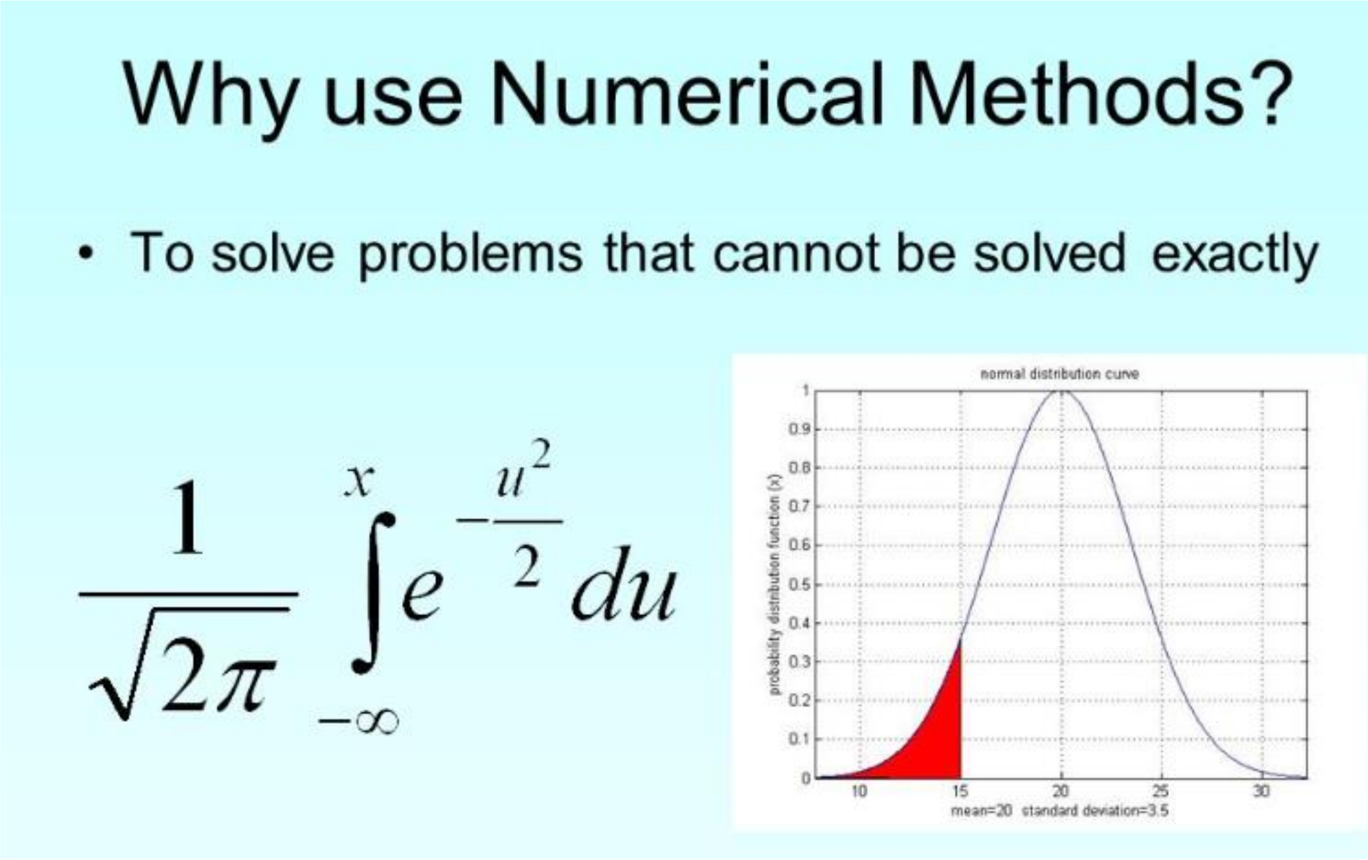
Solution of Non Linear Equation

Yuba Raj Devkota

UNIT 1 [8 hrs]

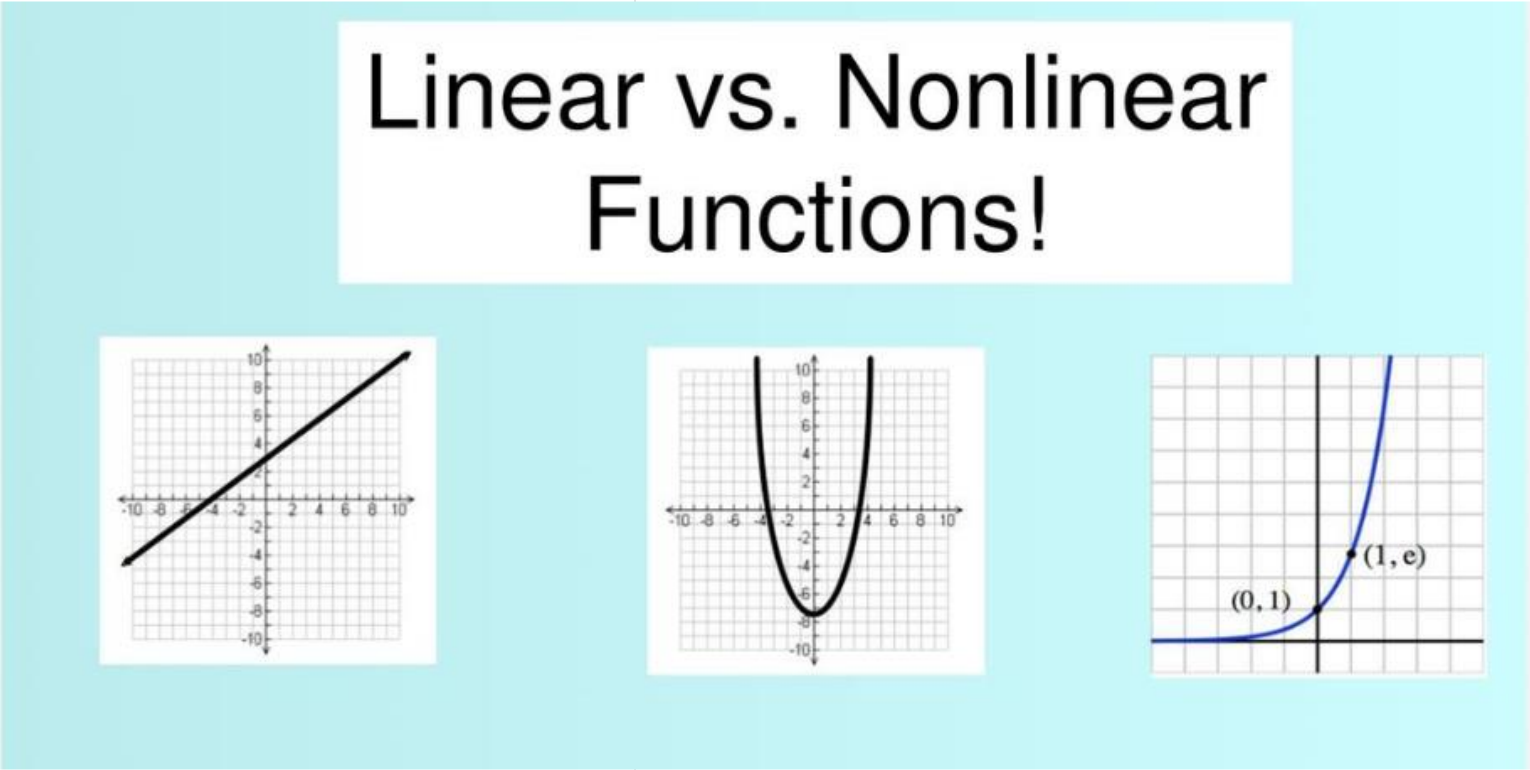
What is Numerical Methods

- A numerical method is **an approximate computer method for solving a mathematical problem which often has no analytical solution**.
- Many engineering problems are too time consuming to solve or may not be able to be solved analytically.
- In these situations, numerical methods are usually employed. Numerical methods are techniques designed to solve a problem using numerical approximations.
- An example of an application of numerical methods is trying to determine the velocity of a falling object. If you know the exact function that determines the position of your object, then you could potentially differentiate the function to obtain an expression for the velocity.



Linear vs Non Linear Equations

Linear means something related to a line. All the linear equations are used to construct a line. A non-linear equation is such which does not form a straight line. It looks like a curve in a graph and has a variable slope value.



Linear Equations	Non-Linear Equations
It forms a straight line or represents the equation for the straight line	It does not form a straight line but forms a curve.
It **has only one degree**. Or we can also define it as an equation having the maximum degree 1.	A nonlinear equation **has the degree as 2 or more than 2**, but not less than 2.
All these equations form a straight line in XY plane. These lines can be extended to any direction but in a straight form.	It forms a curve and if we increase the value of the degree, the curvature of the graph increases.
The general representation of linear equation is; $\mathbf{y = mx + c}$ Where x and y are the variables, m is the slope of the line and c is a constant value.	The general representation of nonlinear equations is; $\mathbf{ax^2 + by^2 = c}$ Where x and y are the variables and a,b and c are the constant values
Examples: • $10x = 1$ • $9y + x + 2 = 0$ • $4y = 3x$ • $99x + 12 = 23y$	**Examples:** • $x^2+y^2 = 1$ • $x^2 + 12xy + y^2 = 0$ • $x^2+x+2 = 25$

Errors in Numerical Calculations

1. **True Error:** True error is denoted by E_t and is defined as the difference between the true value and approximate value i.e. *True error = True value – Approximate value*

2. **Relative Error:** Relative error is denoted by E_r, and is defined as the ratio between the true error and the true value i.e. $Relative\ error = \frac{True\ Error}{True\ Value}$

3. **Approximate Error:** Approximate error is denoted by E_a and is defined as the difference between the present approximation and previous approximation i.e.

   Approximate error= Present approximation- previous approximation

4. **Relative Approximate Error:** Relative approximate error is denoted by E_{ra}, and is defined as the ratio between the approximate error and the present approximation i.e.

   $$Relative\ approximate\ error = \frac{Approximate\ Error}{Present\ Approximation}$$

Sources of Errors

1. **Turncation Errors:** Turncation errors arises from using an approximation in place of exact mathematical procedure. It is the error resulting from the truncation of the numerical process. We often use some finite number of terms to estimate the sum of a finite series. For e.g. $S = \sum_{i=0}^{\infty} a_i x^i$ is replaced by some finite sum which give rise to truncation errors.

2. **Round off Errors:** Round off errors occurs when fixed number of digits are used to represent exact number. Since, the numbers are stored at every stage of computation; round off errors is introduced at the end of every arithmetic operation.

Inherent and Numerical Errors

- **Inherent Errors**: "Inherent" errors typically refer to mistakes or issues that are inherent in a system, process, or concept. These errors are not a result of external factors or mistakes made by individuals but are inherent to the nature of the system or process. They are often challenging to eliminate entirely.
  - For example, in some engineering designs, there may be inherent errors due to physical limitations or constraints. In software, inherent errors can be related to the fundamental design of a system.

- **Numerical Errors**: Numerical errors are inaccuracies that occur when working with numerical data in computer programs, as I explained in my previous response. These errors can result from limited precision in numerical data types, rounding issues, or errors in mathematical algorithms.
  - Numerical errors can impact the results of calculations and may accumulate over time, leading to incorrect outcomes in scientific, engineering, or financial applications. It's essential to understand and manage numerical errors when working with numerical data.

Bracketing & Non Bracketing Methods

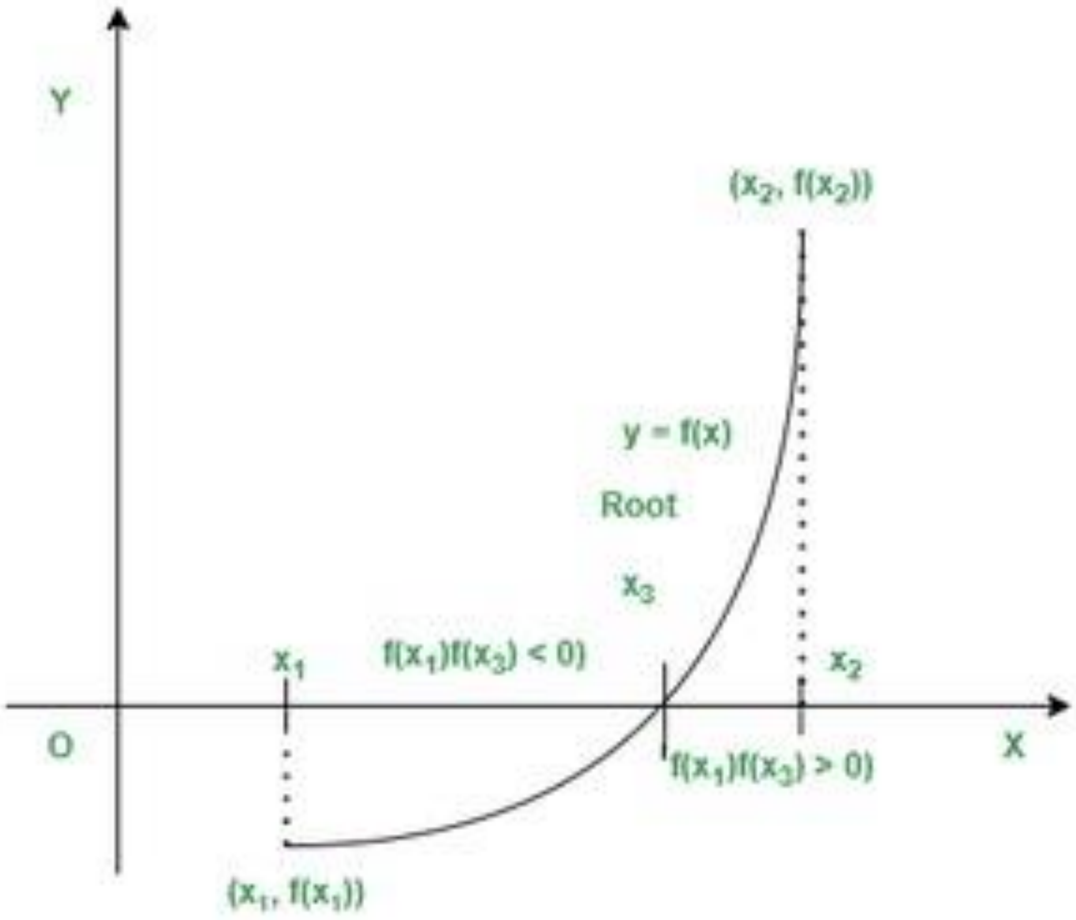
Bracketing & Non-Bracketing Methods

- ***Bracketing Method*** start with two initial guesses that bracket the root and then systematically reduce the width of the bracket until the solution is reached. E.g.
  - Bisection Method.

- ***Non-Bracketing Methods*** (Open-end Method) use a single starting value or two values that do not necessarily bracket the root. E.g.
  - Secant method,
  - Newton-Raphson Method.

Bisection Methods

- **Bisection Method** is one of the simplest, reliable, easy to implement and convergence guaranteed method for finding real root of non-linear equations. It is also known as **Binary Search** or **Half Interval** or **Bolzano Method**.
- Bisection method is **bracketing method** and starts with two initial guesses say **x0** and **x1** such that **x0** and **x1** brackets the root i.e. **f(x0)f(x1)< 0**
- Bisection method is based on the fact that if **f(x)** is real and continuous function, and for two initial guesses **x0** and **x1** brackets the root such that: **f(x0)f(x1) < 0** then there exists at least one root between **x0** and **x1**.
- Root is obtained in Bisection method by successive halving the interval i.e.
  If x0 and x1 are two guesses then we compute new approximated root as:




```
x2 = (x0 + x1)/2
```

Algorithm

Now we have following three different cases:

1.If **f(x2)=0** then the root is **x2**.

2.If **f(x0)f(x2)< 0** then root lies between **x0** and **x2**.

3.If **f(x0)f(x2)> 0** then root lies between **x1** and **x2**.

And then process is repeated until we find the root within desired accuracy.

```
1. start

2. Define function f(x)

3. Choose initial guesses x0 and x1 such that f(x0)f(x1) < 0

4. Choose pre-specified tolerable error e.

5. Calculate new approximated root as x2 = (x0 + x1)/2

6. Calculate f(x0)f(x2)
      a. if f(x0)f(x2) < 0 then x0 = x0 and x1 = x2
      b. if f(x0)f(x2) > 0 then x0 = x2 and x1 = x1
      c. if f(x0)f(x2) = 0 then goto (8)

7. if |f(x2)| > e then goto (5) otherwise goto (8)

8. Display x2 as root.

9. Stop
```

Examples

1. Find the root of equation $x^3 - 2x - 5 = 0$ using bisection method.

Solⁿ:

Given that,

$f(x) = x^3 - 2x - 5$

Let the initial guess be 2 and 3.

$f(2) = -1 < 0$ and $f(3) = 16 > 0$. Therefore root lies between 2 and 3.

Now let us calculate root by tabulation method.

n	x_1	x_2	$x_0 = \frac{x_1 + x_2}{2}$	$f(x_0)$
1	2	3	2.5	5.625
2	2	2.5	2.25	1.891
3	2	2.25	2.215	0.346
4	2	2.125	2.0625	-0.35132
5	2.0625	2.125	2.0938	-0.0083
6	2.0938	2.125	2.1094	0.1671
7	2.0938	2.1094	2.1016	0.0789
8	2.0938	2.1016	2.0977	0.0352
9	2.0938	2.0977	2.0958	0.0139
10	2.0938	2.0958	2.0948	0.0027

Since x1, x2 and x0 are same up to two decimal places, so the root of given equation is 2.0948

Note: To find the root of trigonometric equation, we have to first make sure that calculator is in radian form.

2. ***Estimate a real root of following nonlinear equation using bisection method correct upto two significant figures.***

$$x^2 sinx + e^{-x} = 3$$

Sol:

Given that,

$f(x) = x^2 sinx + e^{-x} - 3 = 0$

Let the initial guess be 1 & 2.

$f(1) = -1.7906 < 0$

$f(2) = 0.7725 > 0$. Therefore root lies between 1 & 2.

Now let us calculate root by tabulation method.

n	x_1	x_2	$x_0 = \frac{x_1 + x_2}{2}$	$f(x_0)$
1	1	2	1.5	-0.5325
2	1.5	2	1.75	0.1872
3	1.5	1.75	1.625	-0.1663
4	1.625	1.75	1.6875	0.0133
5	1.625	1.6875	1.6562	-0.0761
6	1.6562	1.6875	1.6718	-0.0314
7	1.6718	1.6875	1.6796	$-9.17x10^{-3}$
8	1.6796	1.6875	1.6835	$1.91x10^{-3}$
9	1.6796	1.6835	1.6815	$-3.77x10^{-3}$
10	1.6815	1.6835	1.6825	$-9.27x10^{-4}$
11	1.6825	1.6835	1.683	$4.93x10^{-4}$

Since x1, x2 and x0 have same value up to 2 decimal place, so the root of given equation is 1.683

Note: To find the root of trigonometric equation, we have to first make sure that calculator is in radian form.

Q:- Find the zero of $f(x) = x^2 - 2 = 0$; by using bisection method upto four decimal places, in [1, 2].

Sol:-

$f(x) = x^2 - 2$

$f(1) = 1 - 2 = -1 < 0$

$f(2) = (2)^2 - 2 = 2 > 0$

a	b	$c = \frac{a+b}{2}$	f(c)
1	2	1.5000	0.25>0
1	1.5000	1.2500	-0.43<0
1.2500	1.5000	1.3750	-0.1
1.3750	1.5000	1.4375	0.06
1.3750	1.4375	1.4063	-0.02
1.4063	1.4375	1.4219	0.02
1.4063	1.4219	1.4141	-0.0003
1.4141	1.4219	1.4180	0.01
1.4141	1.4180	1.4161	0.005
1.4141	1.4161	1.4151	0.002
1.4141	1.4151	1.4146	0.001
1.4141	1.4146	1.4144	0.0004
1.4141	1.4144	1.4143	0.0001
1.4141	1.4143	1.4142	0.0000

Root of this equation is = 1.4142

- Find the root of $2x - \log_{10} x - 7 = 0$ upto 4 decimal places

No. of Iteration	X_u(A)	X_l(B)	$X=\frac{X_u+X_l}{2}$	F(x)
1.	4	3	$=\frac{4+3}{2}=3.5$	-0.54407
2.	4	3.5	3.75	-0.07403
3.	4	3.75	3.875	0.16173
4.	3.875	3.75	3.8125	0.04379
5.	3.8125	3.75	3.78125	-0.01524
6.	3.8125	3.78125	3.79687	0.01432
7.	3.79687	3.78125	3.78906	-0.00041
8.	3.79627	3.78906	3.79296	0.00694
9.	3.79296	3.78906	3.79101	0.00327
10.	3.79101	3.78906	3.79003	0.00142
11.	3.79003	3.78906	3.78954	0.00049
12.	3.78954	3.78906	3.7893	0.0004
13.	3.7893	3.78906	3.78918	-0.00019
14.	3.7893	3.78918	3.78924	-0.00007
15.	3.7893	3.78924	3.78927	-0.0002

Advantages / Disadvantages of B. M.

1. **Advantages:**
   1. **Convergence is guaranteed:** Bisection method is bracketing method and it is always convergent.
   2. **Error can be controlled:** In Bisection method, increasing number of iteration always yields more accurate root.
   3. **Does not involve complex calculations:** Bisection method does not require any complex calculations. To perform Bisection method, all we need is to calculate average of two numbers.
   4. **Guaranteed error bound:** In this method, there is a guaranteed error bound, and it decreases with each successive iteration. The error bound decreases by ½ with each iteration.
   5. Bisection method is very simple and easy to program in computer.
   6. Bisection method is fast in case of multiple roots.

2. **Disadvantages**
   1. **Slow Rate of Convergence:** Although convergence of Bisection method is guaranteed, it is generally slow.
   2. It can not be applied over an interval where the function takes values of the same sign.
   3. It fails to determine complex roots.

Bisection Method - LAB

```
/* Header Files */
#include<stdio.h>
#include<conio.h>
#include<math.h>
/*
 Defining equation to be solved.
 Change this equation to solve another problem.
*/
#define f(x) cos(x) - x * exp(x)

void main()
{
		float x0, x1, x2, f0, f1, f2, e;
		int step = 1;
		clrscr();
		/* Inputs */
		up:
		printf("\nEnter two initial guesses:\n");
		scanf("%f%f", &x0, &x1);
		printf("Enter tolerable error:\n");
		scanf("%f", &e);
		/* Calculating Functional Value */
		f0 = f(x0);
		f1 = f(x1);
```

```
/* Checking whether given guesses brackets the root or not. */
		if( f0 * f1 > 0.0)
		{
				printf("Incorrect Initial Guesses.\n");
				goto up;
		}


/* Implementing Bisection Method */
		printf("\nStep\t\tx0\t\tx1\t\tx2\t\tf(x2)\n");
		do
		{
				x2 = (x0 + x1)/2;
				f2 = f(x2);

				printf("%d\t\t%f\t%f\t%f\t%f\n",step, x0, x1, x2, f2);
```

```
    if( f0 * f2 < 0)
        {
            x1 = x2;
            f1 = f2;
        }
        else
        {
            x0 = x2;
            f0 = f2;
        }
        step = step + 1;
    }while(fabs(f2)>e);
    printf("\nRoot is: %f", x2);
    getch();
}
```

Bisection Method C Program Output

```
Enter two initial guesses:
0
1
Enter tolerable error:
0.0001

Step        x0              x1              x2              f(x2)
1           0.000000        1.000000        0.500000        0.053222
2           0.500000        1.000000        0.750000        -0.856061
3           0.500000        0.750000        0.625000        -0.356691
4           0.500000        0.625000        0.562500        -0.141294
5           0.500000        0.562500        0.531250        -0.041512
6           0.500000        0.531250        0.515625        0.006475
7           0.515625        0.531250        0.523438        -0.017362
8           0.515625        0.523438        0.519531        -0.005404
9           0.515625        0.519531        0.517578        0.000545
10          0.517578        0.519531        0.518555        -0.002427
11          0.517578        0.518555        0.518066        -0.000940
12          0.517578        0.518066        0.517822        -0.000197
13          0.517578        0.517822        0.517700        0.000174
14          0.517700        0.517822        0.517761        -0.000012

Root is: 0.517761
```

The Method of False Method (Regula Falsi Method)

Regula Falsi (also known as False Position Method) is one of bracketing and convergence guaranteed method for finding real root of non-linear equations.

False Position Method is bracketing method which means it starts with two initial guesses say **x0** and **x1** such that **x0** and **x1** brackets the root i.e. **f(x0)f(x1)< 0**

Regula Falsi is based on the fact that if **f(x)** is real and continuous function, and for two initial guesses **x0** and **x1** brackets the root such that: **f(x0)f(x1) < 0** then there exists atleast one root between **x0** and **x1** .

If `x0` and `x1` are two guesses then we compute new approximated root as:

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

Algorithm for False Position Method

```
1. start

2. Define function f(x)

3. Choose initial guesses x0 and x1 such that f(x0)f(x1) < 0

4. Choose pre-specified tolerable error e.

5. Calculate new approximated root as:

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

6. Calculate f(x0)f(x2)
       a. if f(x0)f(x2) < 0 then x0 = x0 and x1 = x2
       b. if f(x0)f(x2) > 0 then x0 = x2 and x1 = x1
       c. if f(x0)f(x2) = 0 then goto (8)

7. if |f(x2)|>e then goto (5) otherwise goto (8)

8. Display x2 as root.

9. Stop
```

1. If `f(x2)=0` then the root is `x2`.
2. If `f(x0)f(x2)< 0` then root lies between `x0` and `x2`.
3. If `f(x0)f(x2)> 0` then root lies between `x1` and `x2`.

Find a root of an equation $f(x) = x^3 - x - 1$ using False Position method

Approximate root of the equation $x^3 - x - 1 = 0$ using False Position method is 1.32464

n	x_0	$f(x_0)$	x_1	$f(x_1)$	x_2	$f(x_2)$	Update
1	1	-1	2	5	1.16667	-0.5787	$x_0 = x_2$
2	1.16667	-0.5787	2	5	1.25311	-0.28536	$x_0 = x_2$
3	1.25311	-0.28536	2	5	1.29344	-0.12954	$x_0 = x_2$
4	1.29344	-0.12954	2	5	1.31128	-0.05659	$x_0 = x_2$
5	1.31128	-0.05659	2	5	1.31899	-0.0243	$x_0 = x_2$
6	1.31899	-0.0243	2	5	1.32228	-0.01036	$x_0 = x_2$
7	1.32228	-0.01036	2	5	1.32368	-0.0044	$x_0 = x_2$
8	1.32368	-0.0044	2	5	1.32428	-0.00187	$x_0 = x_2$
9	1.32428	-0.00187	2	5	1.32453	-0.00079	$x_0 = x_2$
10	1.32453	-0.00079	2	5	1.32464	-0.00034	$x_0 = x_2$

Example: **Apply Regula Falsi Method to solve the equation**– 3x – cosx – 1 = 0.

Solution:
let f(x) = 3x – cosx – 1 = 0

by hit and trial

f(0.60) = 0.010351 > 0

So the root of the equation lies between x_1=0.60 and x_2 = 0.61, also

f(0.60) f(0.61) < 0

By Regula Falsi Mehtod, the first approximation,

$$x_3 = 0.60 - \frac{0.61 - 0.60}{0.010351 + 0.02533}(-0.02533) = 0.60709$$

now

$$f(x_3) = 3(0.60709) - \cos(0.60709) - 1 = -0.0000415 < 0$$

So, root of the equation f(x) = 0 lies between 0.60709 and 0.61 and also f(0.60709) f(0.61) < 0.

Second approximation

$$x_4 = 0.60709 - \frac{0.61 - 0.60709}{0.010351 + 0.0000415}(-0.0000415) = 0.60710$$

now

$$f(x_4) = 3(0.60710) - \cos(0.60710) - 1 = -0.00000588 < 0$$

So, root of the equation f(x) = 0 lies between 0.60710 and 0.61 and also f(0.60710) f(0.61) < 0.

The third approximation,

$$x_5 = 0.60 - \frac{0.61 - 0.60710}{0.010351 + 0.00000588}(-0.00000588) = 0.607101$$

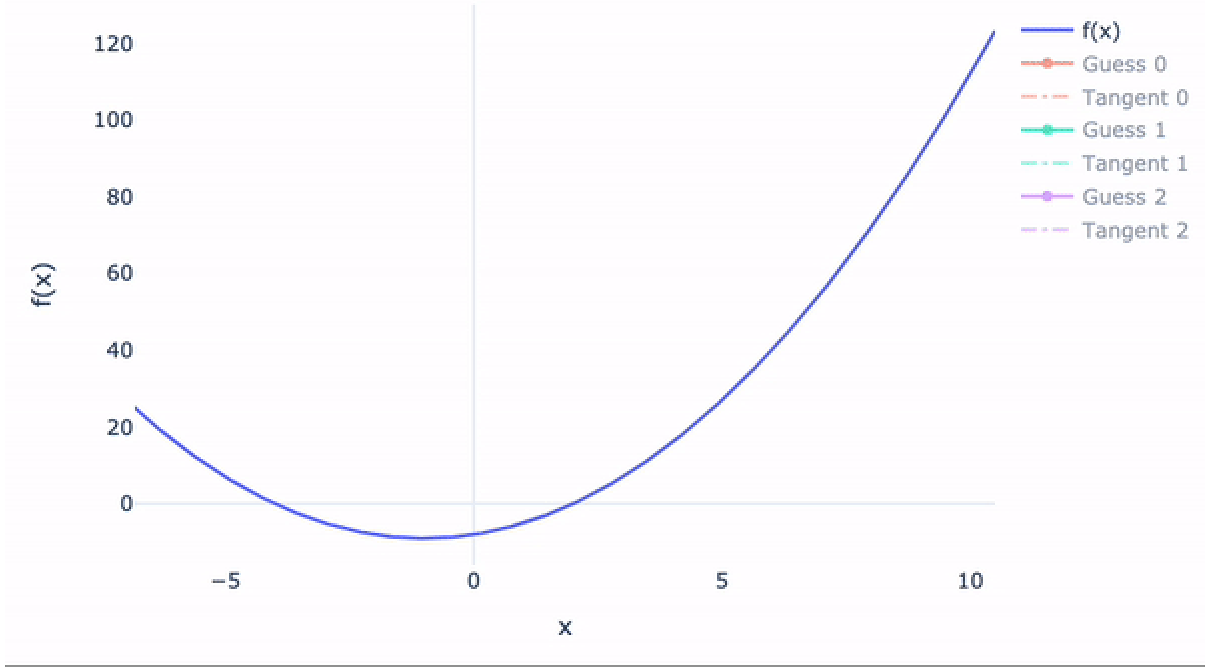
Hence the root of the given equation correct to five decimal place is 0.60710.

Newton-Raphson Method

Newton Raphson Method is an open method and starts with one initial guess for finding real root of non-linear equations.

In Newton Raphson method if **x0** is initial guess then next approximated root **x1** is obtained by following formula:

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



And **an algorithm for Newton Raphson method** involves repetition of above process i.e. we use **x1** to find **x2** and so on until we find the root within desired accuracy.

Examples

1. Find the root of the equation $e^x - 3x = 0$ using Newton Raphson method correct up to 3 decimal places.

Sol^n:

Given that,
$f(x) = e^x - 3x$
$f'(x) = e^x - 3$
Let the initial guess be 0.5.

Now, let us calculate the root using tabular form.

n	x_1	$f(x_1)$	$f'(x_1)$	$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$
1	0.5	0.14872	-1.351278	0.61005
2	0.61005	0.010373	-1.15947	0.61899
3	0.61899	0.0081472	-1.14294	0.61273
4	0.61273	0.0035755	-1.14287	0.61903
5	0.61903	0.00003575	-1.14287	0.61906

Here, the 4th and 5th iteration has same value of x_2 up to 3 decimal place, so that root of given equation is 0.61906.

Algorithm

1. *Guess initial root=x_1 and define stopping criteria error.*
2. *Evaluate $f(x_1)$ & $f'(x_1)$*
3. *Compute new root*

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

4. *Set $x_1=x_2$*
5. *Check if $|f(x_2)| > error$*

   goto step 3,otherwise;
6. *Print root = x_2*
7. *END*

2. ***Use the Newton method to estimate the root of the equation*** $\mathbf{x^2 + 2x - 2 = 0}$.

Sol^n^:

Given that,
$f(x) = x^2 + 2x - 2$

$f'(x) = 2x + 2$

Let the initial guess be 0.

Now, let us calculate the root using tabular form.

n	x_1	$f(x_1)$	$f'(x_1)$	$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$
1	0	-2	2	1
2	1	1	4	0.75
3	0.75	0.0625	3.5	0.73214
4	0.73214	0.000309	3.464	0.73205
5	0.73205	$-2.7975x10^{-6}$	3.464	0.73205

Here, the 4^{th} and 5^{th} iteration has same value of x_2, so that root of given equation is 0.73205.

3. Find the root of equation $xcosx - x^2 = 0$ using newton's method up to 5 decimal places.

Sol^n:

Given that,

$f(x) = xcosx - x^2$

$f'(x) = cosx - xsinx - 2x$

Let the initial guess be 1.

Now, let us calculate the root using tabular form.

n	x_1	$f(x_1)$	$f'(x_1)$	$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$
1	1	-0.4597	-2.3012	0.80023
2	0.80023	-0.0829	-1.4781	0.74414
3	0.74414	$-6.302x10^{-3}$	-1.2566	0.73912
4	0.73912	$-4.313x10^{-5}$	-1.2371	0.73908
5	0.73908	$6.349x10^{-6}$	-1.2369	0.73908

Here, the 4th and 5th iteration has same value of x_2 up to 5 decimal places, so that root of given equation is 0.73908.

4. Find the roots of the following equations using Newton's method.

log x – cos x = 0

Solution:

Given,

$f(x) = logx - cosx$

$f'(x) = \frac{1}{x} + sinx$

Let the initial guess be 0.5.

n	x_1	$f(x_1)$	$f'(x_1)$	$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$
1	0.5	-1.17	2.47	1.62
2	1.62	0.26	1.62	1.45
3	1.45	0.04	1.68	1.43
4	1.43	0.01	1.68	1.42
5	1.42	0.002	1.69	1.42

Here, the 4^{th} and 5^{th} iteration has same value of x_2 up to 2 decimal place, so that root of given equation is 1.42.

```
/* Program: Finding real roots of nonlinear
   equation using Newton Raphson Method */
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<stdlib.h>

/* Defining equation to be solved.
   Change this equation to solve another problem. */
#define    f(x)    3*x - cos(x) -1

/* Defining derivative of g(x).
   As you change f(x), change this function also. */
#define   g(x)   3 + sin(x)

void main()
{
        float x0, x1, f0, f1, g0, e;
        int step = 1, N;
        clrscr();
```

```
/* Inputs */
        printf("\nEnter initial guess:\n");
        scanf("%f", &x0);
        printf("Enter tolerable error:\n");
        scanf("%f", &e);
        printf("Enter maximum iteration:\n");
        scanf("%d", &N);
        /* Implementing Newton Raphson Method */
        printf("\nStep\t\tx0\t\tf(x0)\t\tx1\t\tf(x1)\n");
        do
        {
                g0 = g(x0);
                f0 = f(x0);
                if(g0 == 0.0)
                {
                        printf("Mathematical
Error.");
                        exit(0);
                }
```

```
                x1 = x0 - f0/g0;

printf("%d\t\t%f\t%f\t%f\t%f\n",step,x0,f0,x1,f1);
                x0 = x1;

                step = step+1;

                if(step > N)
                {
                        printf("Not Convergent.");
                        exit(0);
                }

                f1 = f(x1);

        }while(fabs(f1)>e);

        printf("\nRoot is: %f", x1);
        getch();
}
```

Output: Newton Raphson Method Using C

```
Enter initial guess:
1
Enter tolerable error:
0.00001
Enter maximum iteration:
10

Step            x0              f(x0)           x1              f(x1)
1               1.000000        1.459698        0.620016        0.000000
2               0.620016        0.046179        0.607121        0.046179
3               0.607121        0.000068        0.607102        0.000068

Root is: 0.607102
```

Fixed Point Method

- Fixed point iteration method is open and simple method for finding real root of non-linear equation by successive approximation. It requires only one initial guess to start. Since it is open method its convergence is not guaranteed. This method is also known as **Iterative Method**
- To find the root of nonlinear equation **f(x)=0** by fixed point iteration method, we write given equation **f(x)=0** in the form of **x = g(x)**.

If `x0` is initial guess then next approximated root in this method is obtaine by:

```
x1 = g(x1)
```

And similarly, next to next approximated root is obtained by using value of `x1` i.e.

```
x2 = g(x2)
```

And the process is repeated until we get root within desired accuracy.

Algorithm

1. *Define function f(x) and error*
2. *Convert the function f(x) = 0 in the form x = g(x).*
3. *Guess initial values x_0.*
4. *Calculate* $x_{i+1} = g(x_i)$
5. *If* $\left|\frac{x_{i+1} - x_i}{x_{i+1}}\right| <= error$

   goto step 7, otherwise;
6. *Assign* $x_i = x_{i+1}$

   goto step 4
7. *Display x_i as the root*
8. *END*

1. ***Find the one root of the equation $x^2 + x - 2 = 0$ using the fixed point method.***

Sol:

Given that,

$f(x) = x^2 + x - 2 = 0$.............. (1)

For fixed point iteration method, arranging equation (1) in terms of g(x)

$$x^2 + x - 2 = 0$$
$$\text{or, } x(x+1) = 2$$
$$\text{or, } x = \frac{2}{x+1}$$
$$\therefore g(x) = \frac{2}{x+1}$$

Let the initial guess be 0.

Now calculating the root using tabular form

n	x_i	$x_{i+1} = g(x_i)$
1	0	2
2	2	0.667
3	0.667	1.199
4	1.199	0.91
5	0.91	1.04
6	1.04	0.98
7	0.98	1.01
8	1.01	0.99
9	0.99	1.00
10	1.00	1.00

Since the value of g(x) in 9th and 10th iteration has similar value. So the root of given equation is 1.00

2. ***Find the root of equation $\sin x = 5x - 2$ up to 4 decimal places with initial guess 0.5 using fixed point iteration method.***

Sol^n:

Given that,

$f(x) = \sin x - 5x + 2 = 0$ ……………(1)

For fixed point iteration method, arranging equation (1) in terms of g(x)

$$\sin x - 5x + 2 = 0$$

$$\text{or, } \sin x = 5x - 2$$

$$\text{or, } x = \frac{1}{5}(\sin x + 2)$$

$$\therefore g(x) = \frac{1}{5}(\sin x + 2)$$

Now calculating the root using tabular form

n	x_i	$x_{i+1} = g(x_i)$
1	0.5	0.49589
2	0.49589	0.49516
3	0.49516	0.49503
4	0.49503	0.49501

Since the value of g(x) in 3rd and 4th iteration has similar value upto 4 decimal places. So the root of given equation is 0.49501

```
/* Program: Finding real roots of nonlinear
   equation using Fixed Point Iteration */
/* Header Files */
#include<stdio.h>
#include<conio.h>
#include<math.h>

/* Define function f(x) which
   is to be solved */
#define   f(x)   cos(x)-3*x+1
/* Write f(x) as x = g(x) and
   define g(x) here */
#define   g(x)   (1+cos(x))/3

int main()
{
        int step=1, N;
        float x0, x1, e;
        clrscr();
/* Inputs */
printf("Enter initial guess: ");
scanf("%f", &x0);
printf("Enter tolerable error: ");
scanf("%f", &e);
printf("Enter maximum iteration: ");
scanf("%d", &N);
/* Implementing Fixed Point Iteration */
printf("\nStep\tx0\t\tf(x0)\t\tx1\t\tf(x1)\n");
do
{
        x1 = g(x0);
        printf("%d\t%f\t%f\t%f\t%f\n",step, x0, f(x0), x1, f(x1));

        step = step + 1;
```

```
        if(step>N)
        {
                printf("Not Convergent.");
                exit(0);
        }

        x0 = x1;

    }while( fabs(f(x1)) > e);

    printf("\nRoot is %f", x1);

    getch();
    return(0);
}
```

Output: Fixed Point Iteration Using C

```
Enter initial guess: 1
Enter tolerable error: 0.000001
Enter maximum iteration: 10

Step      x0              f(x0)          x1            f(x1)
1          1.000000       -1.459698      0.513434      0.330761
2          0.513434       0.330761        0.623688      -0.059333
3          0.623688       -0.059333      0.603910      0.011391
4          0.603910       0.011391        0.607707      -0.002162
5          0.607707       -0.002162      0.606986      0.000411
6          0.606986       0.000411        0.607124      -0.000078
7          0.607124       -0.000078      0.607098      0.000015
8          0.607098       0.000015        0.607102      -0.000003
9          0.607102       -0.000003      0.607102      0.000001

Root is 0.607102
```

Convergence of different methods

1. Bisection Method: The Bisection method is a simple and robust method for finding a root of a function. It has linear convergence, which means that the error decreases by a constant factor with each iteration. Specifically, the error decreases by a factor of 1/2 with each iteration
2. Secant Method: The Secant method is a variation of the Newton-Raphson method that uses a finite difference approximation of the derivative. It has superlinear convergence, which means that the error decreases faster than linearly with each iteration. Specifically, the error decreases by a factor of $(1+\sqrt{5})/2 \approx 1.618$ with each iteration, which is the golden ratio.
3. Newton-Raphson Method: The Newton-Raphson method is a popular method for finding roots of a function. It has quadratic convergence, which means that the error decreases by a squared factor with each iteration.
4. Fixed Point Iteration Method: The Fixed Point iteration method is a simple method for solving equations of the form $x = g(x)$, where g is a continuous function. It has linear convergence if the absolute value of the derivative of g is less than 1 at the fixed point, which is equivalent to saying that the fixed point is stable. If the absolute value of the derivative of g is greater than 1 at the fixed point, then the method does not converge.
5. Horner's Method: Horner's method is an algorithm for evaluating polynomials that reduces the number of arithmetic operations required. It does not have convergence properties in the same sense as the previous methods because it is not an iterative method. Instead, it is a finite algorithm that computes the value of the polynomial at a given point.

Lab Questions

1. Find the real roots of non linear equation using Bisection Method
2. Find the root of the equation e^x-3x=0 using Newton Raphson method correct up to 3 decimal places.
3. Find the one root of the equation x^2+x-2=0 using the fixed point method

Old TU Questions / Homework - 1

2. why is the study of errors important to a computational scientist ? Differentiate between inherent and numerical errors.

1. What is error? Discuss various types of errors. Estimate a real root of the following nonlinear equation using bisection method or Newton's method correct up to two decimal places $\sin x - x^2 - x + 3 = 0$ (3 + 5) asked in 2075(Old Course)

1. How can you use bisection method for the solution of nonlinear equations? Discuss with suitable example.(8) asked in 2074

2. Describe Newton's method and its convergence. Find the root of equation $f(x) = e^x - 4x^2 = 0$ using Newton method up to 5 decimal places. (4+4) asked in 2072

3. Find the root of equation $x^2 - 4x - 10 = 0$ using bisection method where root lies between 5 and 6.

5. Calculate a real root of the given equation using fixed point iteration correct up to 3 significant figures. asked in 2078

$2x^3 - 2x = 5$

(b) Locate the root of equation $x^2 + x - 2 = 0$ using the fixed point method?