\}_{n=0}^{\infty}$ converges to the solution $p = g(p)$ of order α.

In general, a sequence with a high order of convergence converges more rapidly than a sequence with a lower order. The asymptotic constant affects the speed of convergence but not to the extent of the order. Two cases of order are given special attention.

(i) If $\alpha = 1$ (and $\lambda < 1$), the sequence is **linearly convergent**.

(ii) If $\alpha = 2$, the sequence is **quadratically convergent**.

The next illustration compares a linearly convergent sequence to one that is quadratically convergent. It shows why we try to find methods that produce higher-order convergent sequences.

Based on the above definition of the order of convergence, we have

$$\frac{|p_{n+1} - r|}{|p_n - r|^1} = \frac{(b-a)/2^{n+1}}{(b-a)/2^n} = \frac{1}{2}.$$

That is $\alpha = 1$. Therefore, the order of convergence order of the error is **linear**.

3.4 Numerical Examples.

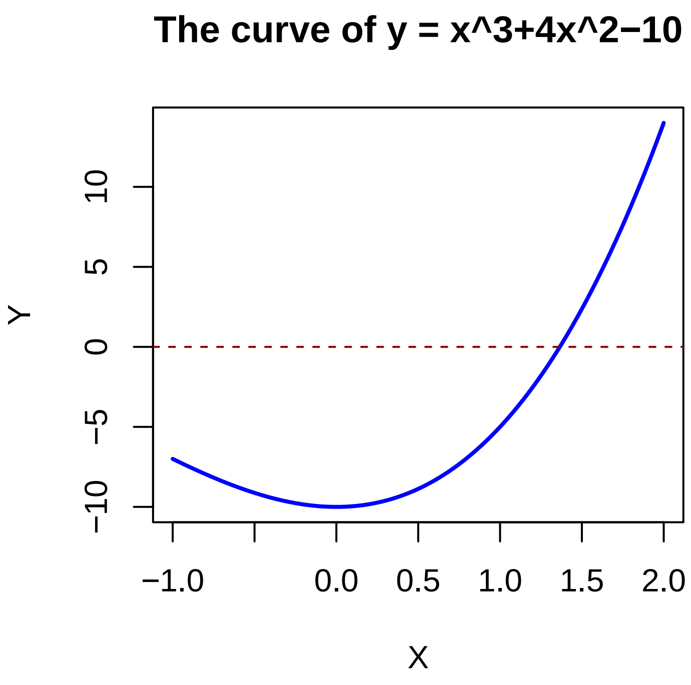
We will present two examples to demonstrate the application of the bisection method.

Example 1: Find a root of equation $x^3 + 4x^2 - 10 = 0$.

Solution: Since we are not given the interval to apply the bisection method. We first sketch $f(x) = x^3 + 4x^2 - 10$ and $y = 0$. The x-coordinate of the intersection of the two curves is the solution to the original equation.

```
x= seq(-1, 2, length = 200)
y = x^3+4*x^2-10
#
```

```r
plot(x, y,
     type = "l",
     lty = 1,
     lwd = 2,
     col = "blue",
     xlab = "X",
     ylab = "Y",
     main = "The curve of y = x^3+4x^2-10")
abline(h = 0, lty = 2, lwd = 1, col = "darkred")
```



Based on the above curve, we choose $[a, b] = [1, 2]$ to implement the bisection algorithm with the following code.

```r
## we first define the R function based on the given equation
fn = function(x){
   f.val =   x^3 + 4*x^2 - 10
   f.val
}

## INPUT INFO
  a = 0
  b = 2
  TOL = 10^(-6)            # tolerance limit
  N = 200                   # max iterations
## Initialization
  n  = 0
  ERR = abs(b - a)
## Loop begins
while(ERR > TOL){
  n = n + 1
  m = a + (b - a) / 2     # midpoint
  if (fn(m)*fn(b) < 0){
      a = m
      b = b
  } else {
      a = a
      b = m
   }
  ERR = abs(b - a)
  ## Output information
  if(ERR < TOL){
    cat("\nThe algorithm converges!")
    cat("\nThe number of iteration n =", n, ".")
    cat("\nThe approximate root r =", (a + b)/2,".")
    cat("\nThe absolute error ERR =", ERR,".")
    break
  } else { ## output of intermediate iteration
    cat("\nIteration:",n,". The absolute error ERR =", ERR, ".")
    }
  ### Test attainment to the max iteration
  if(n == N){
    cat("\n\n Iteration limit reached. The algorithm diverges!")
    break
  }
}
```

```
##
## Iteration: 1 . The absolute error ERR = 1 .
## Iteration: 2 . The absolute error ERR = 0.5 .
## Iteration: 3 . The absolute error ERR = 0.25 .
## Iteration: 4 . The absolute error ERR = 0.125 .
## Iteration: 5 . The absolute error ERR = 0.0625 .
## Iteration: 6 . The absolute error ERR = 0.03125 .
## Iteration: 7 . The absolute error ERR = 0.015625 .
## Iteration: 8 . The absolute error ERR = 0.0078125 .
## Iteration: 9 . The absolute error ERR = 0.00390625 .
## Iteration: 10 . The absolute error ERR = 0.001953125 .
## Iteration: 11 . The absolute error ERR = 0.0009765625 .
## Iteration: 12 . The absolute error ERR = 0.0004882812 .
## Iteration: 13 . The absolute error ERR = 0.0002441406 .
## Iteration: 14 . The absolute error ERR = 0.0001220703 .
## Iteration: 15 . The absolute error ERR = 6.103516e-05 .
## Iteration: 16 . The absolute error ERR = 3.051758e-05 .
## Iteration: 17 . The absolute error ERR = 1.525879e-05 .
## Iteration: 18 . The absolute error ERR = 7.629395e-06 .
## Iteration: 19 . The absolute error ERR = 3.814697e-06 .
## Iteration: 20 . The absolute error ERR = 1.907349e-06 .
## The algorithm converges!
## The number of iteration n = 21 .
## The approximate root r = 1.36523 .
## The absolute error ERR = 9.536743e-07 .
```

4 Chapter 2 Homework - Part I.

Section 2.1: 5(a), 7, 13 (see the hint in problem 12 to set up the equation)

Requirements:

1. Use the RMarkdown document to complete these problems.
2. For each problem, sketch the equation to find an appropriate interval to search the root.
3. Convert the RMarkdown to a PDF file and submit it to the D2L dropbox