NUMERICAL METHODS
Week-4
04.03.2014

Nonlinear Systems & Equations

Asst. Prof. Dr. Berk Canberk

Nonlinear System& Equation

- **What** is a nonlinear System/equation?
- **Why** do we use nonlinear systems?
- **How** do we use/represent a nonlinear system?

What is a Nonlinear System?

- It is a mathematical model of a system which can not be represented by a scalar multiplication and addition!
- It is a system in which the effect of external factors is not purely additive, and may be disruptive.
- It is a system which cannot be decomposed into several parts and reassembled into the same thing
- It is a system of which the output do not change in proportion to a change in the input.

Why do we use Nonlinear systems?

- Indeterminism
  - The behavior of a system that cannot be predicted (stochastic behaviors).
- Aperiodic events
  - behaviors that do not repeat values after some period (also known as chaotic events or chaos).
- Multistability
  - Alternating between two or more states.
- Other examples:
  - AC power flow model
  - Optics &water waves (Nonlinear Schrödinger Eqs)
  - Diffusion
  - Atmospheric models
  - ---

How do we represent?

- By nonlinear algebraic equations
  - quadratic equations
  - Cubic equations
- By nonlinear recurrence relations
- By nonlinear differential equations

How do we represent?

$$\lambda \cosh\left(\frac{50}{\lambda}\right) = \lambda + 10$$

$$f(x) = 6x^2 - 7x + 2$$

$$\begin{cases} I = a(e^{bV} - 1) \\ c = dI + V \end{cases}$$

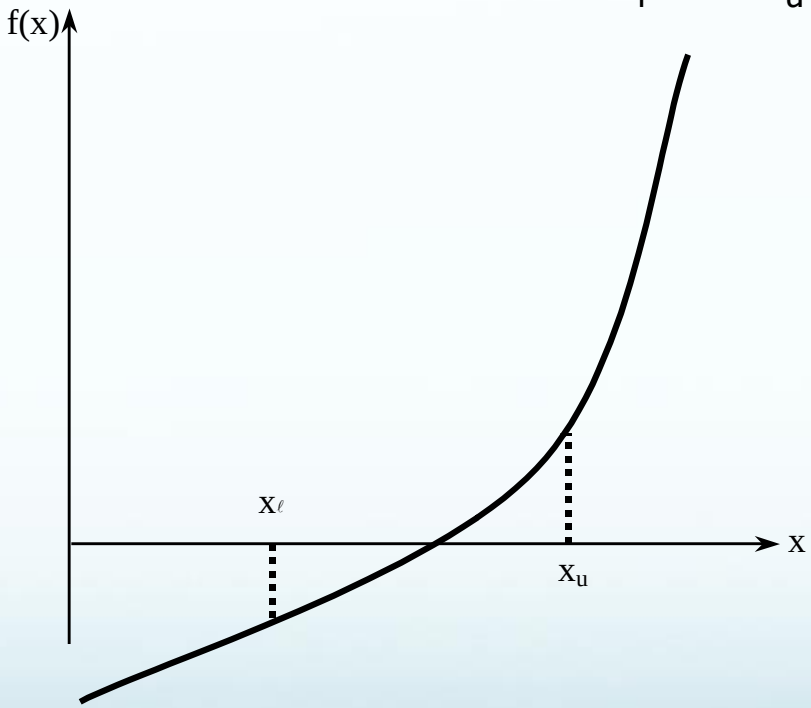
$$f(x) = 3.24x^8 - 2.42x^7 + 10.34x^6 + 11.01x^2 + 47.98$$
$$g(x) = 2^{x^2} - 10x + 1$$
$$h(x) = \cosh\left(\sqrt{x^2+1} - e^x\right) + \log|\sin x|$$

Basic NLS solving methods

- Bisection Method
- Newton-Rapson Method
- Secant Method

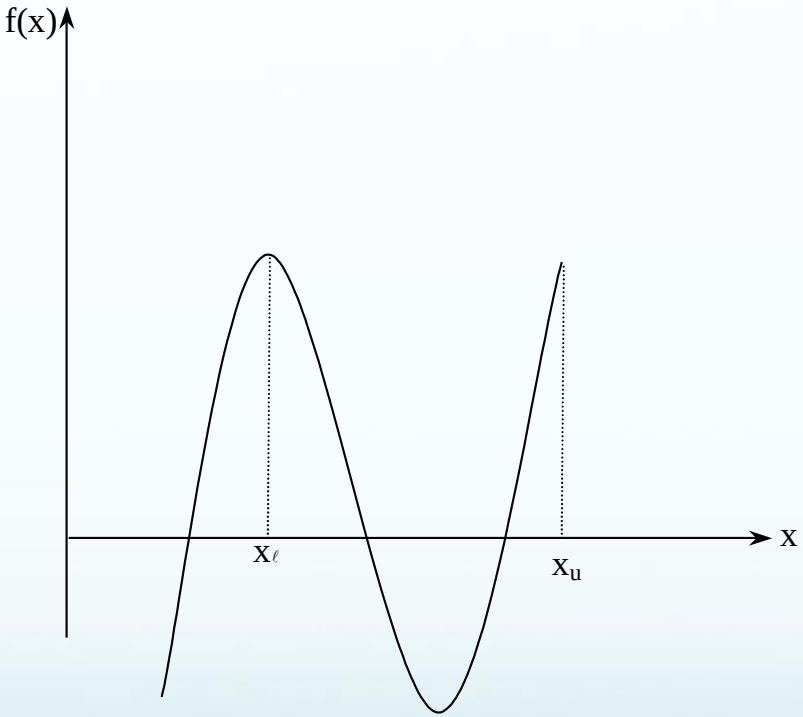
Basis of Bisection Method

Theorem An equation f(x)=0, where f(x) is a real continuous function, has at least one root between x_l and x_u **if $f(x_l)\, f(x_u) < 0$.**



At least one root exists between the two points if the function is real, continuous, and changes sign.

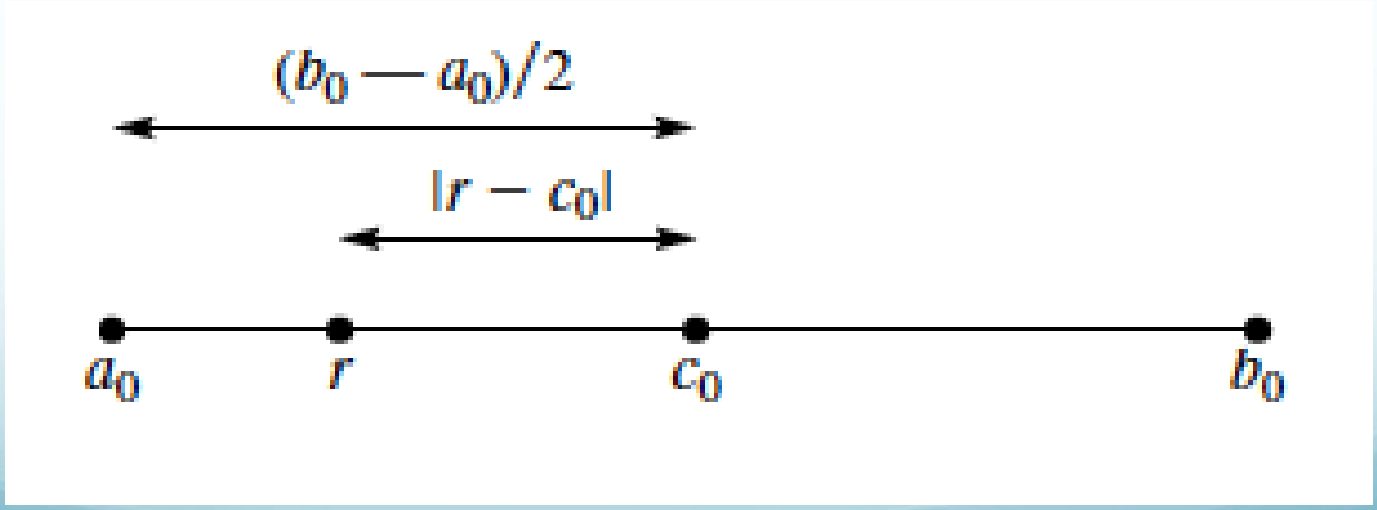
Basis of Bisection Method



If function $f(x)$ does not change sign between two points, roots of the equation $f(x)=0$ may still exist between the two points.

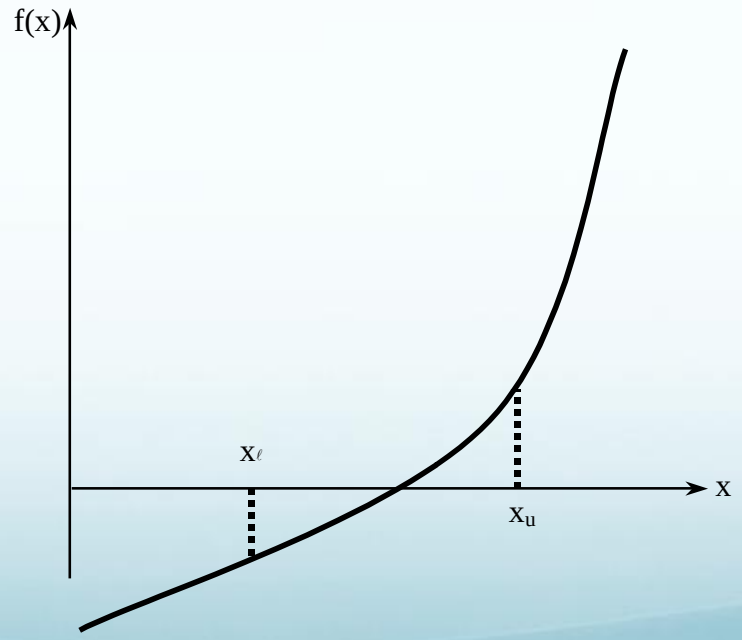
Algorithm for Bisection Method

Basic Idea:



Step 1

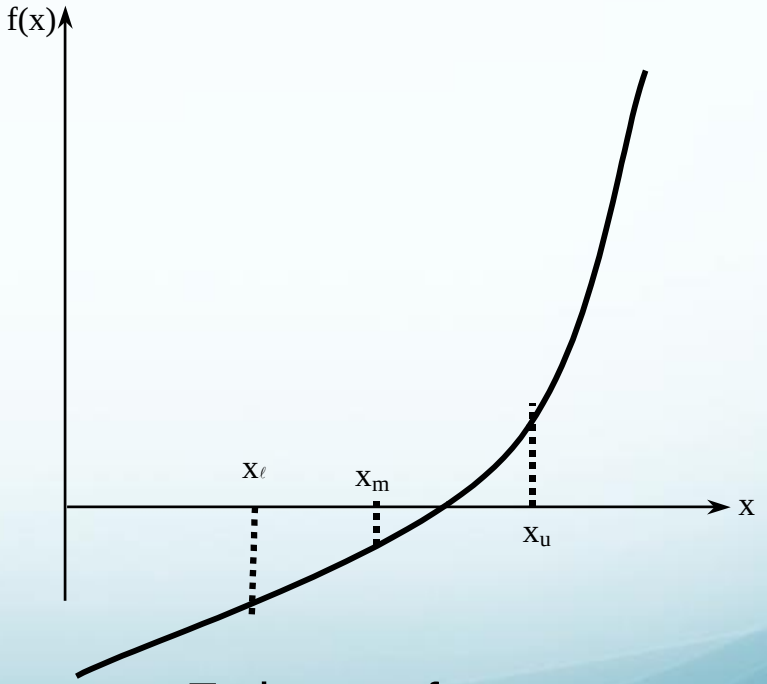
Choose x_ℓ and x_u as two guesses for the root such that $f(x_\ell)\, f(x_u) < 0$, or in other words, f(x) changes sign between x_ℓ and x_u.



Step 2

Estimate the root, x_m of the equation f (x) = 0 as the mid point between x_ℓ and x_u as

$$x_m = \frac{x_\ell + x_u}{2}$$



Estimate of x_m

Step 3

Now check the following

a) If $f(x_l)f(x_m)<0$, then the root lies between x_ℓ and x_m; then $x_\ell = x_\ell$; $x_u = x_m$.

b) If $f(x_l)f(x_m)>0$, then the root lies between x_m and x_u; then $x_\ell = x_m$; $x_u = x_u$.

c) If $f(x_l)f(x_m)=0$; then the root is x_m. Stop the algorithm if this is true.

Step 4

Find the new estimate of the root

$$x_m = \frac{x_\ell + x_u}{2}$$

Find the absolute relative approximate error

$$\left|\epsilon_a\right| = \left|\frac{x_m^{new} - x_m^{old}}{x_m^{new}}\right| \times 100$$

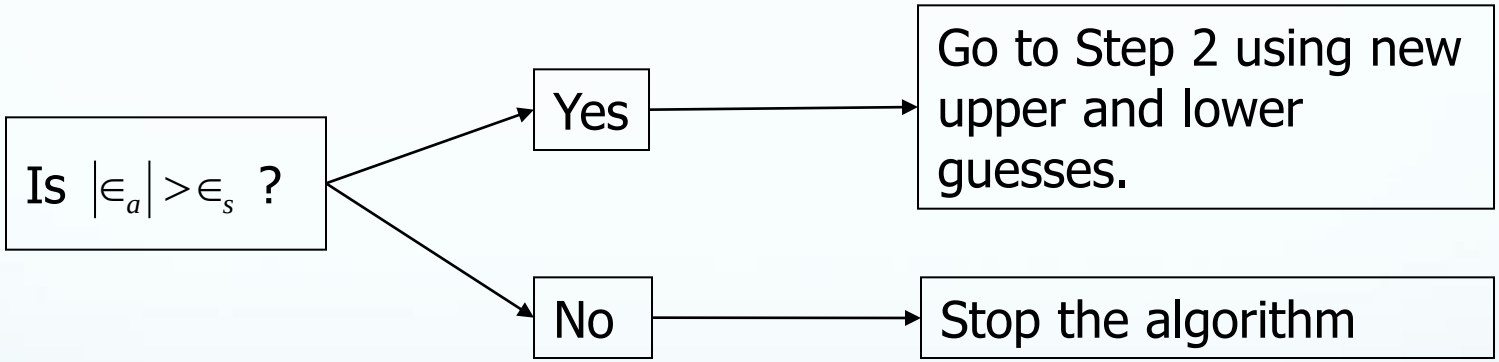
where

$$x_m^{old} = \text{previous estimate of root}$$

$$x_m^{new} = \text{current estimate of root}$$

Step 5

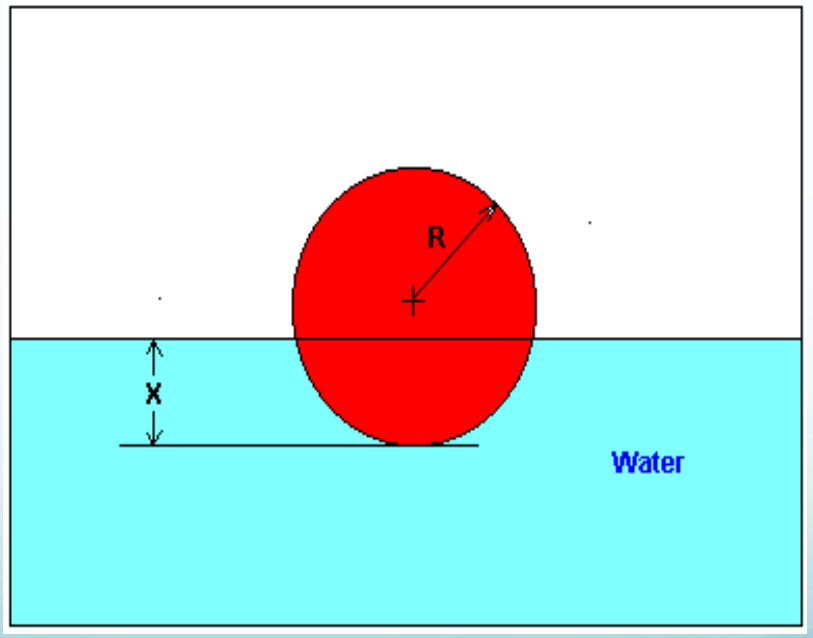
Compare the absolute relative approximate error $|\epsilon_a|$ with the pre-specified error tolerance ϵ_s.



Note one should also check whether the number of iterations is more than the maximum number of iterations allowed. If so, one needs to terminate the algorithm and notify the user about it.

Example 1

The floating ball has a specific gravity of 0.6 and has a radius of 5.5 cm. You are asked to find the depth to which the ball is submerged when floating in water.



Example 1 Cont.

The equation that gives the depth x to which the ball is submerged under water is given by

$$x^3 - 0.165x^2 + 3.993\times10^{-4} = 0$$

a) Use the bisection method of finding roots of equations to find the depth x to which the ball is submerged under water. Conduct three iterations to estimate the root of the above equation.

b) Find the absolute relative approximate error at the end of each iteration, and the number of significant digits at least correct at the end of each iteration.

Example 1 Cont.

From the physics of the problem, the ball would be submerged between $x = 0$ and $x = 2R$,

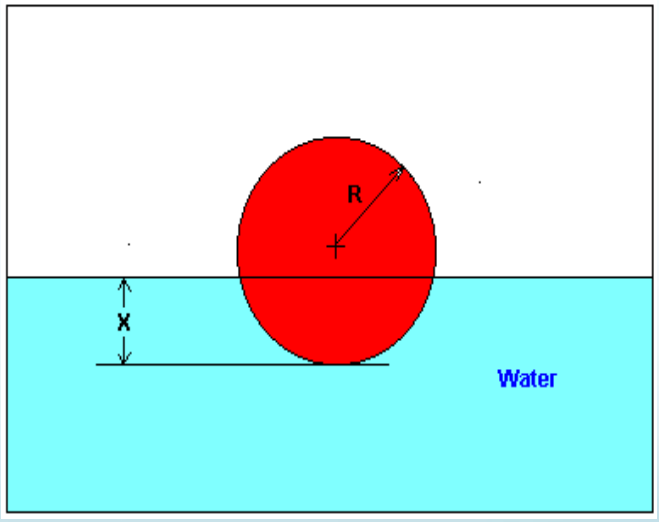
where R = radius of the ball,

that is

$$0 \le x \le 2R$$

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

$$0 \le x \le 0.11$$



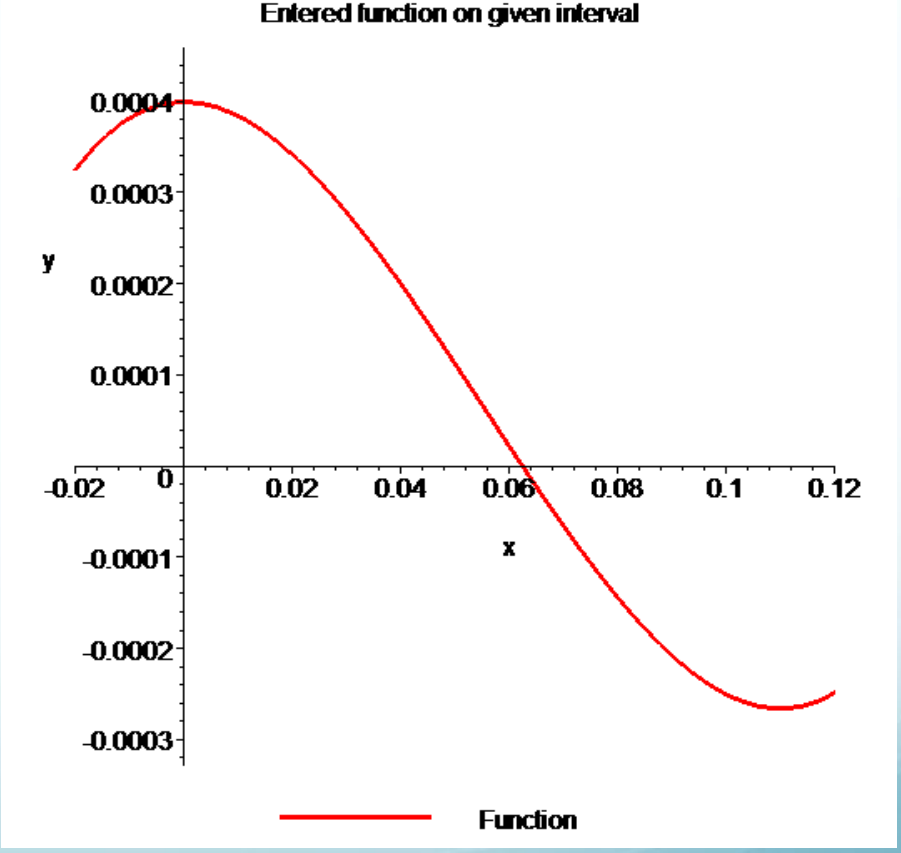
Example 1 Cont.

Solution

To aid in the understanding of how this method works to find the root of an equation, the graph of f(x) is shown to the right,

where

$$f(x) = x^3 - 0.165x^2 + 3.993\times10^{-4}$$



Example 1 Cont.

Let us assume

$$x_\ell = 0.00$$

$$x_u = 0.11$$

Check if the function changes sign between x_ℓ and x_u .

$$f(x_l) = f(0) = (0)^3 - 0.165(0)^2 + 3.993\times10^{-4} = 3.993\times10^{-4}$$

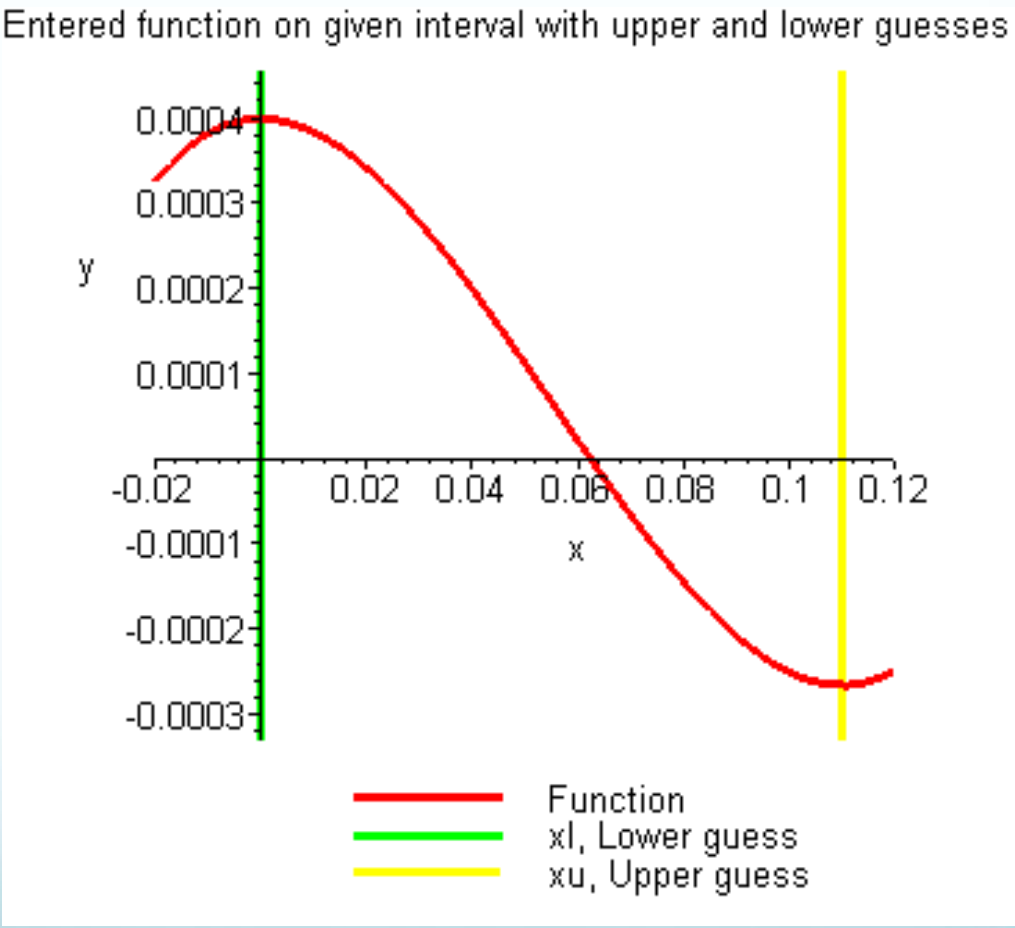
$$f(x_u) = f(0.11) = (0.11)^3 - 0.165(0.11)^2 + 3.993\times10^{-4} = -2.662\times10^{-4}$$

Hence

$$f(x_l)f(x_u) = f(0)f(0.11) = \left(3.993\times10^{-4}\right)\left(-2.662\times10^{-4}\right) < 0$$

So there is at least on root between x_ℓ and x_u, that is between 0 and 0.11

Example 1 Cont.



Graph demonstrating sign change between initial limits

Example 1 Cont.

Iteration 1

The estimate of the root is $x_m = \frac{x_\ell + x_u}{2} = \frac{0+0.11}{2} = 0.055$

$$f(x_m) = f(0.055) = (0.055)^3 - 0.165(0.055)^2 + 3.993\times10^{-4} = 6.655\times10^{-5}$$

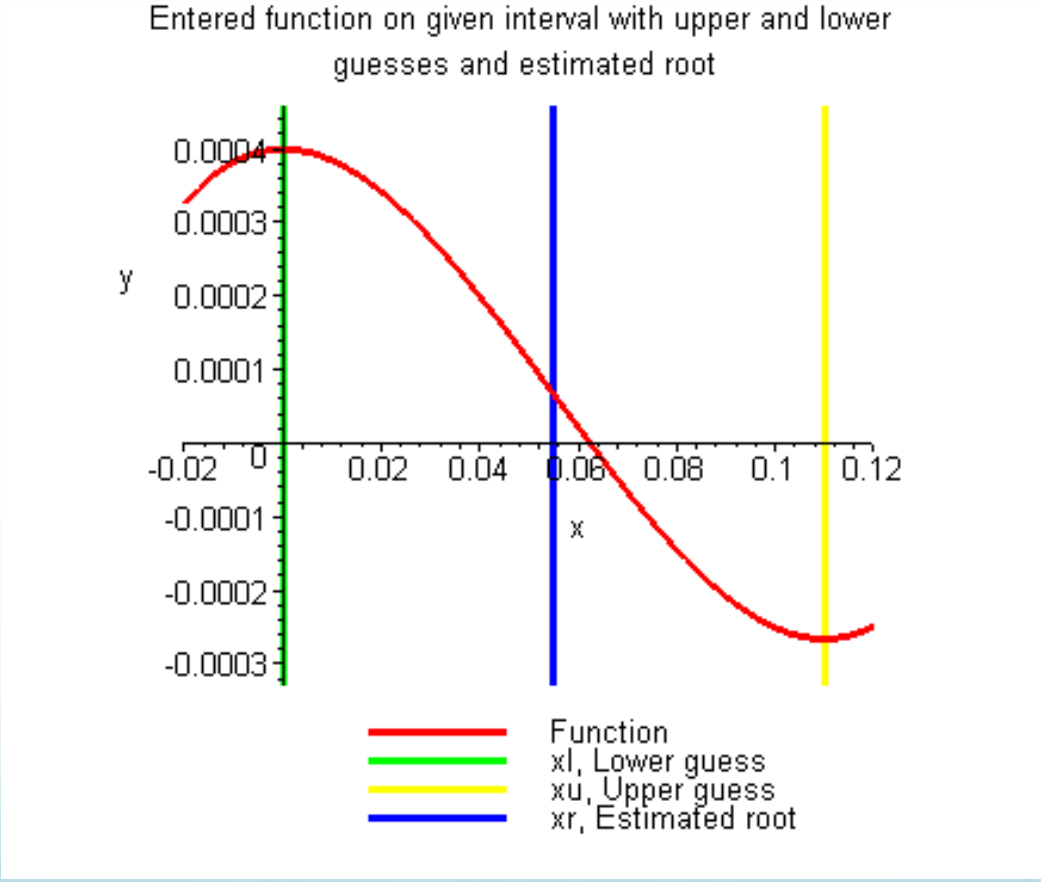
$$f(x_l)f(x_m) = f(0)f(0.055) = \left(3.993\times10^{-4}\right)\left(6.655\times10^{-5}\right) > 0$$

Hence the root is bracketed between x_m and x_u, that is, between 0.055 and 0.11. So, the lower and upper limits of the new bracket are

$$x_l = 0.055,\ x_u = 0.11$$

At this point, the absolute relative approximate error $|\in_a|$ cannot be calculated as we do not have a previous approximation.

Example 1 Cont.



Estimate of the root for Iteration 1

Example 1 Cont.

Iteration 2

The estimate of the root is $x_m = \dfrac{x_\ell + x_u}{2} = \dfrac{0.055 + 0.11}{2} = 0.0825$

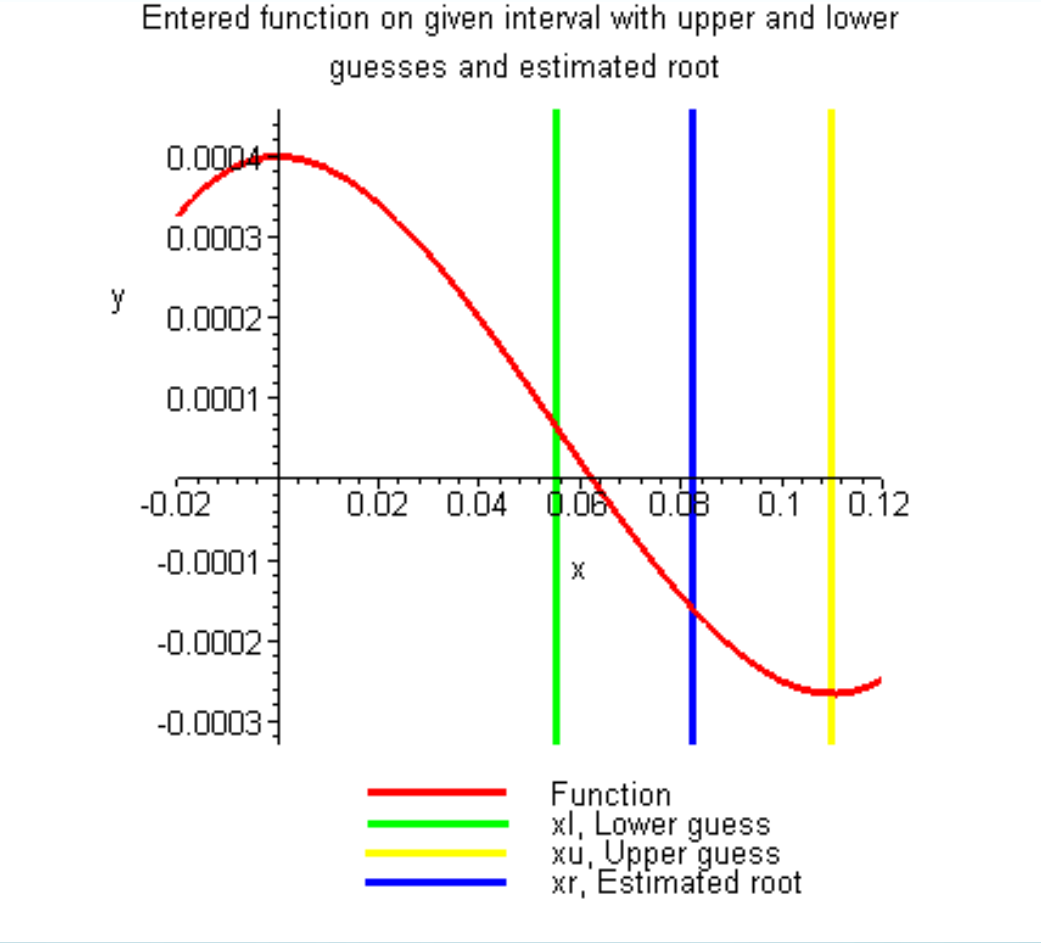
$$f(x_m) = f(0.0825) = (0.0825)^3 - 0.165(0.0825)^2 + 3.993 \times 10^{-4} = -1.622 \times 10^{-4}$$

$$f(x_l) f(x_m) = f(0.055) f(0.0825) = \left(-1.622 \times 10^{-4}\right)\left(6.655 \times 10^{-5}\right) < 0$$

Hence the root is bracketed between x_ℓ and x_m, that is, between 0.055 and 0.0825. So, the lower and upper limits of the new bracket are

$$x_l = 0.055, \quad x_u = 0.0825$$

Example 1 Cont.



Estimate of the root for Iteration 2

Example 1 Cont.

The absolute relative approximate error $|\epsilon_a|$ at the end of Iteration 2 is

$$|\epsilon_a| = \left| \frac{x_m^{new} - x_m^{old}}{x_m^{new}} \right| \times 100$$

$$= \left| \frac{0.0825 - 0.055}{0.0825} \right| \times 100$$

$$= 33.333\%$$

Example 1 Cont.

Iteration 3

The estimate of the root is $x_m = \frac{x_\ell + x_u}{2} = \frac{0.055 + 0.0825}{2} = 0.06875$

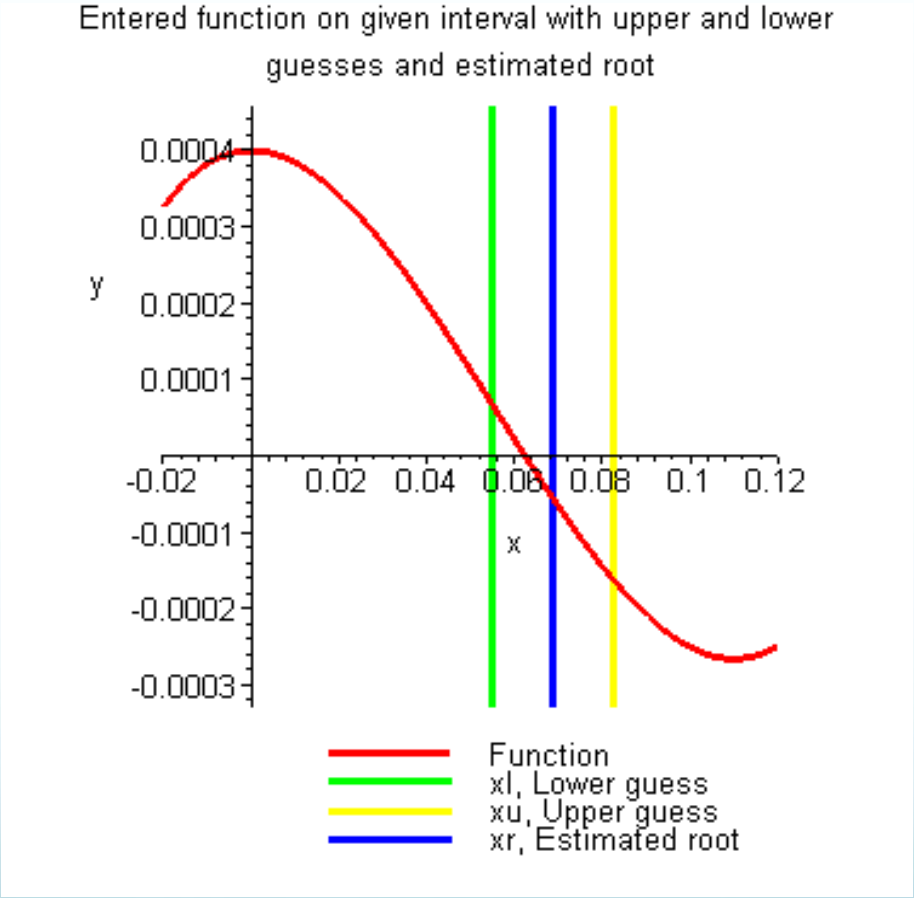
$$f(x_m) = f(0.06875) = (0.06875)^3 - 0.165(0.06875)^2 + 3.993\times10^{-4} = -5.563\times10^{-5}$$

$$f(x_l)f(x_m) = f(0.055)f(0.06875) = (6.655\times10^{-5})(-5.563\times10^{-5}) < 0$$

Hence the root is bracketed between x_ℓ and x_m, that is, between 0.055 and 0.06875. So, the lower and upper limits of the new bracket are

$$x_l = 0.055, \quad x_u = 0.06875$$

Example 1 Cont.



Estimate of the root for Iteration 3

Example 1 Cont.

The absolute relative approximate error $|\epsilon_a|$ at the end of Iteration 3 is

$$|\epsilon_a| = \left| \frac{x_m^{new} - x_m^{old}}{x_m^{new}} \right| \times 100$$

$$= \left| \frac{0.06875 - 0.0825}{0.06875} \right| \times 100$$

$$= 20\%$$

Table 1 Cont.

Root of f(x)=0 as function of number of iterations for bisection method:

Iteration	x_ℓ	x_u	x_m	$\lvert\epsilon_a\rvert$ %	$f(x_m)$
1	0.00000	0.11	0.055	----------	6.655×10^{-5}
2	0.055	0.11	0.0825	33.33	-1.622×10^{-4}
3	0.055	0.0825	0.06875	20.00	-5.563×10^{-5}
4	0.055	0.06875	0.06188	11.11	4.484×10^{-6}
5	0.06188	0.06875	0.06531	5.263	-2.593×10^{-5}
6	0.06188	0.06531	0.06359	2.702	-1.0804×10^{-5}
7	0.06188	0.06359	0.06273	1.370	-3.176×10^{-6}
8	0.06188	0.06273	0.0623	0.6897	6.497×10^{-7}
9	0.0623	0.06273	0.06252	0.3436	-1.265×10^{-6}
10	0.0623	0.06252	0.06241	0.1721	-3.0768×10^{-7}

Table 1 Cont.

Hence the number of significant digits at least correct is given by the largest value or *m* for which

$$\left|\in_a\right| \le 0.5\times 10^{2-m}$$

$$0.1721 \le 0.5\times 10^{2-m}$$

$$0.3442 \le 10^{2-m}$$

$$\log(0.3442) \le 2-m$$

$$m \le 2-\log(0.3442) = 2.463$$

So

$$m = 2$$

The number of significant digits at least correct in the estimated root of 0.06241 at the end of the 10th iteration is 2.

Advantages

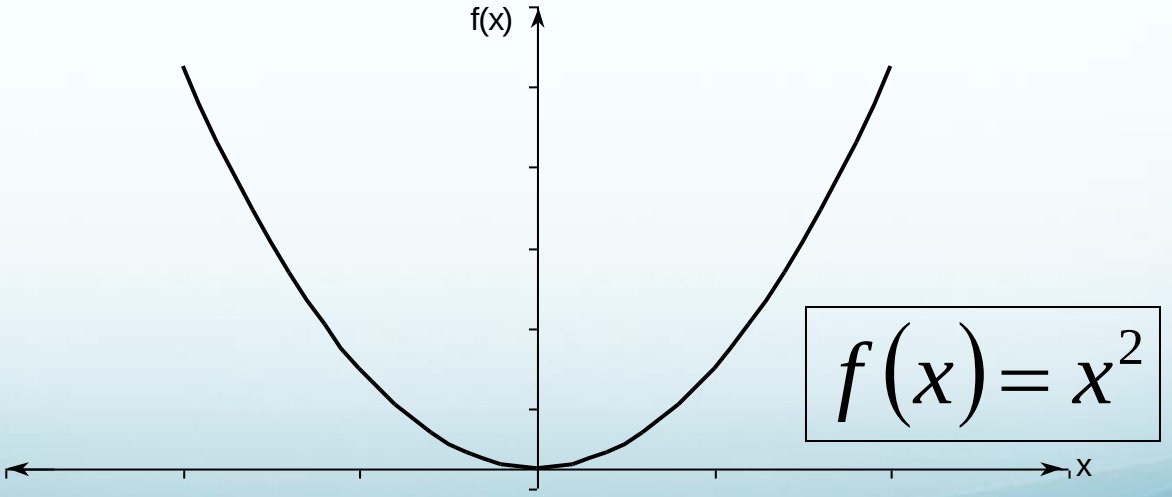
- Always convergent
- The root bracket gets halved with each iteration - guaranteed.

Drawbacks

- Slow convergence
- If one of the initial guesses is close to the root, the convergence is slower

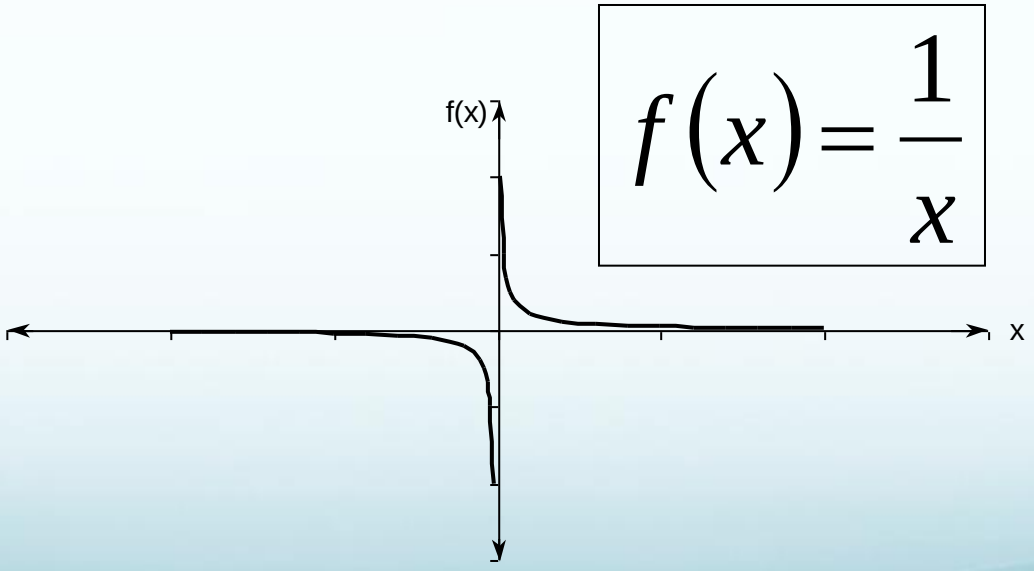
Drawbacks (continued)

- If a function f(x) is such that it just touches the x-axis it will be unable to find the lower and upper guesses.



Drawbacks (continued)

- Function changes sign but root does not exist



Algorithm

```
procedure Bisection(f, a, b, nmax, ε)
integer n, nmax;    real a, b, c, fa, fb, fc, error
fa ← f(a)
fb ← f(b)
if sign(fa) = sign(fb) then
      output a, b, fa, fb
      output "function has same signs at a and b"
      return
end if
error ← b − a
for n = 0 to nmax do
      error ← error/2
      c ← a + error
      fc ← f(c)
      output n, c, fc, error
      if |error| < ε then
            output "convergence"
            return
      end if
      if sign(fa) ≠ sign(fc) then
            b ← c
            fb ← fc
      else
            a ← c
            fa ← fc
      end if
end for
end procedure Bisection
```

Example of the Algorithm

$f(x) = x^3 - 3x + 1$ on $[0, 1]$

$g(x) = x^3 - 2\sin x$ on $[0.5, 2]$

```
program Test_Bisection
integer n, nmax ← 20
real a, b, ε ← ½10^−6
external function f, g
a ← 0.0
b ← 1.0
call Bisection(f, a, b, nmax, ε)
a ← 0.5
b ← 2.0
call Bisection(g, a, b, nmax, ε)
end program Test_Bisection

real function f(x)
real x
f ← x^3 − 3x + 1
end function f

real function g(x)
real x
g ← x^3 − 2 sin x
end function g
```

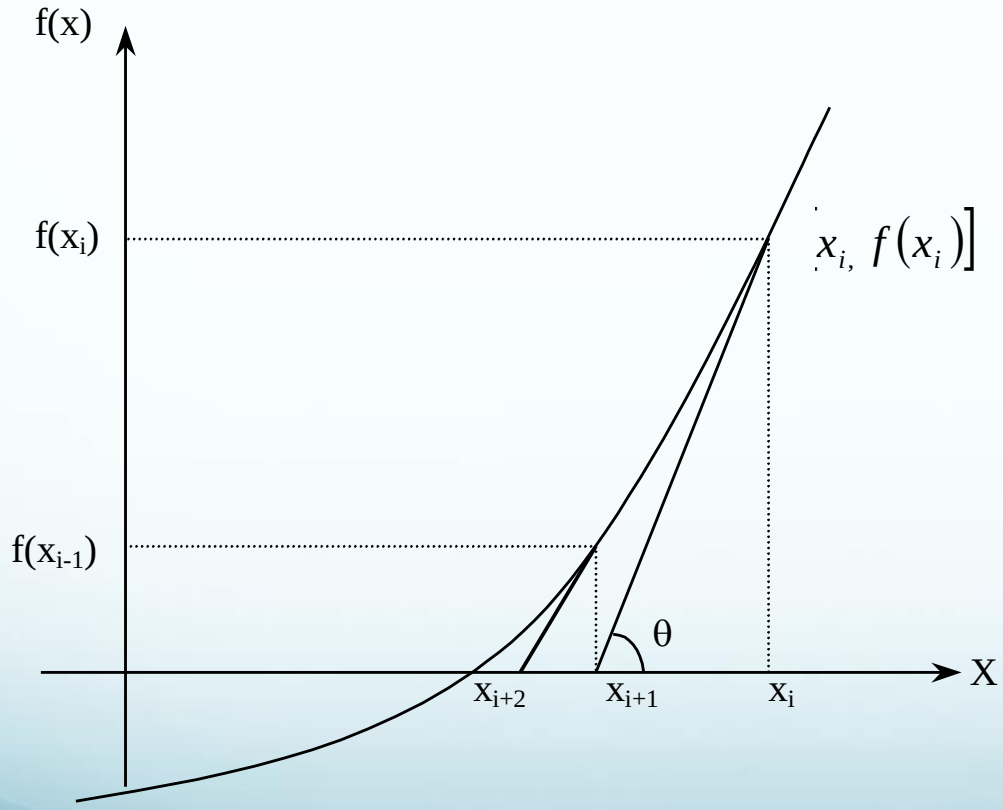
The computer results for the iterative steps of the bisection method for $f(x)$:

n	c_n	$f(c_n)$	error
0	0.5	-0.375	0.5
1	0.25	0.266	0.25
2	0.375	-7.23×10^{-2}	0.125
3	0.3125	9.30×10^{-2}	6.25×10^{-2}
4	0.34375	9.37×10^{-3}	3.125×10^{-2}
⋮			
19	0.3472967	-9.54×10^{-7}	9.54×10^{-7}
20	0.3472962	3.58×10^{-7}	4.77×10^{-7}

Also, the results for $g(x)$ are as follows:

n	c_n	$g(c_n)$	error
0	1.25	5.52×10^{-2}	0.75
1	0.875	-0.865	0.375
2	1.0625	-0.548	0.188
3	1.15625	-0.285	9.38×10^{-2}
4	1.203125	-0.125	4.69×10^{-2}
⋮			
19	1.2361827	-4.88×10^{-6}	1.43×10^{-6}
20	1.2361834	-2.15×10^{-6}	7.15×10^{-7}

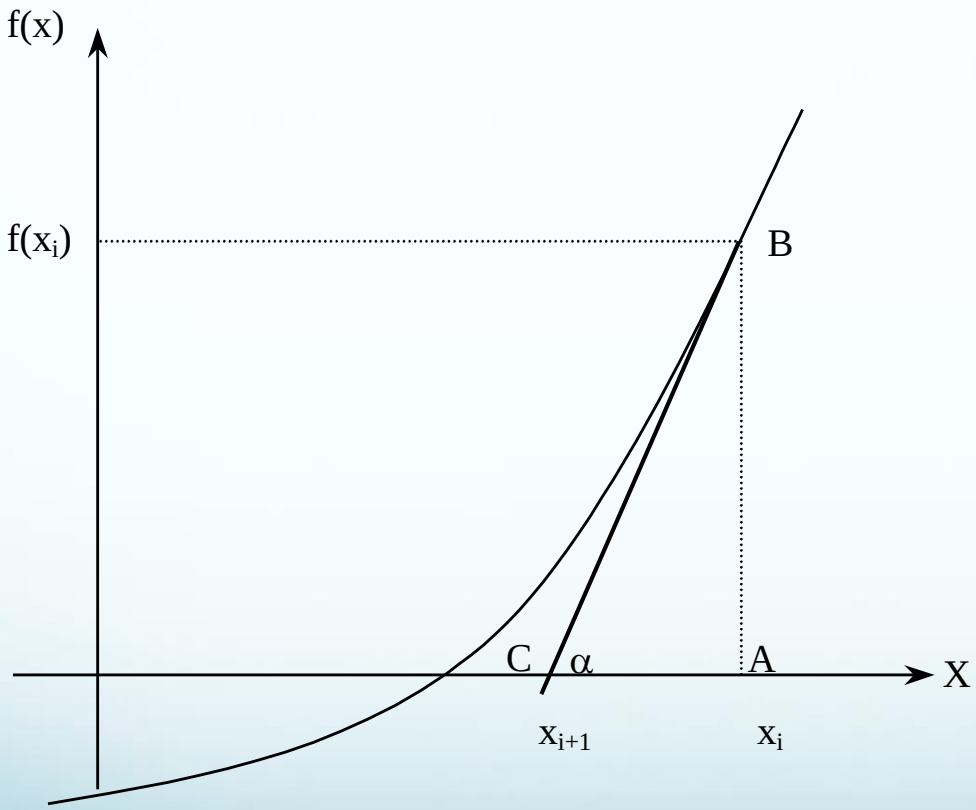
Newton-Raphson Method



$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Geometrical illustration of the Newton-Raphson method.

Derivation



$$\tan(\alpha) = \frac{AB}{AC}$$

$$f'(x_i) = \frac{f(x_i)}{x_i - x_{i+1}}$$

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Derivation of the Newton-Raphson method.

Algorithm for Newton-Raphson Method

Step 1

Evaluate $f'(x)$ symbolically.

Step 2

Use an initial guess of the root, x_i , to estimate the new value of the root, x_{i+1} , as

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

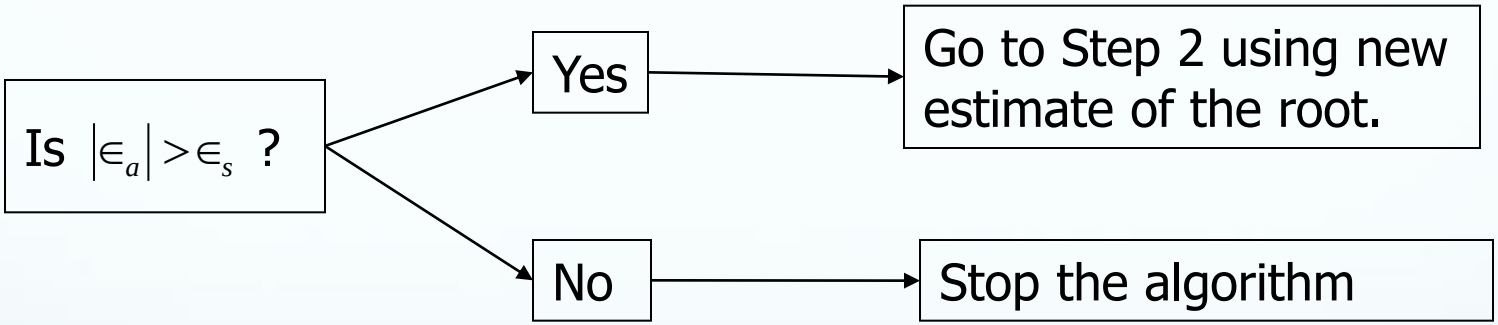
Step 3

Find the absolute relative approximate error $|\epsilon_a|$ as

$$|\epsilon_a| = \left| \frac{x_{i+1} - x_i}{x_{i+1}} \right| \times 100$$

Step 4

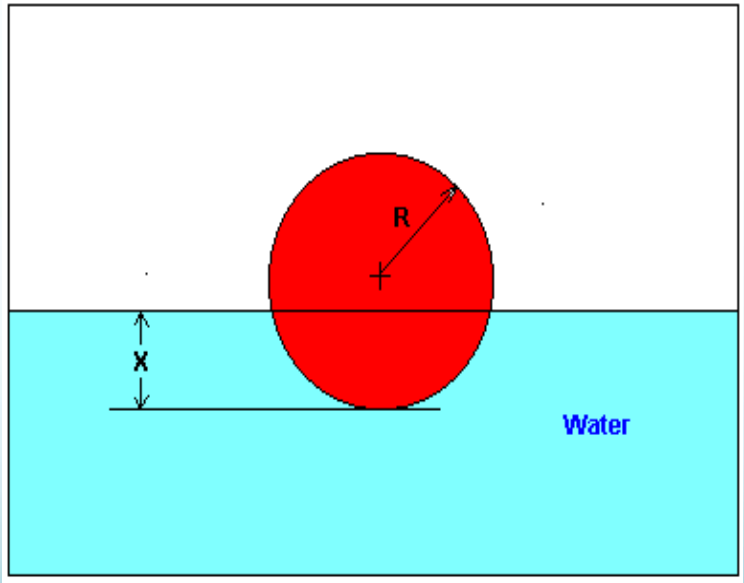
Compare the absolute relative approximate error with the pre-specified relative error tolerance $\in_s$.



Also, check if the number of iterations has exceeded the maximum number of iterations allowed. If so, one needs to terminate the algorithm and notify the user.

Example 1

The floating ball has a specific gravity of 0.6 and has a radius of 5.5 cm. You are asked to find the depth to which the ball is submerged when floating in water.

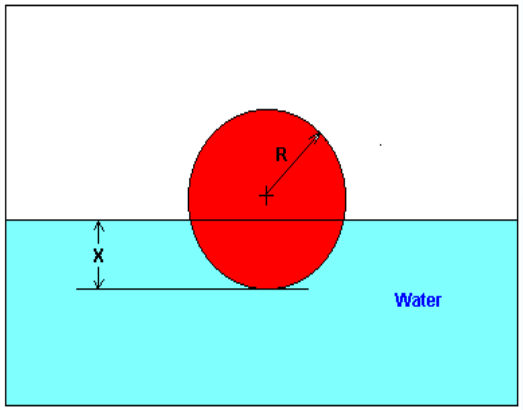


Floating ball problem.

Example 1 Cont.

The equation that gives the depth x in meters to which the ball is submerged under water is given by

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



Floating ball problem.

Use the Newton's method of finding roots of equations to find
a)the depth 'x' to which the ball is submerged under water. Conduct three iterations to estimate the root of the above equation.
b)The absolute relative approximate error at the end of each iteration, and
c)The number of significant digits at least correct at the end of each iteration.

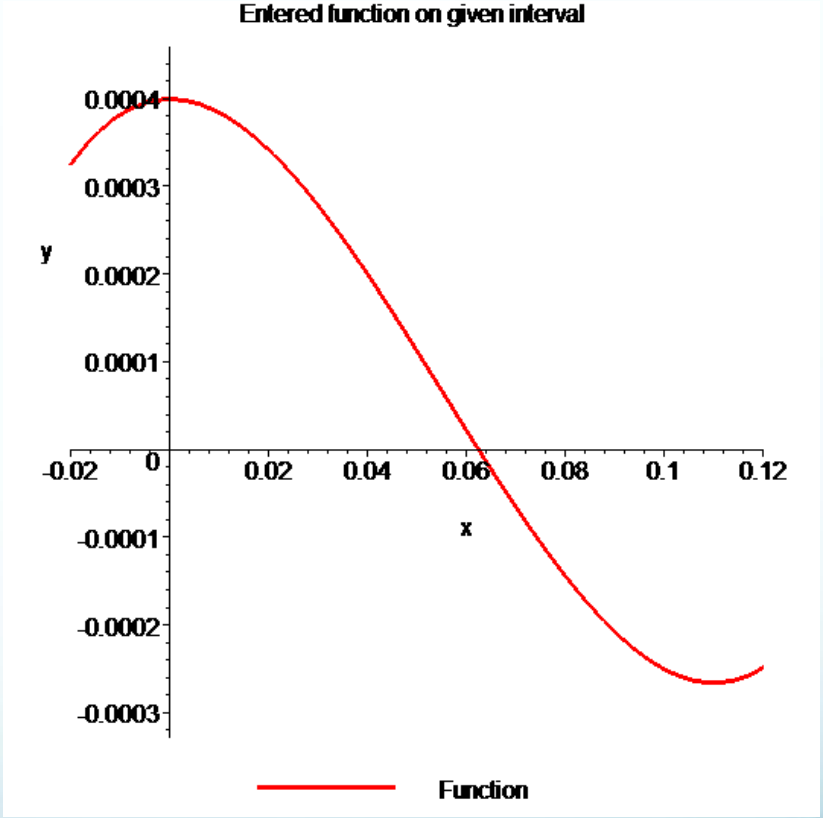
Example 1 Cont.

Solution

To aid in the understanding of how this method works to find the root of an equation, the graph of f(x) is shown to the right,

where

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



Graph of the function f(x)

Example 1 Cont.

Solve for $f'(x)$

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

$$f'(x) = 3x^2 - 0.33x$$

Let us assume the initial guess of the root of $f(x) = 0$ is $x_0 = 0.05\text{m}$.

Example 1 Cont.

Iteration 1

The estimate of the root is

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

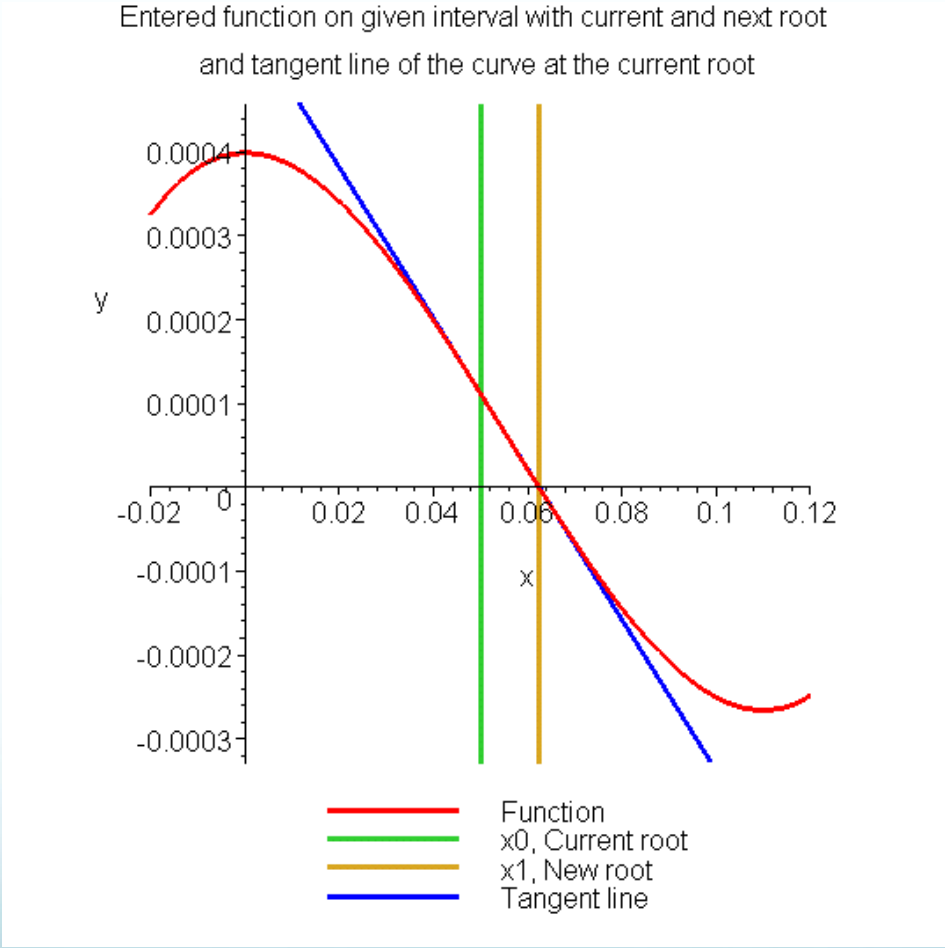
$$= 0.05 - \frac{(0.05)^3 - 0.165(0.05)^2 + 3.993\times10^{-4}}{3(0.05)^2 - 0.33(0.05)}$$

$$= 0.05 - \frac{1.118\times10^{-4}}{-9\times10^{-3}}$$

$$= 0.05 - (-0.01242)$$

$$= 0.06242$$

Example 1 Cont.



Estimate of the root for the first iteration.

Example 1 Cont.

The absolute relative approximate error $|\in_a|$ at the end of Iteration 1 is

$$|\in_a| = \left|\frac{x_1 - x_0}{x_1}\right| \times 100$$

$$= \left|\frac{0.06242 - 0.05}{0.06242}\right| \times 100$$

$$= 19.90\%$$

Example 1 Cont.

Iteration 2

The estimate of the root is

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

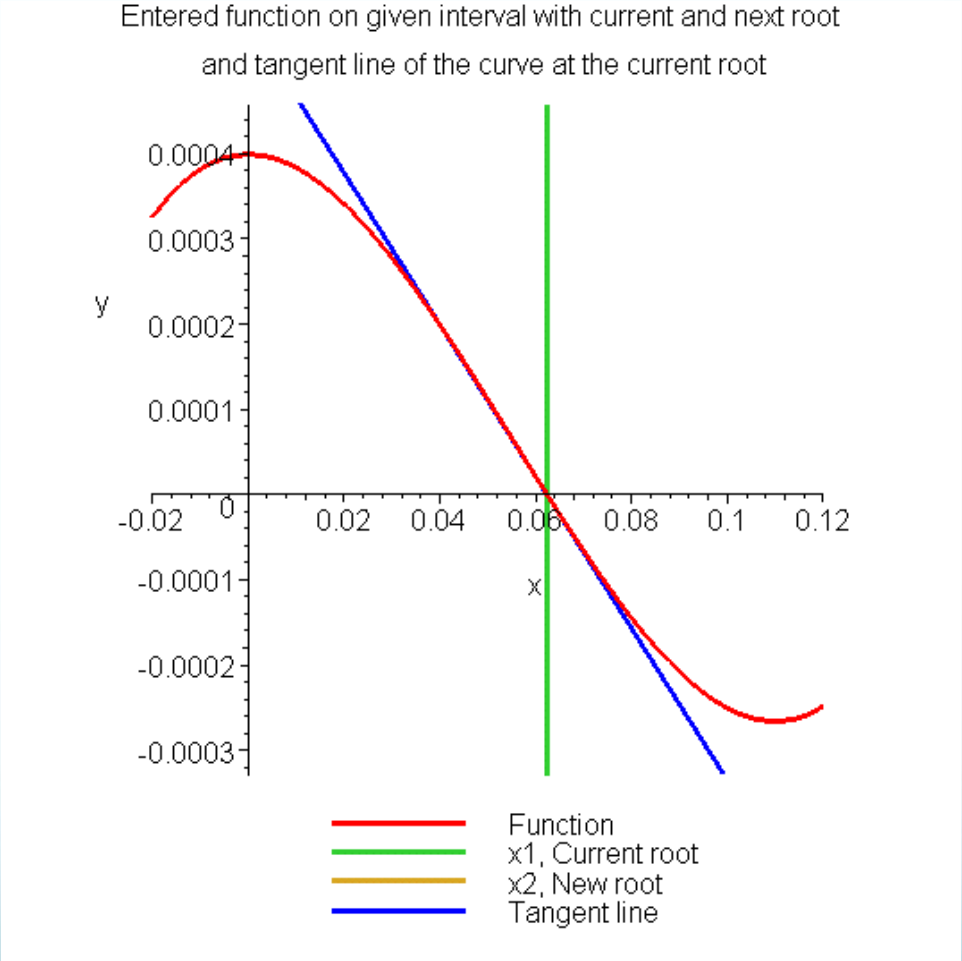
$$= 0.06242 - \frac{(0.06242)^3 - 0.165(0.06242)^2 + 3.993\times10^{-4}}{3(0.06242)^2 - 0.33(0.06242)}$$

$$= 0.06242 - \frac{-3.97781\times10^{-7}}{-8.90973\times10^{-3}}$$

$$= 0.06242 - \left(4.4646\times10^{-5}\right)$$

$$= 0.06238$$

Example 1 Cont.



Estimate of the root for the Iteration 2.

Example 1 Cont.

The absolute relative approximate error $|\epsilon_a|$ at the end of Iteration 2 is

$$|\epsilon_a| = \left|\frac{x_2 - x_1}{x_2}\right| \times 100$$

$$= \left|\frac{0.06238 - 0.06242}{0.06238}\right| \times 100$$

$$= 0.0716\%$$

Example 1 Cont.

Iteration 3

The estimate of the root is

$$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)}$$

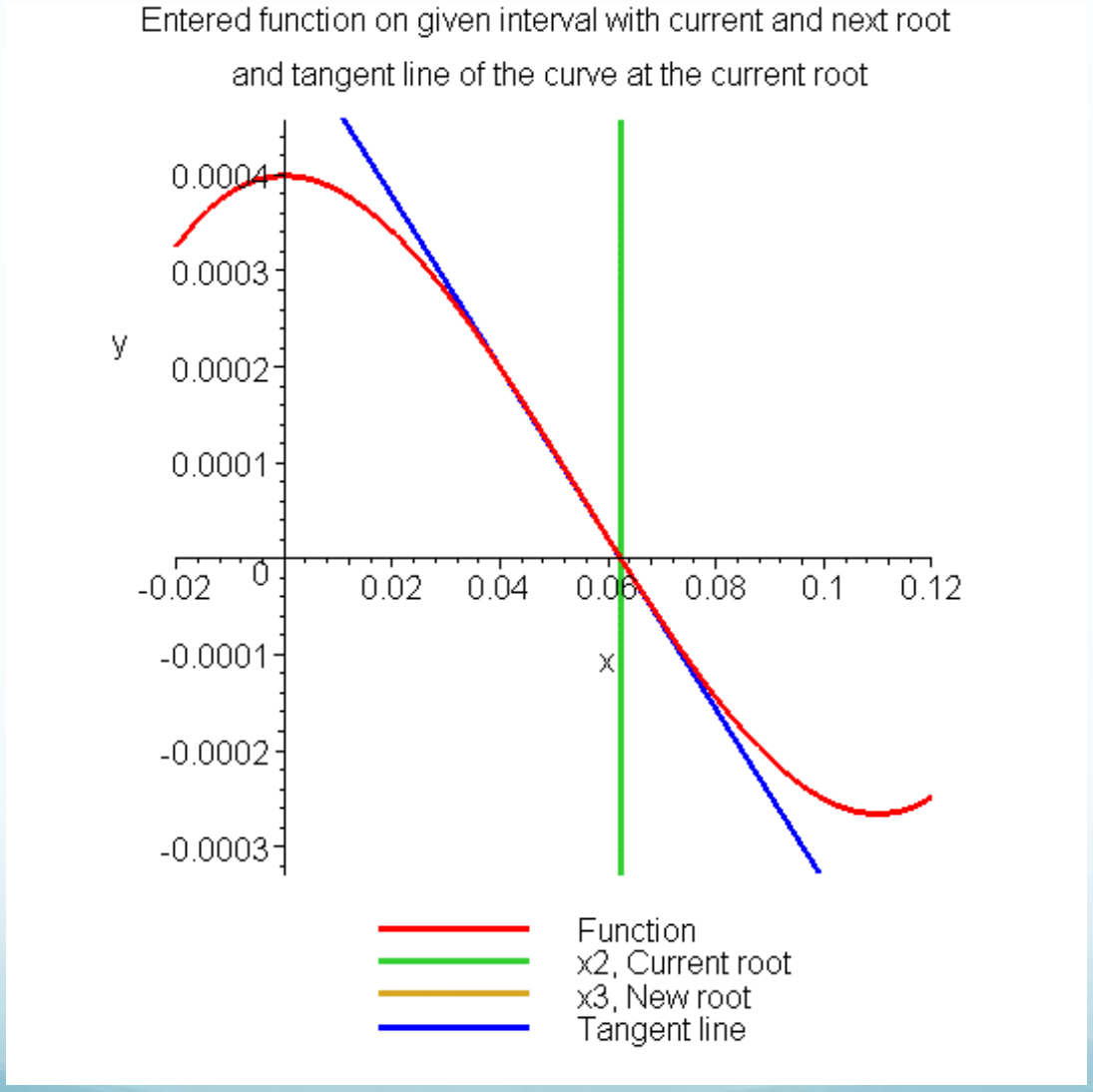
$$= 0.06238 - \frac{(0.06238)^3 - 0.165(0.06238)^2 + 3.993\times10^{-4}}{3(0.06238)^2 - 0.33(0.06238)}$$

$$= 0.06238 - \frac{4.44\times10^{-11}}{-8.91171\times10^{-3}}$$

$$= 0.06238 - \left(-4.9822\times10^{-9}\right)$$

$$= 0.06238$$

Example 1 Cont.



Estimate of the root for the Iteration 3.

Example 1 Cont.

The absolute relative approximate error $|\epsilon_a|$ at the end of Iteration 3 is

$$|\epsilon_a| = \left|\frac{x_2 - x_1}{x_2}\right| \times 100$$

$$= \left|\frac{0.06238 - 0.06238}{0.06238}\right| \times 100$$

$$= 0\%$$

Algorithm

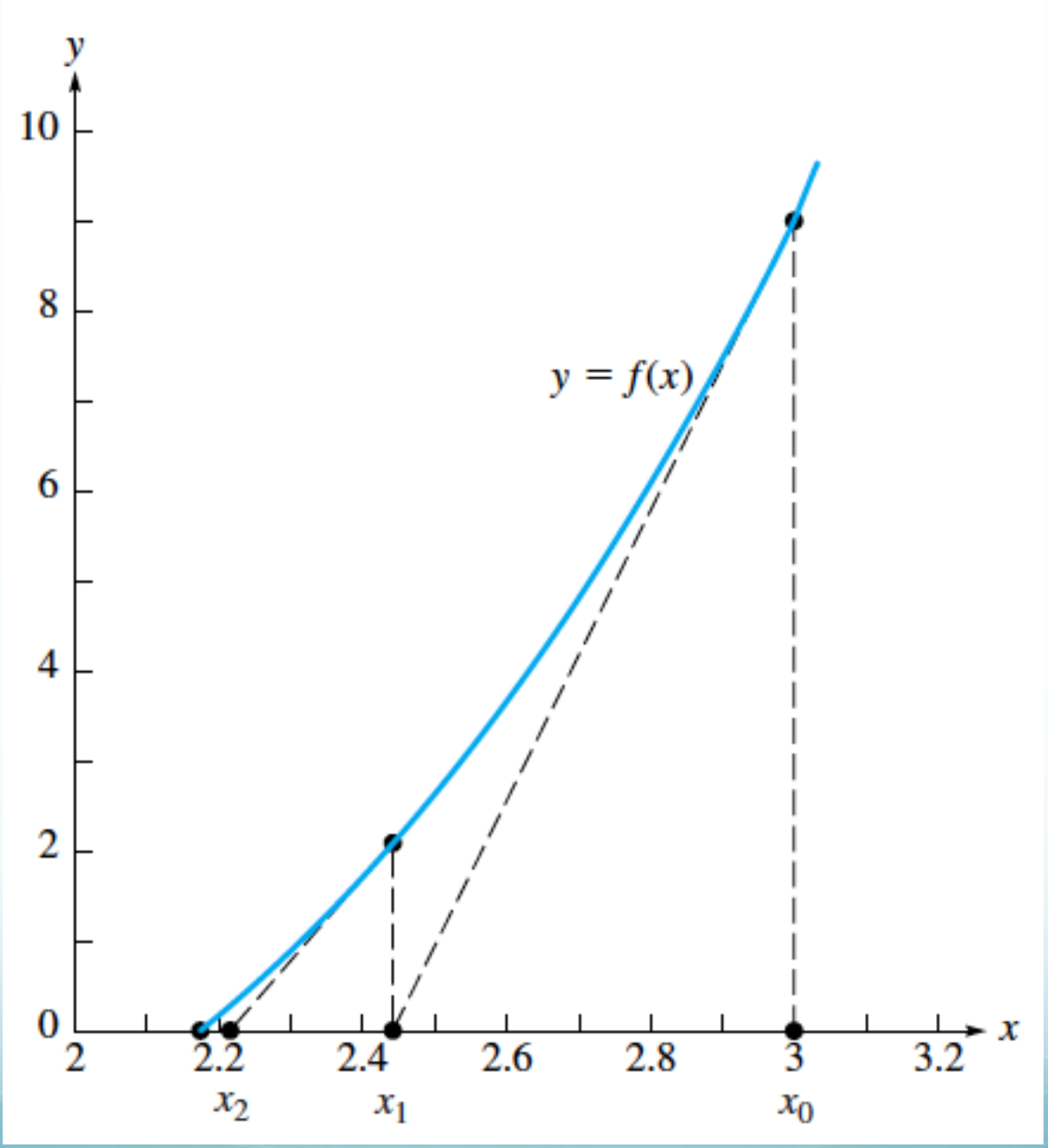
```
procedure Newton(f, f', x, nmax, ε, δ)
integer n, nmax;    real x, fx, fp, ε, δ
external function f, f'
fx ← f(x)
output 0, x, fx
for n = 1 to nmax do
    fp ← f'(x)
    if |fp| < δ then
        output "small derivative"
        return
    end if
    d ← fx/fp
    x ← x − d
    fx ← f(x)
    output n, x, fx
    if |d| < ε then
        output "convergence"
        return
    end if
end for
end procedure Newton
```

Example of the Algorithm

$f(x) = x^3 - 2x^2 + x - 3$ starting with $x_0 = 3$

$$f(x) = ((x - 2)x + 1)x - 3$$
$$f'(x) = (3x - 4)x + 1$$

n	x_n	$f(x_n)$
0	3.0	9.0
1	2.4375	2.04
2	2.21303 27224 73144 5	0.256
3	2.17555 49386 14368 4	6.46×10^{-3}
4	2.17456 01006 55071 4	4.48×10^{-6}
5	2.17455 94102 93284 1	1.97×10^{-12}


y
10
8
6
4
2
0
y = f(x)
x
2
2.2
2.4
2.6
2.8
3
3.2
x2
x1
x0

Advantages

- Converges fast (quadratic convergence), if it converges.
- Requires only one guess

Drawbacks

1. Divergence at inflection points
   Selection of the initial guess or an iteration value of the root that is close to the inflection point of the function $f(x)$ may start diverging away from the root in the Newton-Raphson method.

For example, to find the root of the equation $f(x)=(x-1)^3+0.512=0$.

The Newton-Raphson method reduces to $x_{i+1}=x_i-\dfrac{(x_i^3-1)^3+0.512}{3(x_i-1)^2}$.

Table 1 shows the iterated values of the root of the equation.

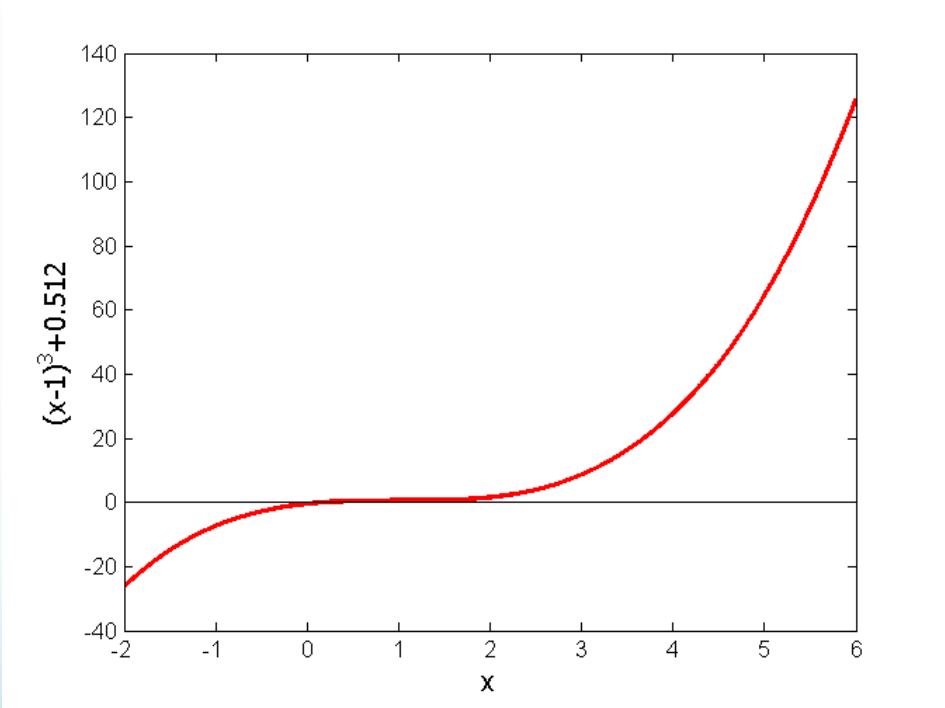
The root starts to diverge at Iteration 6 because the previous estimate of 0.92589 is close to the inflection point of $x=1$.

Eventually after 12 more iterations the root converges to the exact value of $x=0.2$.

Drawbacks – Inflection Points

Table 1 Divergence near inflection point.

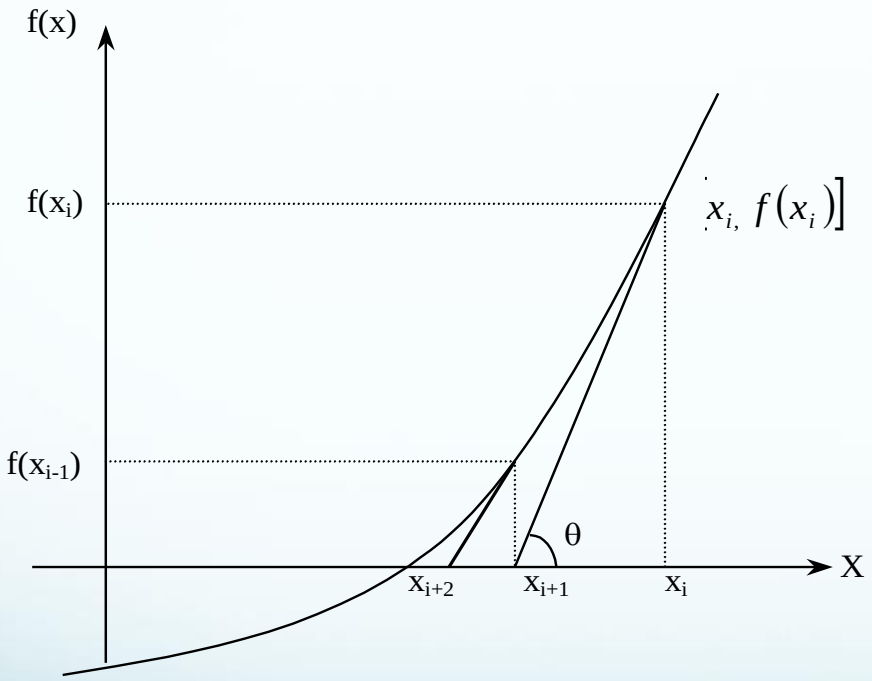
Iteration Number	x_i
0	5.0000
1	3.6560
2	2.7465
3	2.1084
4	1.6000
5	0.92589
6	−30.119
7	−19.746
18	0.2000



Divergence at inflection point for

$$f(x)=(x-1)^3+0.512=0$$

Secant Method – Derivation



Geometrical illustration of the Newton-Raphson method.

Newton's Method

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \qquad (1)$$

Approximate the derivative

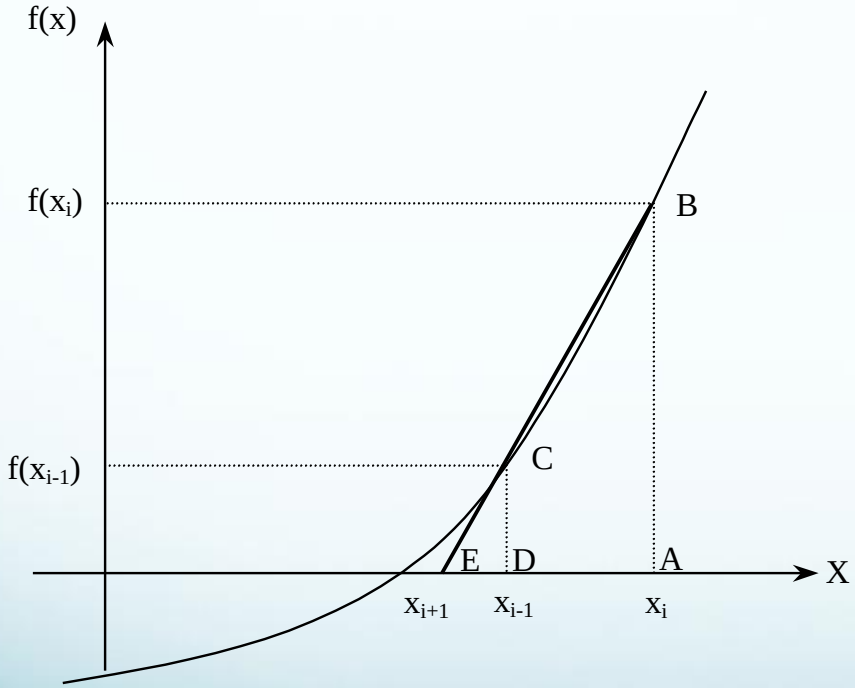
$$f'(x_i) = \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \qquad (2)$$

Substituting Equation (2) into Equation (1) gives the Secant method

$$x_{i+1} = x_i - \frac{f(x_i)(x_i - x_{i-1})}{f(x_i) - f(x_{i-1})}$$

Secant Method – Derivation

The secant method can also be derived from geometry:



Geometrical representation of the Secant method.

The Geometric Similar Triangles

$$\frac{AB}{AE} = \frac{DC}{DE}$$

can be written as

$$\frac{f(x_i)}{x_i - x_{i+1}} = \frac{f(x_{i-1})}{x_{i-1} - x_{i+1}}$$

On rearranging, the secant method is given as

$$x_{i+1} = x_i - \frac{f(x_i)(x_i - x_{i-1})}{f(x_i) - f(x_{i-1})}$$

Algorithm for Secant Method

Step 1

Calculate the next estimate of the root from two initial guesses

$$x_{i+1} = x_i - \frac{f(x_i)(x_i - x_{i-1})}{f(x_i) - f(x_{i-1})}$$

Find the absolute relative approximate error

$$\left|\epsilon_a\right| = \left|\frac{x_{i+1} - x_i}{x_{i+1}}\right| \times 100$$

Step 2

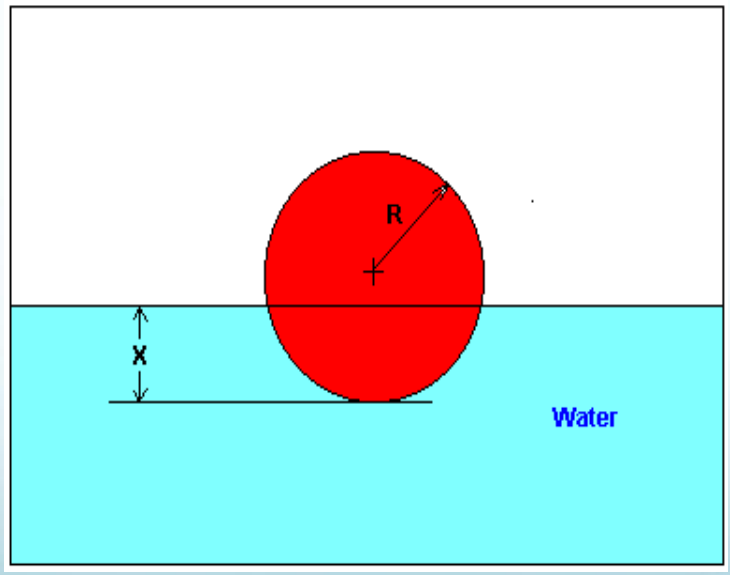
Find if the absolute relative approximate error is greater than the prespecified relative error tolerance.

If so, go back to step 1, else stop the algorithm.

Also check if the number of iterations has exceeded the maximum number of iterations.

Example 1

The floating ball has a specific gravity of 0.6 and has a radius of 5.5 cm. You are asked to find the depth to which the ball is submerged when floating in water.



Floating Ball Problem.

Example 1 Cont.

The equation that gives the depth x to which the ball is submerged under water is given by

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

Use the Secant method of finding roots of equations to find the depth x to which the ball is submerged under water.

- Conduct three iterations to estimate the root of the above equation.
- Find the absolute relative approximate error and the number of significant digits at least correct at the end of each iteration.

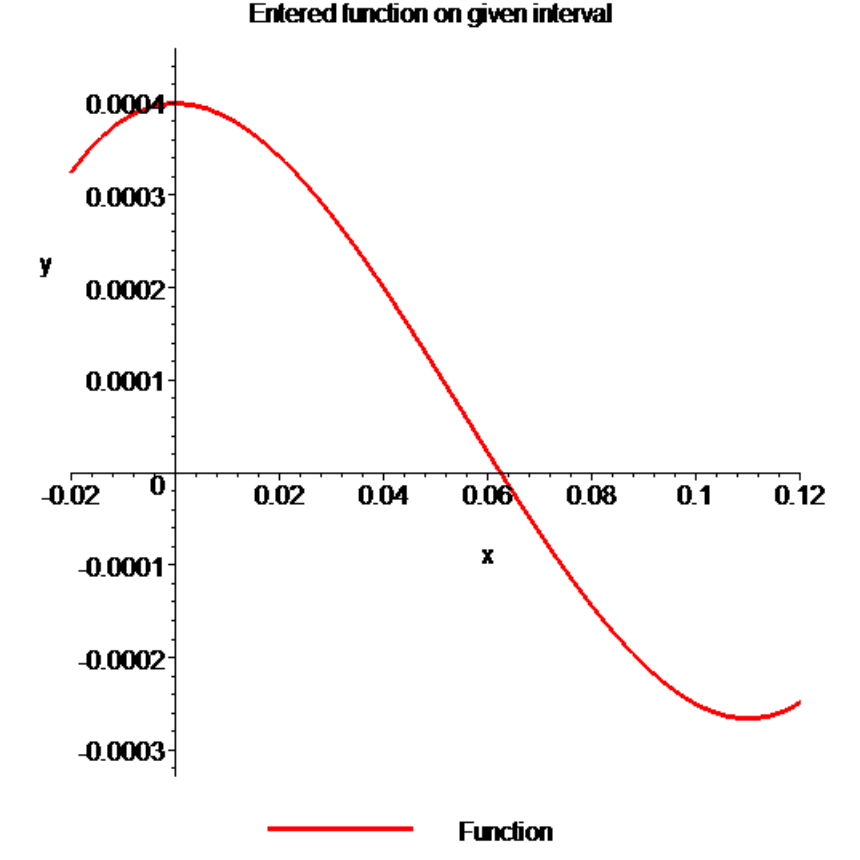
Example 1 Cont.

Solution

To aid in the understanding of how this method works to find the root of an equation, the graph of f(x) is shown to the right,

where

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



Graph of the function f(x).

Example 1 Cont.

Let us assume the initial guesses of the root of $f(x)=0$ as $x_{-1}=0.02$ and $x_0=0.05$.

Iteration 1

The estimate of the root is

$$x_1 = x_0 - \frac{f(x_0)(x_0 - x_{-1})}{f(x_0) - f(x_{-1})}$$

$$= 0.05 - \frac{\left(0.05^3 - 0.165(0.05)^2 + 3.993\times10^{-4}\right)(0.05-0.02)}{\left(0.05^3 - 0.165(0.05)^2 + 3.993\times10^{-4}\right) - \left(0.02^3 - 0.165(0.02)^2 + 3.993\times10^{-4}\right)}$$

$$= 0.06461$$

Example 1 Cont.

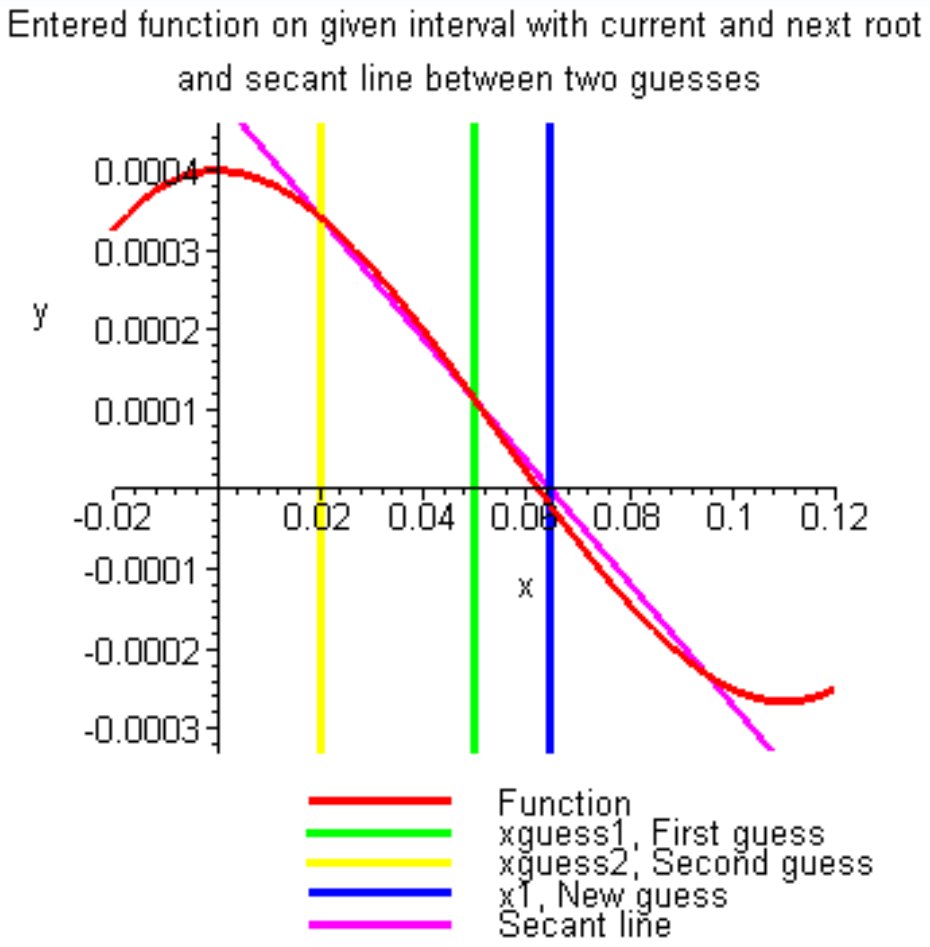
The absolute relative approximate error $|\epsilon_a|$ at the end of Iteration 1 is

$$|\epsilon_a| = \left|\frac{x_1 - x_0}{x_1}\right| \times 100$$

$$= \left|\frac{0.06461 - 0.05}{0.06461}\right| \times 100$$

$$= 22.62\%$$

Example 1 Cont.



Graph of results of Iteration 1.

Example 1 Cont.

Iteration 2

The estimate of the root is

$$x_2 = x_1 - \frac{f(x_1)(x_1 - x_0)}{f(x_1) - f(x_0)}$$

$$= 0.06461 - \frac{\left(0.06461^3 - 0.165(0.06461)^2 + 3.993\times10^{-4}\right)(0.06461 - 0.05)}{\left(0.06461^3 - 0.165(0.06461)^2 + 3.993\times10^{-4}\right) - \left(0.05^3 - 0.165(0.05)^2 + 3.993\times10^{-4}\right)}$$

$$= 0.06241$$

Example 1 Cont.

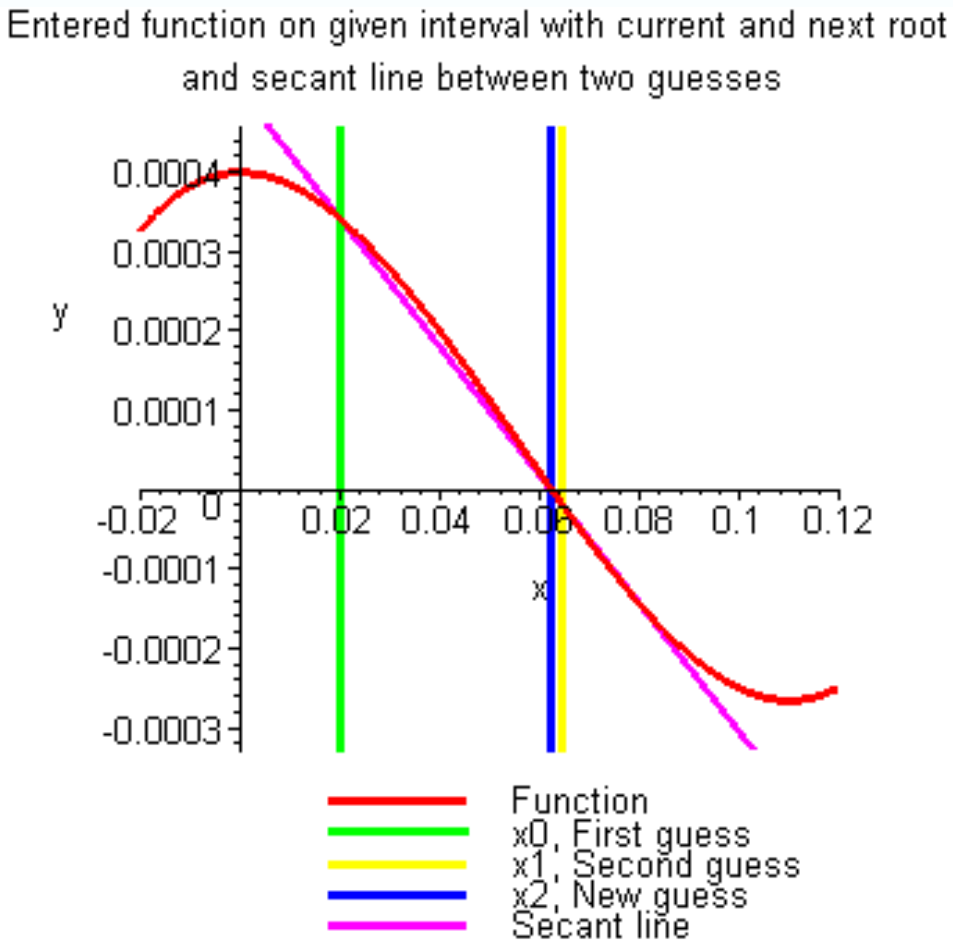
The absolute relative approximate error $|\epsilon_a|$ at the end of Iteration 2 is

$$|\epsilon_a| = \left|\frac{x_2 - x_1}{x_2}\right| \times 100$$

$$= \left|\frac{0.06241 - 0.06461}{0.06241}\right| \times 100$$

$$= 3.525\%$$

Example 1 Cont.



Graph of results of Iteration 2.

Example 1 Cont.

Iteration 3

The estimate of the root is

$$x_3 = x_2 - \frac{f(x_2)(x_2 - x_1)}{f(x_2) - f(x_1)}$$

$$= 0.06241 - \frac{\left(0.06241^3 - 0.165(0.06241)^2 + 3.993\times10^{-4}\right)(0.06241 - 0.06461)}{\left(0.06241^3 - 0.165(0.06241)^2 + 3.993\times10^{-4}\right) - \left(0.05^3 - 0.165(0.06461)^2 + 3.993\times10^{-4}\right)}$$

$$= 0.06238$$

Example 1 Cont.

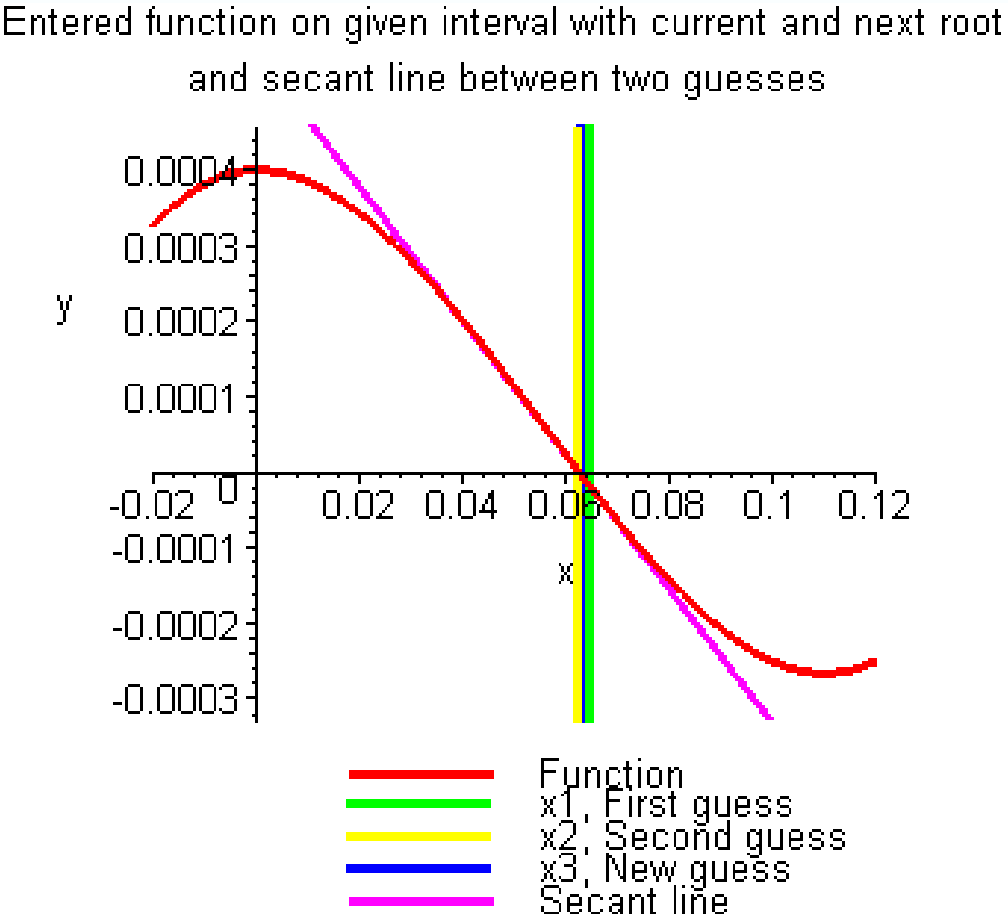
The absolute relative approximate error $|\epsilon_a|$ at the end of Iteration 3 is

$$|\epsilon_a| = \left|\frac{x_3 - x_2}{x_3}\right| \times 100$$

$$= \left|\frac{0.06238 - 0.06241}{0.06238}\right| \times 100$$

$$= 0.0595\%$$

Iteration #3

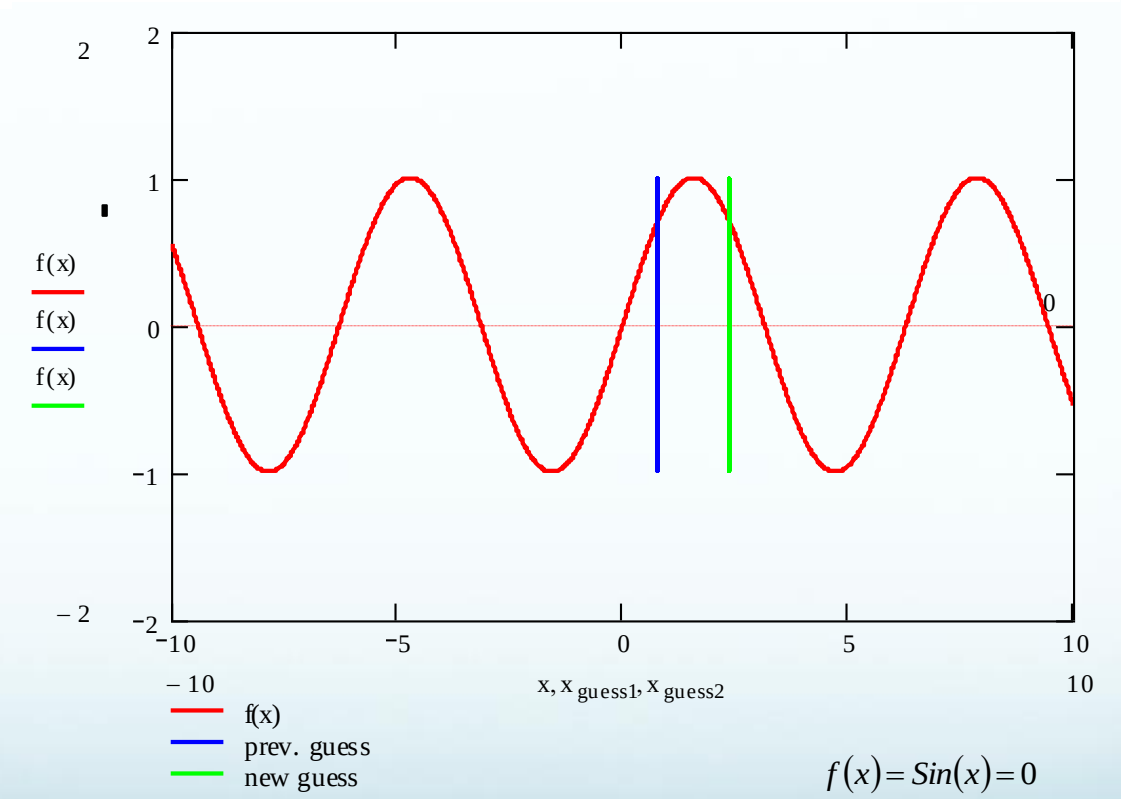


Graph of results of Iteration 3.

Advantages

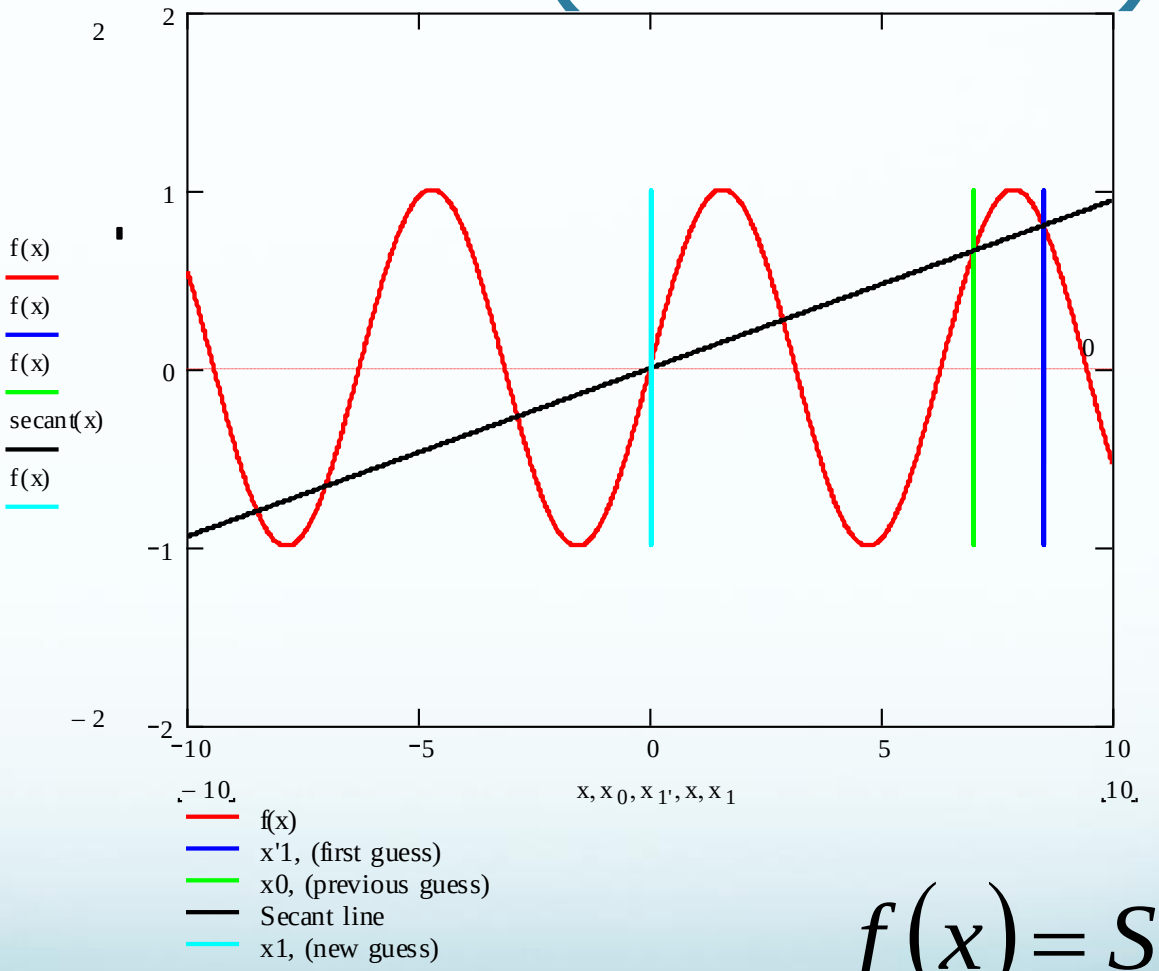
- Converges fast, if it converges
- Requires two guesses that do not need to bracket the root

Drawbacks



Division by zero

Drawbacks (continued)



$$f(x) = Sinx = 0$$

Root Jumping

Algorithm

```
procedure Secant(f, a, b, nmax, ε)
integer n, nmax;   real a, b, fa, fb, ε, d
external function f
fa ← f(a)
fb ← f(b)
  if |fa| > |fb| then
        a ⟷ b
        fa ⟷ fb
  end if
  output 0, a, fa
  output 1, b, fb
  for n = 2 to nmax do
        if |fa| > |fb| then
              a ⟷ b
              fa ⟷ fb
        end if
        d ← (b − a)/(fb − fa)
        b ← a
        fb ← fa
        d ← d · fa
        if |d| < ε then
              output "convergence"
              return
        end if
        a ← a − d
        fa ← f(a)
        output n, a, fa
  end for
  end procedure Secant
```

Example of the Algorithm

$p(x) = x^5 + x^3 + 3$ with $x_0 = -1$ and $x_1 = 1$, what is x_8?

n	x_n	$p(x_n)$
0	-1.0	1.0
1	1.0	5.0
2	-1.5	-7.97
3	-1.05575	0.512
4	-1.11416	-9.991×10^{-2}
5	-1.10462	7.593×10^{-3}
6	-1.10529	1.011×10^{-4}
7	-1.10530	2.990×10^{-7}
8	-1.10530	2.990×10^{-7}