Numerical Analysis

(ENME 602)

Spring 2021

Lecture 2

Dr. Hesham H. Ibrahim
Associate Professor, Mechatronics Department.
hesham.hamed-ahmed@guc.edu.eg
Office C7.04

Lecture 2

Roots of Non-linear Equations

2.1 Algorithms Stability and Convergence

2.2 Root Finding Problem Statement

2.3 Bisection Method

2.1 Algorithms Stability and Convergence

2.1 Algorithms Stability and Convergence

Algorithm: is an approximation procedure that describes a finite sequence of steps to be performed in a specified order.

The object of the algorithm is to implement a procedure to solve a problem or approximate a solution to the problem.

Pseudocode: used to describe an algorithm. It follows the rules of structured program construction.

It specifies the input, output and a stopping technique independent of the numerical technique. It can be easily translated in any programming language: e.g Matlab, Python or Java.

2.1 Algorithms Stability and Convergence

Example: Write an algorithm to compute:

$$\sum_{i=1}^{N} x_i = x_1 + x_2 + \ldots + x_N,$$

where N and the numbers $x_1, x_2, \ldots, x_N$ are given.

Pseudocode:

INPUT $N, x_1, x_2, \ldots, x_N$.

OUTPUT $SUM = \sum_{i=1}^{N} x_i$.

Step 1 Set $SUM = 0$. (*Initialize accumulator.*)

Step 2 For $i = 1, 2, \ldots, N$ do
set $SUM = SUM + x_i$. (*Add the next term.*)

Step 3 OUTPUT (SUM);
STOP.

2.1 Algorithms Stability and Convergence

Algorithm stability if small changes in the initial data (input) produce correspondingly small changes in the final results (output), the algorithm is called **stable**; otherwise it is **unstable.**

Round-off errors growth & algorithm stability

Let $E_0 > 0$ be an initial error, E_n error after n operations. Let C a constant independent of n:

- The **growth of error** is said to be **linear** and **the algorithm is stable** if

$$E_n \approx C \times n \times E_0$$

- The **growth of error** is said to be **exponential** and **the algorithm is unstable** if

$$E_n \approx C^n \times E_0$$

2.1 Algorithms Stability and Convergence

Algorithm stability-Example: The recursive equation

$$p_n = \frac{10}{3} p_{n-1} - p_{n-2}, \quad n = 2, 3, \ldots$$

has exact solution

$$p_n = c_1 \left(\frac{1}{3}\right)^n + c_2 \cdot 3^n \tag{1}$$

for any constants c_1, c_2 (determined by starting values p_0 & p_1).
In particular, if $p_0 = 1$ and $p_1 = \frac{1}{3}$, substitute into (1) to get

$$\begin{cases} \quad c_1 + \quad c_2 = p_0 = 1 \\ \left(\frac{1}{3}\right) c_1 + 3c_2 = p_1 = \frac{1}{3} \end{cases}$$

we get $c_1 = 1$, $\quad c_2 = 0$ and $p_n = \left(\frac{1}{3}\right)^n$ for all n.

2.1 Algorithms Stability and Convergence

Algorithm stability-Example: Now, consider what happens when p_n is estimated in **5-digit rounding arithmetic**, where $p_0 = 1$, $p_1 = \frac{1}{3}$ are rounded:

$$p_0^* = 1.0000 \quad p_1^* = 0.33333$$

Substitute p_0^* and p_1^* into the general solution

$$p_n = c_1 \left(\frac{1}{3}\right)^n + c_2 \cdot 3^n$$

and solve for c_1 and c_2

$$\begin{cases} c_1 + \ \ c_2 = p_0^* = 1.0000 \\ \left(\frac{1}{3}\right) c_1 + 3c_2 = p_1^* = 0.33333 \end{cases}$$

we get modified constants $c_1^* = 1.0000$, $\quad c_2^* = -0.12500 \times 10^{-5}$ and the approximation of the general solution is

$$p_n^* = c_1^* \left(\frac{1}{3}\right)^n + c_2^* \, 3^n$$

2.1 Algorithms Stability and Convergence

Algorithm stability-Example: To recapitulate:

For $p_0 = 1$, $p_1 = \frac{1}{3}$, we get $c_1 = 1$, $c_2 = 0$ and the exact solution is

$$p_n = \left(\frac{1}{3}\right)^n$$

With 5-digit rounding, $p_0^* = 1.0000$, $p_1^* = 0.33333$, we get $c_1^* = 1.0000$, $c_2^* = -0.12500 \times 10^{-5}$ and the approximate solution is

$$p_n^* = 1.0000\left(\frac{1}{3}\right)^n - 0.12500 \times 10^{-5}(3)^n$$

The round-off error clearly grows exponentially with n:

$$E_n = |p_n - p_n^*| = \underbrace{0.12500 \times 10^{-5}(3)^n}_{\text{exponential growth}}$$

2.1 Algorithms Stability and Convergence

Algorithm stability-Example: Exponential growth gives very large relative errors for the first few iterations

n	Computed $\hat{p}_n$	Correct p_n	Relative error
0	0.10000×10^{1}	0.10000×10^{1}	
1	0.33333×10^{0}	0.33333×10^{0}	
2	0.11110×10^{0}	0.11111×10^{0}	9×10^{-5}
3	0.37000×10^{-1}	0.37037×10^{-1}	1×10^{-3}
4	0.12230×10^{-1}	0.12346×10^{-1}	9×10^{-3}
5	0.37660×10^{-2}	0.41152×10^{-2}	8×10^{-2}
6	0.32300×10^{-3}	0.13717×10^{-2}	8×10^{-1}
7	-0.26893×10^{-2}	0.45725×10^{-3}	7×10^{0}
8	-0.92872×10^{-2}	0.15242×10^{-3}	6×10^{1}

2.1 Algorithms Stability and Convergence

Algorithms Convergence

- The concepts of limits and convergence of sequences you have covered in early calculus courses still apply to the sequence of approximations generated by the numerical iterative techniques in this course.
- A numerical iterative technique will start with an initial value (let us say α_o) and generates a sequence of approximations (α_n) to the solution (α) we are trying to compute.
- Hence in numerical analysis, the list of approximations generated by the given numerical method (α_n) **is the sequence** and the solution (α) we are trying to approximate i**s the limit.**

2.1 Algorithms Stability and Convergence

Algorithms Convergence

- First let us recall What does it mean when we say a sequence "α_n" converges to its limit "α"?
  - ✓ It means when "n" starts getting enough large, the value of "α_n" gets closer to the value of the limit "α".
  - ✓ Remember n here represents the iteration index, which means when you generate enough iterations, you are likely to get a better approximation for the limit value "α".
  - ✓ IF this doesn't happen, the sequence is said to be *divergent*.
- In numerical analysis, it is not enough to *converge* to a limit, the speed of convergence also matters, in order to say that a method is efficient.
- For example, The Bisection Method always converges but because it converges with a very slow rate, it is not used in practice.
- In general, we would like the numerical technique to converge as rapidly as possible.

2.1 Algorithms Stability and Convergence

Rate of convergence: To describe the rate at which convergence occurs, when a numerical technique produces a sequence of approximations α_n, we compare to the convergence rate of a known sequence β_n:

Definition: Suppose $(\beta_n)_{n\geq 1}$ is a sequence known to converge to zero, and $(\alpha_n)_{n\geq 1}$ converges to α. If there exists $K > 0$ with

$$|\alpha_n - \alpha| \leq K \times |\beta_n| \quad \text{for large } n,$$

then we say that $(\alpha_n)_{n\geq 1}$ converges to α with rate of convergence $O(\beta_n)$. (Read as "big oh of β_n") and written as

$$\alpha_n = \alpha + O(\beta_n).$$

Often β_n is taken as $\beta_n = \dfrac{1}{n^p} \longrightarrow 0$, the larger is p the faster is the rate of convergence.

2.1 Algorithms Stability and Convergence

Example:(Ex.6(b), Sec.1.3) Find the rate of convergence of the sequence

$$\lim_{n\to\infty} \sin\left(\frac{1}{n^2}\right) = 0$$

- Let $\alpha_n = \sin\left(\frac{1}{n^2}\right)$ and $\alpha = 0$, we show that $|\alpha_n - \alpha|$ is bounded above by a sequence β_n converging to zero.

  $$|\alpha_n - \alpha| = \left|\sin\left(\frac{1}{n^2}\right) - 0\right| = \left|\sin\left(\frac{1}{n^2}\right)\right| \le \left|\frac{1}{n^2}\right| \quad \text{for all } n \ge 1$$

  (using the inequality $|\sin(x)| \le |x|, \quad$ for all $x \in \mathbb{R}$).
- Hence $\beta_n = \dfrac{1}{n^2} \longrightarrow 0 \quad \text{as} \quad n \to \infty$.

The rate of convergence of $\sin(\frac{1}{n^2})$ is $O(\frac{1}{n^2})$. We write

$$\sin\left(\frac{1}{n^2}\right) = 0 + O\left(\frac{1}{n^2}\right)$$

That is, $\sin(\frac{1}{n^2}) \longrightarrow 0$ about as fast as $\frac{1}{n^2}$ converges to zero.

Solutions of Non-linear Equations in One Variable

- Bisection Method
- Newton's Method and its extensions
  - ✓ Newton's Method
  - ✓ Secant Method
  - ✓ Method of False Position
- Fixed-Point Method

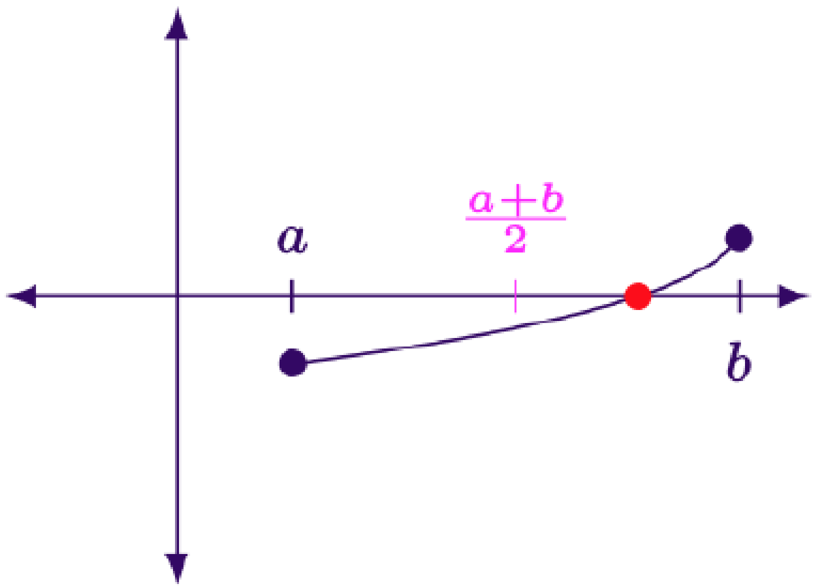
2.2 Root Finding Problem Statement

2.2 Root Finding Problem Statement

The root of an equation

The value of x that satisfies $f(x) = 0$ is called the root (solution) of the equation $f(x) = 0$ or the zero of the function f.

Graphically, it is the x-intercept when the graph of f crosses the x-axis.



The root-finding problem

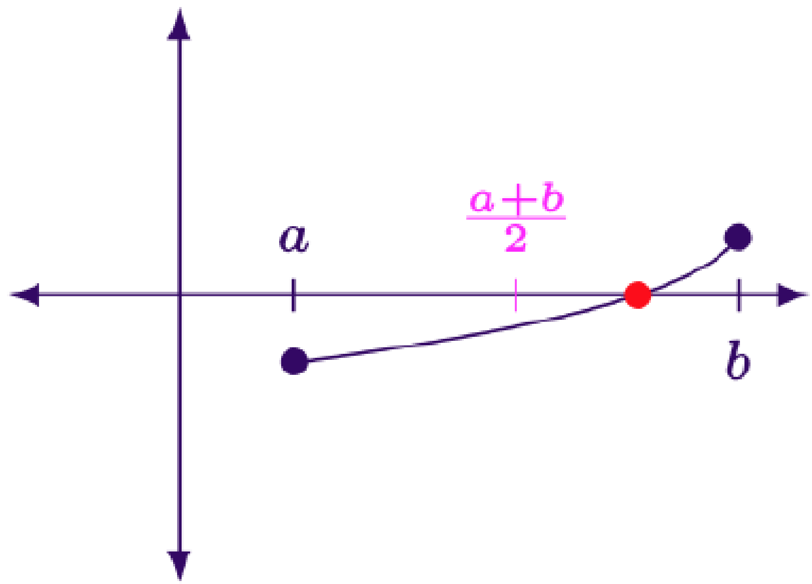
is the process of approximating the root (solution) of the equation $f(x) = 0$ (to within a given tolerance).

2.3 Bisection Method

2.3 Bisection Method

Idea of bisection method

When there is a root (red dot here), it occurs either in the right half of the interval [a,b] (like in the graph) or in the left half or at the midpoint. Take the half that contains the root!

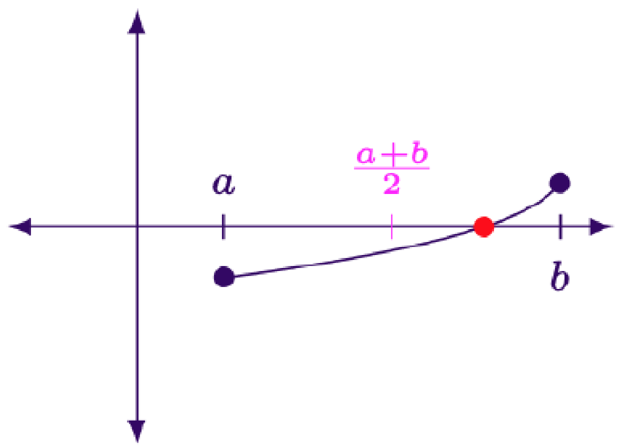


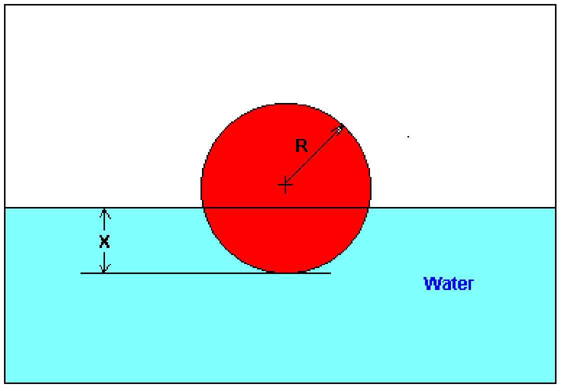
We obtain an approximation of the root by repeatedly halving the subintervals of [a,b], and "trapping" the root in smaller and smaller interval at each step.

2.3 Bisection Method

Idea of bisection method

When there is a root (red dot here), it occurs either in the right half of the interval [a,b] (like in the graph) or in the left half or at the midpoint. Take the half that contains the root!





Example: The equation that gives the depth to which the ball (given its specific gravity and radius) is submerged under water is given by *x:*

$$x^3 - 0.165x^2 + 3.993 \times 10^{-4} = 0$$

$$0 \le x \le 2R$$
$$0 \le x \le 2(0.055)$$
$$0 \le x \le 0.11$$

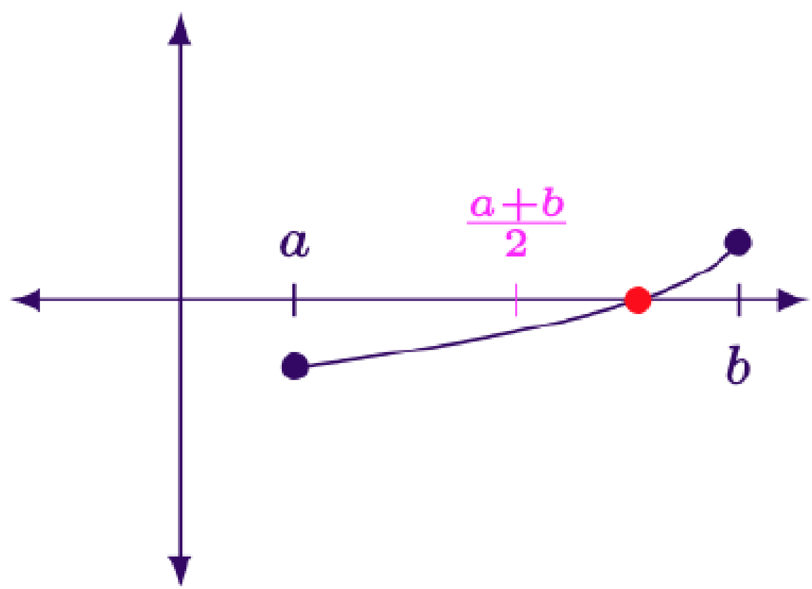
2.3 Bisection Method

Idea of bisection method

Q: How to know that a root exists in $[a, b]$?

Answer: apply the Intermediate Value Theorem:

If f is continuous on $[a, b]$ and $f(a)$ and $f(b)$ are of opposite signs (which is usually expressed as their product $f(a) \cdot f(b) < 0$) then there exist p such that $f(p) = 0$.



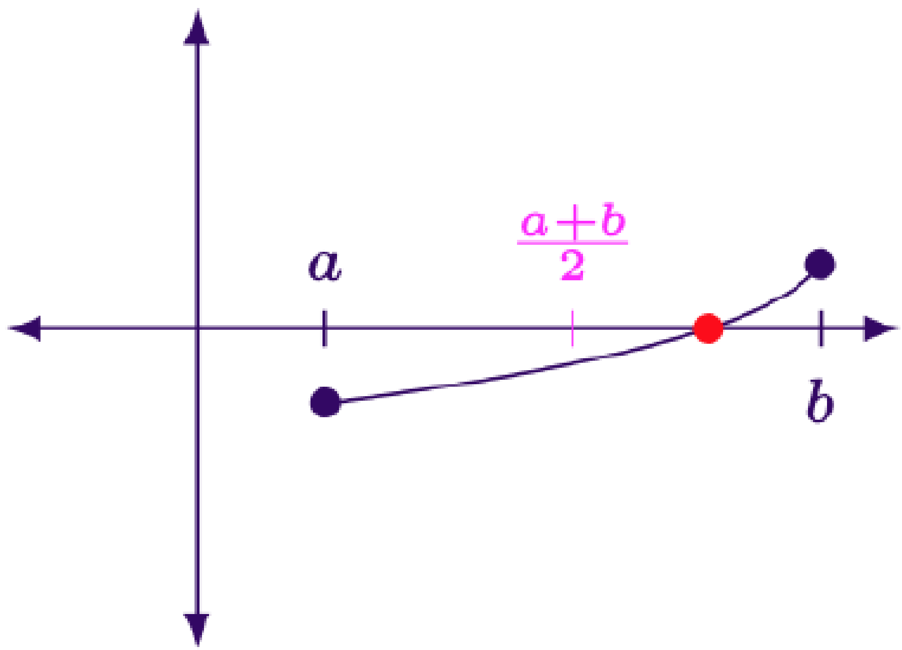
Note: We assume there is a unique solution inside the interval.

2.3 Bisection Method

Idea of bisection method

Halving the interval $[a, b]$

The method calls for a repeated halving of subintervals of [a, b] and, at each step, locating the half containing p.



2.3 Bisection Method

Idea of bisection method

Q: How to select the half that contains the root?

Construct the sequence of subintervals $[a_k, b_k]$ iteratively:

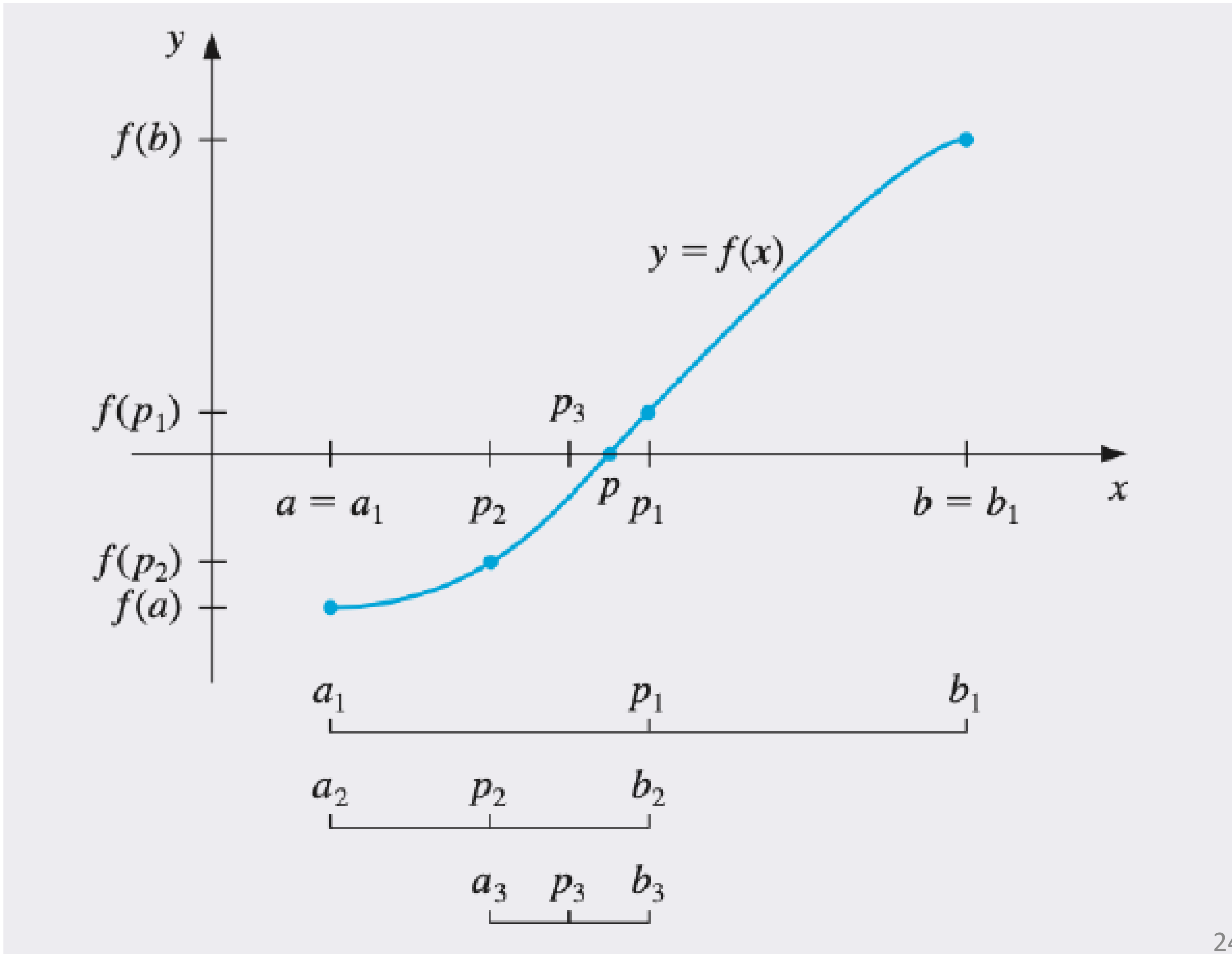
- Begin with first interval $[a_1, b_1] = [a, b]$ such that $f(a)f(b) < 0$ (so f has a root in $[a, b]$).
- Compute the midpoint $p = \dfrac{a+b}{2}$.
- To choose left or right subinterval: check the sign of $f(p)$ and compare it to the signs of $f(a)$ and $f(b)$.

  If $f(a)f(p) < 0$ then f change signs in $[a, p]$, the new interval $[a_2, b_2] = [a, p]$, otherwise, take $[p, b]$.

- Set either $[a, p]$ or $[p, b]$ as new interval. At each iteration use p as new approximation of the root.

2.3 Bisection Method

Bisection illustration



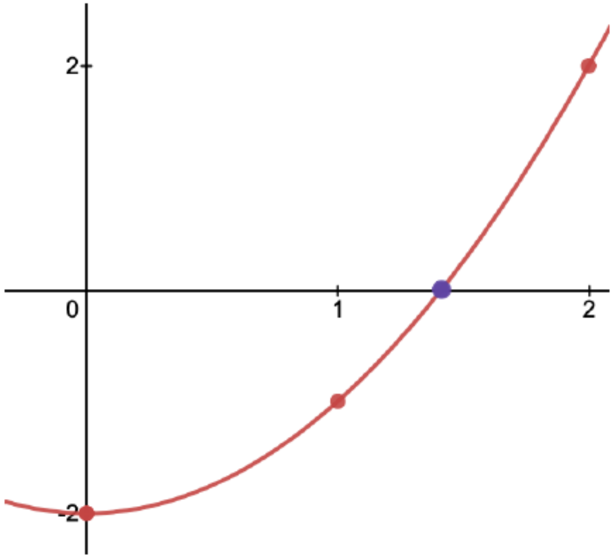
2.3 Bisection Method

Example 1

Approximate $\sqrt{2}$, using bisection method

Let $x = \sqrt{2}$, to approximate x, convert it to a root-finding problem by squaring both sides $x^2 = 2$, then let $f(x) = x^2 - 2$, and seek a positive value for x that is a root of $f(x) = 0$. Select a starting interval $[0, 2]$ that contains the root.

Note that $f(0) \cdot f(2) = (-2) \cdot (2) = -4 < 0$, so there exists $p \in [0, 2]$ such that $f(p) = p^2 - 2 = 0$.

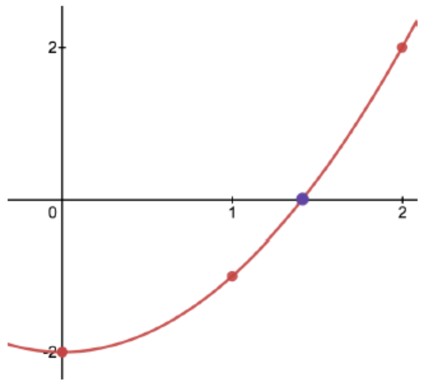


2.3 Bisection Method

Example 1

Approximate $\sqrt{2}$, using bisection method (compute p_3)

Given $[a, b] = [0, 2]$ with $f(0) < 0$ and $f(2) > 0$:



- Iteration 1: Compute the midpoint & its image:

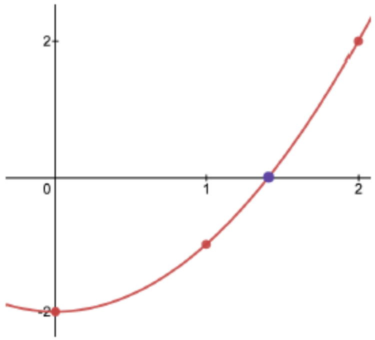
$$p_1 = \frac{a+b}{2} = 1, \quad f(p_1) = -1 < 0$$

- Clearly $f(p_1) \cdot f(b) = f(1) \cdot f(2) = (-1)(2) = -2 < 0$, hence the root lies in $[p_1, b] = [1, 2]$, the new interval.

2.3 Bisection Method

Example 1

Approximate $\sqrt{2}$, using bisection method (compute p_3)



- Iteration 2: The new interval is $[a_2, b_2] = [p_1, b] = [1, 2]$, compute the new midpoint p_2 & its image:

$$p_2 = \frac{p_1 + b}{2} = \frac{3}{2} \approx 1.500, \quad f(p_2) = \frac{1}{4} > 0$$

- This time

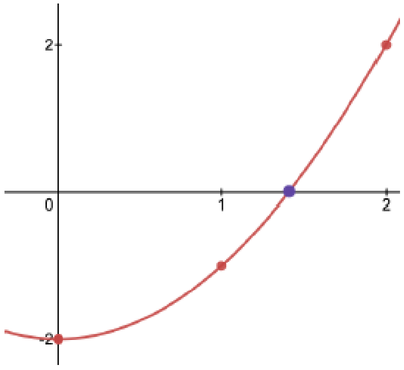
$$f(a_2) \cdot f(p_2) = f(p_1) \cdot f(p_2) = f(1) \cdot f(1.5) = (-1)(2) = -2 < 0$$

hence the root lies in $[p_1, p_2] = [1, 1.5]$.

2.3 Bisection Method

Example 1

Approximate $\sqrt{2}$, using bisection method (compute p_3)



- Iteration 3: The new interval is $[a_3, b_3] = [p_1, p_2] = [1, 1.500]$, compute p_3:

$$p_3 = \frac{p_1 + p_2}{2} = \frac{5}{4} = 1.2500 \approx \sqrt{2} = 1.4142\ldots$$

Example 1

Approximate $\sqrt{2}$, using bisection method (compute p_3)

Get used to organize your iterations in a table:

n	a_n	b_n	p_n	$f(p_n)$	$E_{relative} = \frac{\|p_n - p_{n-1}\|}{\|p_n\|}$
1	0	2	$p_1 = 1.0000$	-1	
2	1	2	$p_2 = 1.5000$	0.2500	
3	1	1.5000	$p_3 = 1.2500$		

Note that $p_3 = 1.2500$ is not yet the best approximation to $\sqrt{2} \approx 1.4142$ since it approximates it to only one decimal precision.

2.3 Bisection Method

Stopping Criteria: Error bound for bisection method

Q1: How do we know when to stop iterations?
Q2: Is convergence guaranteed?

Theorem (2.1)

Suppose that $f \in C[a,b]$ and $f(a) \cdot f(b) < 0$. The Bisection method generates a sequence $\{p_n\}_{n=1}^{\infty}$ approximating a zero p of f with

$$|p_n - p| \le \frac{b-a}{2^n}, \quad \text{when} \quad n \ge 1.$$

This Theorem is a powerful tool, because:

- It guarantees convergence (under assumptions f continuous and $f(a)f(b) < 0$)
- it gives an upper bound for the absolute error $|p - p_n| \le \frac{b-a}{2^n}$.
- it gives the exact number of steps needed to achieve a desired accuracy, and therefore to stop iterating.

2.3 Bisection Method

Stopping Criteria: Error bound for bisection method

Q1: How do we know when to stop iterations?

If we iterate until the n^{th} sub-interval length is at most ϵ:

$$\frac{b-a}{2^n} \leq \epsilon \tag{1}$$

The Theorem guarantees that the absolute error is at most ϵ:

$$|p - p_n| \leq \frac{b-a}{2^n} \leq \epsilon$$

To determine the number of iteration **n** necessary to reach a tolerance ϵ in $[a, b]$, solve (1) for **n**: taking ln of both sides

$$\ln\left(\frac{b-a}{2^{\mathbf{n}}}\right) < \ln(\epsilon)$$

$$\ln(b-a) - \mathbf{n}\ln 2 < \ln(\epsilon) \Longrightarrow \mathbf{n} > \frac{\ln(b-a) - \ln(\epsilon)}{\ln 2} = \frac{\ln\left(\frac{b-a}{\epsilon}\right)}{\ln 2}$$

2.3 Bisection Method

Example on error bound

How many iterations needed to approximate $\sqrt{2}$ to within a tolerance 10^{-3}, using 5-digits rounding arithmetic?

- To determine the number of iteration **n** necessary to reach a tolerance ϵ, in $[0, 2]$, requires finding **n** that satisfies:

$$|p - p_{\mathbf{n}}| \leq \frac{b-a}{2^{\mathbf{n}}} < 10^{-3}$$

- For $\epsilon = 10^{3}$, $a = 0$, and $b = 2$, the minimum number of iteration to reach the precision ϵ satisfies

$$\mathbf{n} > \frac{\ln\left(\dfrac{b-a}{\epsilon}\right)}{\ln 2} = \frac{\ln\left(\dfrac{2-0}{10^{-3}}\right)}{\ln 2} \approx 10.9658$$

Hence we need at least $\mathbf{n} = 11$ iterations to reach the precision $\epsilon = 10^{-3}$.

2.3 Bisection Method

Bisection Algorithm

To find a solution to $f(x) = 0$ given the continuous function f on the interval $[a, b]$, where $f(a)$ and $f(b)$ have opposite signs:

INPUT endpoints a, b; tolerance TOL; maximum number of iterations N_0.

OUTPUT approximate solution p or message of failure.

Step 1 Set $i = 1$;
$FA = f(a)$.

Step 2 While $i \leq N_0$ do Steps 3–6.

Step 3 Set $p = a + (b - a)/2$; *(Compute p_i.)*
$FP = f(p)$.

Step 4 If $FP = 0$ or $(b - a)/2 < TOL$ then
OUTPUT (p); *(Procedure completed successfully.)*
STOP.

Step 5 Set $i = i + 1$.

Step 6 If $FA \cdot FP > 0$ then set $a = p$; *(Compute a_i, b_i.)*
$FA = FP$
else set $b = p$. *(FA is unchanged.)*

Step 7 OUTPUT ('Method failed after N_0 iterations, $N_0 =$', N_0);
(The procedure was unsuccessful.)
STOP.

2.3 Bisection Method

- The Bisection method, though conceptually clear, has significant drawbacks. It is relatively slow to converge (that is, N may become quite large before $|p - p_n|$ is sufficiently small), and a good intermediate approximation might be unintentionally discarded.

- However, the method has the important property that it always converges to a solution, and for that reason it is often used as a starter for the more efficient methods we will see later in False Position Method.

Thank You

- ***Extension of Newton's Method***
  - ✓ ***Secant Method***
  - ✓ ***Method of False Position***
- ***Fixed-Point Method***