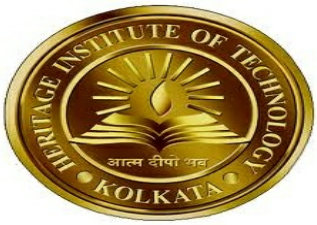
LAB MANUAL



Numerical and Statistical Methods Lab

SUB CODE:- MATH2012/ M(CS)391/ M(CS)491

Department of Mathematics

HERITAGE INSTITUTE OF TECHNOLOGY

List of Experiments(MATH2012)

Serial No:	Name of the Experiments
1.	Numerical Method to implement the ***Newton's Forward Interpolation*** formula.
2.	Numerical method to implement the ***Lagrange's Interpolation*** formula.
3.	Numerical solution for a definite integral using ***Trapezoidal Rule and Simpson's*** $\frac{1}{3}$ ***Rule***.
4.	To find the roots of non linear transcendental equation using ***Regula-Falsi method*** and ***Newton-Raphson*** method.
5.	Numerical method to solve a linear system of algebraic equations by ***Gaussian Elimination*** method
6.	Numerical method to solve a linear system of algebraic equation by ***Gauss – Seidel's*** method.
7.	Numerical method to approximate the solution of a ordinary differential equation by ***Euler's , Modified Euler's and Runge - Kutta*** method of order 4.
8.	Calculate the ***Mean , Median , Mode , Standard deviation*** for the grouped and ungrouped data.
9.	Detemine the ***Correlation Co-efficient*** between two random variables and the ***Regression equation y on x***.

Why do we study Numerical and statistical methods?

There was a popular joke in numerical analysis invented by a very old professor. Whenever he would write on the blackboard, his hands would shake and trying to write =, it would always come out ≈.

There are many things that you want to compute that can not be computed exactly. The roots of high degree polynomials. The solution of simple differential equations on irregular domains. Any calculation with real numbers that you do with a computer .

Numerical analysis is the branch of mathematics that is about approximate computing. It tells you how quickly you can get how close to the true solution. If you study any sort of engineering science you need inevitably learn some corner of numerical analysis.

A numerical method is the same as an algorithm, the steps required to solve a numerical problem. Algorithms became very important as computers were increasingly used to solve problems.

1. Write a program in 'C' to implement the ***Newton's Forward Interpolation*** formula.

(x)	100	150	200	250	300	350	400
(y)	10.66	13.06	15.07	16.84	18.45	19.93	21.30

Hence test your program to find the value of y when x =120 using ***Newton's Forward Interpolation***.

Code:-

```
#include<stdio.h>
#include<conio.h>
void main()
{
float a[10][10],sum,term,u,h,x;
int i,j,n;
clrscr();
printf("\n Enter the number of data points:");
scanf("%d",&n);
printf("\n Enter the point to be interpolated:");
scanf("%f",&x);
printf("\n Enter the data points:\n");
                                for(i=0;i<n;i++)
                                {
                                scanf("%f",&a[i][0]);
                                }
printf("\n Enter the functional values at data points:\n");
 for(i=0;i<n;i++)
                                {
                                scanf("%f",&a[i][1]);
                                }
                                   for(j=2;j<=n;j++)
                                {
                                for(i=0;i<n;i++)
```

```
                                        {
                                        a[i][j]=a[i+1][j-1]-a[i][j-1];
                                        }
                                }
printf("\n The Forward Difference Table is:\n");
 for(i=0;i<n;i++)
                                {
                                for(j=0;j<n+1-i;j++)
                                        {
                                        printf("  %7.5f ",a[i][j]);
                                        }
                                printf("\n");
                                }
term=1;
h=a[1][0]-a[0][0];
u=(x-a[0][0])/h;
sum=a[0][1];
for(j=2;j<=n;j++)
                                {
                                term*=(u-(j-2))/(j-1);
                                sum+=term*a[0][j];
                                }
printf("\n The required interpolated value of f(%4.3f)=%7.6f",x,sum);
getch();
}
```

/***Output:-**

```
 Enter the number of data points:7

 Enter the point to be interpolated:120

 Enter the data points:
100  150  200  250  300  350  400

 Enter the functional values at data points:
10.66  13.06  15.07  16.84  18.45  19.93  21.30

 The Forward Difference Table is:
  100.00000   10.66000   2.40000   -0.39000   0.15000   -0.07000   0.02000   0.02000
  150.00000   13.06000   2.01000   -0.24000   0.08000   -0.05000   0.04000
  200.00000   15.07000   1.77000   -0.16000   0.03000   -0.01000
  250.00000   16.84000   1.61000   -0.13000   0.02000
  300.00000   18.45000   1.48000   -0.11000
  350.00000   19.93000   1.37000
  400.00000   21.30000

 The required interpolated value of f(120.000)=11.679452 */
```

2. Write a program in 'C' to implement the ***Lagrange's Interpolation*** formula and Using ***Lagrange's Interpolation*** formula

find The value of f(102) from the following table .

x :	93.0	96.2	100.0	104.2	108.7
$y = f(x)$:	11.38	12.80	14.70	17.07	19.91

Code:-

```
#include<stdio.h>

#include<conio.h>

void main()
```

```
{   int i,j,k,n;
    float x[10],y[10],a[10],X,Y,p,u;
    clrscr();
    printf("Enter the value of n(No. of data pairs): \n");
    scanf("%d",&n);
    printf("Enter the value of xi and yi : \n");
    for(i=0;i<n;i++)
            scanf("%f%f",&x[i],&y[i]);
            printf("Enter the interpolating point: \n");
            scanf("%f",&X);
            Y=0.0;
            for(i=0;i<n;i++)
            {
               a[i]=1;
               for(j=0;j<n;j++)
                        {
                        if(i!=j)
                        a[i]=a[i]*((X-x[j])/(x[i]-x[j]));
                        }
                         Y=Y+a[i]*y[i];
            }
                        printf("The value y(%f)= %f",X,Y);
                        getch();}
/*Output:-
Enter the value of n(No. of data pairs):
5
Enter the value of xi and yi :
 93.0
11.38
96.2
12.80
100.0
14.70
104.20
17.07
108.7
19.91
Enter the interpolating point:
102
The value f(102)= 15.793631 */
```

3. (a) Write a program in 'C' to approximate a definite integral by ***Trapezoidal Rule***

(b) Write a program in'C' to approximate a definite integral, by ***Simpson's*** $\frac{1}{3}$ ***Rule***.

Test the above programs with the following integral

$\int_1^2 \frac{1}{x} dx$ [Take n=6]

(a) **Code:-**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define f(x) (1/(x))
void main()
{
float a,b,h,sum=0.0;
int i,n;
clrscr();
printf("\n\t enter the lower limit: ");
scanf("%f",&a);
printf("\n\t enter the upper limit: ");
scanf("%f",&b);
printf("\n\t enter the number of subinterval: ");
scanf("%d",&n);
h=(b-a)/n;
sum=f(a)+f(b);
for(i=1;i<n;i++)
{
sum=sum+2*f(a+i*h);
}
sum=h/2*(sum);
printf("\n\t the value of the integral is%f ",sum);
getch();
}
```

/*Output:-

```
        enter the lower limit: 1

        enter the upper limit: 2

        enter the number of subinterval: 6

    the value of the integral is  0.694877
 */
```

(b) **Code:-**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define f(x) (1/(x))
void main()
{
  int n,i;
  float a,b,h,sum=0.0;
  clrscr();
  printf("Enter the lower limit of the integration:  ");
  scanf("%f",&a);
  printf("\nEnter the upper limit of the integration:  ");
  scanf("%f",&b);
  printf("\nEnter the sub-interval of the integration:  ");
  scanf("%d",&n);
  h=(b-a)/n;
  sum=f(a)+f(b);
  for(i=1;i<=n-1;i++)
  {
   if(i%2==0)
```

```
   sum+=2*f((a+(i*h)));
   else
   sum+=4*f((a+(i*h)));
   }
   sum=(h/3)*sum;
   printf("\n Integration value is: %f",sum);
    getch();

}
```
/***Output:-**
Enter the lower limit of the integration: 1
Enter the upper limit of the integration: 2
Enter the sub-interval of the integration: 6
Integration value is: 0.693170*/

4.(a) Write a program in 'C' to find a root of a transcendental equation by ***Regula-Falsi method***
   (b) Write a program in 'C' to find a real root of a transcendental equation using ***Newton-Raphson*** method.
        Find a real positive root of the equation $x^4 - x - 10 = 0$ using above two methods,
        using a tolerance limit of 0.00001.

(a) **Code:-**
```
#include<stdio.h>
 #include<conio.h>
 #include<math.h>
 #define f(x) (pow(x,4)-x-10)
 void main()
 {int i=0;
  float a,b,oldx,newx;
  clrscr();
  a=0;
  b=1;
  while(f(a)*f(b)>0)
  {
  a++;
  b++;
  }
  printf("the positive root lies between %f  and  %f\n",a,b);
  do
  {
  i++;
  oldx=(a*f(b)-b*f(a))/(f(b)-f(a));
  if(f(a)*f(oldx)<0)
  b=oldx;
  else
  a=oldx;
  newx=(a*f(b)-b*f(a))/(f(b)-f(a));
  printf("Root of the given equation is:= %f and number of it=%d\n",newx,i);
  }while(fabs(newx-oldx)>0.001);

  getch();
 }
```
 /***Output:-**
 the positive root lies between 1.000000 and 2.000000
Root of the given equation is:= 1.85584
 */

(b) **Code:-**
```
#include<stdio.h>
  #include<conio.h>
  #include<math.h>
  #define f(x) (pow(x,4)-x-10)
```

```
 #define df(x) (4*pow(x,3)-1))
void main()
{int i;
 float a,b,oldx,newx;
 int c;
 clrscr();
 a=0;
 b=1;
 getch();
 while(f(a)*f(b)>0)
 {
 a++;
 b++;
 }
 printf("the root lies between %f  and  %f\n",a,b);
 newx=b;
 c=0;
 do
 {
 c++;
 oldx=newx;
 newx=oldx-f(oldx)/df(oldx);
 printf("Root of the given equation in iteration no %d is:= %f \n",c,newx);
 }while(fabs(newx-oldx)>0.00001);

 getch();
     }
```

/***Output:-**

```
the root lies between 1.000000  and  2.000000
Root of the given equation in iteration no 1 is:= 1.85584
```

5. Write a program in 'C' to approximate the solution of linear system of equations by ***Gaussian Elimination method.*** Hence use the program to solve the following problem.

$$x - 2y + 9z = 8$$
$$3x + y - z = 3$$
$$2x - 8y + z = -5$$

Code:-

```
#include<stdio.h>
#include<conio.h>
void main()
{
  int n,i,j,k;
  float a[50][50],x[50],d,sum=0.0;
  clrscr();
  printf("\nEnter the no. of unknowns:");
  scanf("%d",&n);
  printf("\nEnter the value of augmented matrix:");
  for(i=1;i<=n;i++)
  {
    for(j=1;j<=(n+1);j++)
    scanf("%f",&a[i][j]);
  }
  printf("\nAugmented matrix:");
  for(i=1;i<=n;i++)
  {
    printf("\n");
    for(j=1;j<=(n+1);j++)
```

```
    printf("%f   ",a[i][j]);
  }
  for(k=1;k<=(n-1);k++)
  {
   for(i=(k+1);i<=n;i++)
   {
     d=a[i][k]/a[k][k];
     for(j=1;j<=(n+1);j++)
     a[i][j]=a[i][j]-(d*a[k][j]);
   }
  }
  printf("\nUpper triangular matrix:");
  for(i=1;i<=n;i++)
  {
   printf("\n");
   for(j=1;j<=(n+1);j++)
   printf("%f   ",a[i][j]);
  }
  x[n]=a[n][n+1]/a[n][n];
  printf("\n%f was unknown no. %d",x[n],n);
  for(i=(n-1);i>=1;i--)
  {
   for(j=(i+1);j<=n;j++)
   {
     sum=sum+(a[i][j]*x[j]);
   }
   x[i]=(a[i][n+1]-sum)/a[i][i];
   sum=0.0;
   printf("\n%f was unknown no. %d",x[i],i);
  }

  getch();
}
```

/*__Output:__-

```
Enter the no. of unknowns:3

Enter the value of augmented matrix:
1     -2  9          8
3     1   -1         3
2     -8  1          -5
Augmented matrix:
1.000    -2.000   9.000    8.000
3.000    1.000    -1.000   3.000
2.000    -8.000   1.000    -5.00
```

Upper triangular form: $\begin{bmatrix} 1 & -2 & 9 & 8 \\ 0 & 7 & -28 & -21 \\ 0 & 0 & -33 & -33 \end{bmatrix}$

```
x[3]=1.00000
x[2]=1.00000
x[1]=1.00000
*/
```

6. Write a program in 'C' to approximate the solution of a linear system of equations by ***Gauss – Seidel's*** method. Hence use the program to solve the following problem.

$$5x - 2y + z = 4$$
$$x + 6y - 2z = -1$$
$$3x + y + 5z = 13$$

Code:-

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
float a[10][10],x[20],sum,temp[10],t;
int i,j,k,n,f;
clrscr();
printf("\n\t enter the order of the matrix:");
scanf("%d",&n);
printf("\n\t enter the tolerance limit:");
scanf("%f",&t);
printf("\n\t enter the element of the matrix\n\t");
for(i=1;i<=n;i++)
{
      for(j=1;j<=n+1;j++)
      scanf("%f",&a[i][j]);
printf("\n\t");
}
for(i=1;i<=n;i++)
x[i]=0;
f=1;
  while(f)
  {
      for(i=1;i<=n;i++)
      {
      sum=0;
      temp[i]=x[i];
           for(j=1;j<=n;j++)
           {
                     if(j!=i)
                     sum+=a[i][j]*x[j];
           }
      x[i]=(a[i][n+1]-sum)/a[i][i];
      }
           f=0;
           for(i=1;i<=n;i++)
             {if(fabs(x[i]-temp[i])>t)
                     f=1;
                     }
}
for(i=1;i<=n;i++)
printf("x[%d]=%f\n",i,x[i]);
getch();
}
```

/***Output**

```
      enter the order of the matrix:3

      enter the tolerance limit:.00001

      enter the element of the matrix
5
-2
1
4
1
6
-2
-1
```

3
1
5
13

x[1]=0.551516
x[2]=0.466666
x[3]=2.175757
*/

7.(a) Write a program in 'C' to approximate the solution of a ordinary differential equation by ***Euler's*** method.
(b) Write a program in 'C' to approximate the solution of a ordinary differential equation by ***Modified Euler's*** method.

(c) Write a program in 'C' to approximate the solution of a ordinary differential equation by ***Runge - Kutta*** method of order 4.
Calculate the value of y at the point x= 0.5, of the following differential equation: [take $h=0.1$]

$\frac{dy}{dx} = x * y, y(0) = 2$ using above two programs.

(a) **Code:-**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define f(x,y) ((x)*(y))


void main()
{
  float h,x,y,i,xn;
  clrscr();
  printf("Enter the initial value of x and y: \n");
  scanf("%f%f",&x,&y);
  printf("Enter the step size");
  scanf("%f",&h);
  printf("Enter the final value of x");
  scanf("%f",&xn);
  printf("\n Euler's-method solution:\n\n");
       for(i=x;fabs(i-xn)>0.001;i=i+h)
            {
                      y=y+h*f(i,y);
                     printf("y(%f)=%f\n",i+h,y);

            }

  getch();
  }
```

/***Output:-**
Enter the initial value of x and y:
0
2
Enter the step size0.1
Enter the final value of x 0.5

Euler's-method solution:
y(0.500000)=2.207100 */

(b) **Code:-**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define f(x,y) ((x)*(y))
```

```
void main()
{ int i,n;
  float x,y,h,y2,y1,xn;
  clrscr();
  printf("Enter the initial value of x and y respectively");
  scanf("%f%f",&x,&y);
  printf("Enter the step size and the final value of x");
  scanf("%f%f",&h,&xn);
  do
  {    y1=y+h*f(x,y);
       do
       {
            y2=y1;
            y1=y+(h/2)*(f(x,y)+f(x+h,y2));
            printf("\n\tmodified y(%f)=%f",x,y1);
       }while(fabs(y2-y1)>.0001);
       x=x+h;
       y=y1;
       printf("\nfinal y(%f)=%f",x,y);
  }while(fabs(x-xn)>0.01);
  getch();
 }
 /*Output:-
Enter the initial value of x and y respectively0
2
Enter the step size and the final value of x0.1
0.5

final y(0.500000)=2.267036*/
```

(c) **Code:**-

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define f(x,y) ((x)*(y))

void main()
{
  float h,k1,k2,k3,k4,x,y,ym,xn,i;
  clrscr();
  printf("Enter the initial value of x and y: \n");
  scanf("%f%f",&x,&y);
  printf("Enter the step size");
  scanf("%f",&h);
  printf("Enter the final value of x");
  scanf("%f",&xn);
  printf("\n Runge-kutta method- solution:\n\n");
       for(i=x;i<xn;i=i+h)
            { k1=h*f(i,y);
              k2=h*f((i+h/2),(y+k1/2));
              k3=h*f((i+h/2),(y+k2/2));
              k4=h*f((i+h),(y+k3));
              y=y+(k1+2*(k2+k3)+k4)/6;
                        printf("y(%f)=%f\n",i+h,y);

            }

  getch();
  }
/*Output:-
Enter the initial value of x and y:
```

0
2
Enter the step size0.01
Enter the final value of x0.5

Runge-kutta method- solution:

y(0.500000)=2.266297*/

8.(a) Find the mean ,median, mode and standard deviation of the following ungrouped data:
31, 23, 34, 56, 34, 12, 78, 12, 34, 45

(b) Find the mean ,median, mode and standard deviation of the following grouped data:

Variate(x):	20-30	30-40	40-50	50-60
Frequency(f):	13	12	14	10

(a) **Code:-**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
float x[15];
float mean,median,sum,sumsd=0,temp;
int i,n,j,max,c[15],p;
clrscr();
printf("enter the no of data");
scanf("%d",&n);
printf("enter the elements");
for(i=1;i<=n;i++)
{
   scanf("%f",&x[i]);
}
for(i=1;i<=n;i++)
{
   sum+=x[i];
}
mean=sum/n;
printf("\nthe required mean is=%f",mean);
for(i=1;i<=n;i++)
{
   for(j=i+1;j<=n;j++)
   {
          if(x[i]>= x[j])
          {
                 temp=x[j];
                 x[j]=x[i];
                 x[i]=temp;
          }
   }
}
if(n%2==0)
{
   median=(x[n/2]+x[n/2+1])/2;
}
else
   median=x[(n+1)/2];
printf("\nthe required median is=%f",median);
for(i=1;i<=n;i++)
{
   c[i]=1;
   if(c[i]!=-999)
```

```
  {
        for(j=i+1;j<=n;j++)
        {
              if(x[i]==x[j])
              {
                    c[i]=c[i]+1;
                    c[j]=-999;
              }
        }
  }
}
max=c[0];
for(i=1;i<=n;i++)
{
  if(c[i]>max)
  {
        max=c[i];
        p=i;
  }
}
printf("\nthe required mode is=%f",x[p]);
for(i=1;i<=n;i++)
{
  sumsd+=pow((x[i]-mean),2);
}
sumsd=sqrt(sumsd/n);
printf("\nthe required standard deviation is=%f",sumsd);
getch();
}
/*Output:-
enter the no of data10
enter the elements31
23
34
56
34
12
78
12
34
45
the required mean is=35.900002
the required median is=34.000000
the required mode is=0.000000
the required standard deviation is=18.981306*/
```

(b) **Code:-**

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
float xl[10],xu[10],xm[10],cf[10],f[10],fx[10],sumf=0,sumfx=0;
float mean,median,mode,sd,max,sumsd=0;
int i,n,p,k;
clrscr();
printf("enter the no of data");
scanf("%d",&n);
printf("enter the lower boundary");
for(i=1;i<=n;i++)
{
  scanf("%f",&xl[i]);
```

```
}
printf("enter the upper boundary");
for(i=1;i<=n;i++)
{
   scanf("%f",&xu[i]);
}
printf("enter the frequency");
for(i=1;i<=n;i++)
{
   scanf("%f",&f[i]);
}
for(i=1;i<=n;i++)
{
   xm[i]=(xl[i]+xu[i])/2;
   fx[i]=xm[i]*f[i];
   sumf+=f[i];
   sumfx+=fx[i];
}
mean=sumfx/sumf;
printf("\nthe required mean is=%f",mean);
cf[1]=f[1];
for(i=2;i<=n;i++)
{
   cf[i]=cf[i-1]+f[i];
}
i=1;k=1;
while(k==1)
{
   if((sumf/2)<cf[i])
   {
            p=i;
            k=0;
   }
i++;
}
median=xl[p]+ ((sumf/2-cf[p-1])/f[p])*(xu[p]-xl[p]);
printf("\nthe required median is=%f",median);
max=f[1];
for(i=1;i<=n;i++)
{
   if(max<f[i])
   {
            max=f[i];
            p=i;
   }
}
mode=xl[p] + ((f[p]-f[p-1])/(2*f[p]-f[p+1]-f[p-1]))*(xu[p]-xl[p]);
printf("\nthe required mode is=%f",mode);
for(i=1;i<=n;i++)
{
   sumsd+=f[i]*(xm[i]- mean)*(xm[i]-mean);
}
sd=sqrt(sumsd/sumf);
printf("\n the required standard deviation is=%f",sd);
getch();
}
/*Output:-
enter the no of data4
enter the lower boundary20 30 40 50
enter the upper boundary30 40 50 60
enter the frequency13 12 14 10
```

the required mean is=39.285713
the required median is=39.583332
the required mode is=43.333332
the required standard deviation is=10.879676 */

9. Write a program in 'C' to determine the ***Correlation Co-eeficient*** between the following two random variables and the regression equation of y on x.

Variable(x):	38	48	43	40	41
Variable(y):	31	38	43	33	35

code:-

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
float x[100],y[100],sumy=0,sumx=0;
float meanx,meany,sdy=0,sdx=0,cov=0,b,r;
int i,n;
clrscr();
printf("enter the no of data");
scanf("%d",&n);
printf("enter the x values");
for(i=1;i<=n;i++)
{
        scanf("%f",&x[i]);
        sumx=sumx+x[i];

}
meanx=sumx/n;
printf("enter the y values");
for(i=1;i<=n;i++)
{
        scanf("%f",&y[i]);
        sumy=sumy+y[i];
}
meany=sumy/n;
for(i=1;i<=n;i++)
{
        cov=cov+(x[i]-meanx)*(y[i]-meany);
        sdx=sdx+pow((x[i]-meanx),2);
        sdy=sdy+pow((y[i]-meany),2);
}
sdx=sqrt(sdx/n);
```

```
sdy=sqrt(sdy/n);
printf("sdx=%f  sdy=%f\n",sdx,sdy);
cov=cov/n;
r=cov/(sdx*sdy);
b=cov/pow(sdx,2);
printf("correlation coefficient is=%f\n",r);
printf("regression line is (y-%f)=%f(x-%f)",meany,b,meanx);
getch();
}
/*Output:-
enter the no of data5
enter the x values38 48 43 40 41
enter the y values31 38 43 33 35
sdx=3.405877  sdy=4.195235
correlation coefficient is=0.643876
regression line is (y-36.000000)=0.793103(x-42.000000)*/
```