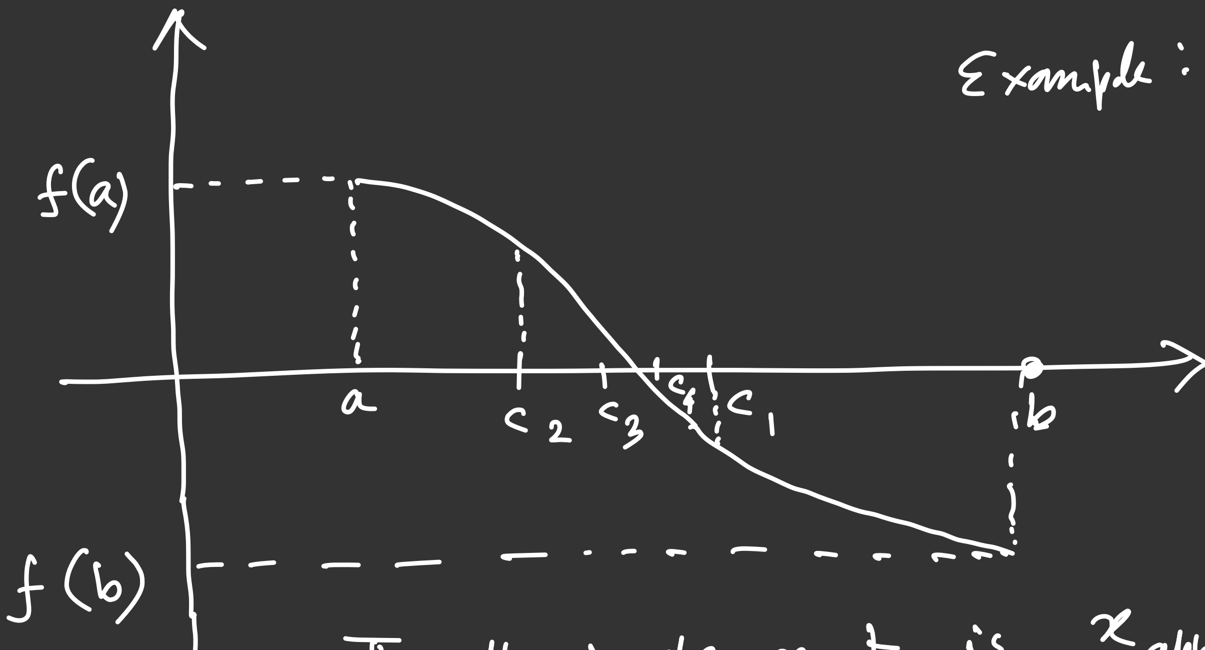
Lecture # 6

01/23/2023

$[a, b] \longrightarrow [a, c_1] \rightarrow [c_2, c_1] \rightarrow [c_2, c_3] \rightarrow \cdots$

Example: $\varepsilon = 10^{-3}, 10^{-4}$ etc.



The approximate root is $x_{approx} = \left(\dfrac{\text{current } a + \text{current } b}{2} \right)$

Analysis of the bisection algorithm:

$$\underbrace{\begin{matrix}[a,b] \\ [a_1,b_1]\end{matrix}} \xrightarrow{2^{nd}} [a_2, b_2] \xrightarrow{3^{rd}} [a_3, b_3] \longrightarrow$$

$[a_1, b_1]$ has length $= b - a$

$[a_2, b_2]$ has length

$$b_2 - a_2 = \frac{b_1 - a_1}{2} = \frac{b-a}{2}$$

$[a_3, b_3]$ has length

$$b_3 - a_3 = \frac{b_2 - a_2}{2} = \frac{b_1 - a_1}{2 . 2} = \frac{b-a}{2^2}$$

$$\ldots \xrightarrow{n^{th} \text{ pass}} [a_n, b_n]$$

has length $= b_n - a_n = \dfrac{b-a}{2^{n-1}}$

$\therefore$ The absolute error

$$= \left| x_{approx} - x_{true} \right|$$

$$\leq \frac{1}{2} \cdot \frac{b-a}{2^{n-1}} = \frac{b-a}{2^{n}}$$

because $x_{approx} = \dfrac{a_n + b_n}{2}$

$$x_{approx} = \frac{a_n + b_n}{2}$$

$$b_n - a_n = \frac{b-a}{2^{n-1}}$$

Given the desired numerical tolerance ε supplied by the user, what should be the # of iterations n_{min} should the bisection algorithm execute?

User wants: $\underbrace{\dfrac{b-a}{2^n}}_{\text{absolute error}} \leq \varepsilon$

$$\Rightarrow \quad 2^n \geq \frac{b-a}{\varepsilon}$$

$$\Rightarrow \quad n \ln(2) \geq \ln\left(\frac{b-a}{\varepsilon}\right)$$

$\uparrow$ natural logarithm

$$\Rightarrow \quad n \geq \frac{\ln\left(\frac{b-a}{\varepsilon}\right)}{\ln(2)}$$

$$\Rightarrow \quad n_{\text{minimum}} = \left\lceil \frac{\ln\left(\frac{b-a}{\varepsilon}\right)}{\ln(2)} \right\rceil$$

Ceiling function $\lceil \cdot \rceil$

MATLAB command ceil(·)

Example: $\lceil 27.2 \rceil = 28$

Example: Suppose we want to solve $\underbrace{f(x)}_{\text{continuous}} = 0$

and $x \in [0, 1]$.

So, $a = 0$, $b = 1$

desired tolerance $\varepsilon = 0.001 = 10^{-3}$

Then: $n_{\text{minimum}} = \left\lceil \dfrac{\ln\left(\frac{1-0}{\varepsilon}\right)}{\ln 2} \right\rceil$

$= \left\lceil \dfrac{\ln(1000)}{\ln(2)} \right\rceil = \lceil 9.9658 \rceil$

$= 10$

Advantage / Pro	Disadvantage / Con
It always <u>converges</u> (No divergence, no osillation)	It is "slow" (only linear convergence)

Suppose e_n denote the "absolute error" after the n^{th} iteration $\iff$ $e_n := |x_n - x_{true}|$

We say: an algorithm has "<u>order of convergence</u>" <u>α</u> and "<u>asymptotic error constant</u>" λ

(next pg.)

$$\iff \lim_{n\to\infty} \frac{e_{n+1}}{e_n^{\alpha}} = \lambda$$

Special cases:

- $\alpha = 1$, we say algorithm has linear convergence
- $\alpha = 2$, " " " " quadratic convergence
- $\alpha = 3$, " " " " cubic convergence

For large n, we get:

$$\alpha \approx \underbrace{\frac{\log(e_{n+1}) - \log(e_n)}{\log(e_n) - \log(e_{n-1})}}$$

we can estimate this quantity from data

<u>Claim:</u> Bisection algorithm has linear convergence.

<u>Why?</u> $$\frac{e_{n+1}}{e_n} = \frac{(b-a)/2^{n+1}}{(b-a)/2^n} = \frac{2^n}{2^{n+1}} = \frac{1}{2}$$

$$\Rightarrow \lim_{n\to\infty} \frac{e_{n+1}}{e_n} = 1/2 \rightarrow \lambda.$$

$\therefore$ $\alpha = 1$, $\lambda = 1/2$