LABORATORY MANUAL

FOR

MATHEMATICS PRACTICALS

(WITH FOSS TOOLS)

FOR 6TH SEMESTER B.Sc.-PAPER-VIII

LIST OF PROGRAMS

Sl. No.	Title
01.	***Some problems on Cauchy-Riemann equations (polar form).***
02.	***Implementation of Milne-Thomson method of constructing analytic function(simple examples).***
03.	***Illustrating orthogonality of the surfaces obtained from the real and imaginary part of an analytic function.***
04.	***Verifying real and imaginary parts of an analytic function being harmonic (in polar form).***
05.	***Preservance of cross-ratio under bilinear transformation.***
06.	***Illustrating that circles are transformed to circles by a bilinear transformation.***
07.	***Examples connected with Cauchy's integral theorem.***
08.	***Solving algebraic equation (Bisection method).***
09.	***Solving algebraic equation (Regula-Falsi and Newton-Raphson methods).***
10.	***Solving system of equations (Jacobi and Gauss-Seidel methods).***
11.	***Solving for Largest Eigen value by Power method.***
12.	***Solving ordinary differential equation by modified Euler's method.***
13.	***Solving ordinary differential equation by Runge-Kutta method of 4th order.***

NOTE:
In each lab one program has to be executed and relevant problems have to be solved manually.

1.Cauchy-Riemann equations

A function *f(z)* is said to be analytic if it satisfies Cauchy-Riemann equations $\frac{\partial u}{\partial x}=\frac{\partial v}{\partial y}$ and $\frac{\partial u}{\partial y}=-\frac{\partial v}{\partial x}$.

When u and v are functions of r and θ, then these equations take the form,

$$\frac{\partial u}{\partial r}=\frac{1}{r}\frac{\partial v}{\partial \theta} \text{ and } \frac{\partial u}{\partial \theta}=-r\frac{\partial v}{\partial r}.$$

Maxima program to check the analyticity of the function (polar form)

```
kill(all)$
W:(r+1/r)*cos(theta)+%i*(r-1/r)*sin(theta)$
u:realpart(W)$
v:imagpart(W)$
ur:ratsimp(diff(u,r));
ut:ratsimp(diff(u,theta));
vr:ratsimp(diff(v,r));
vt:ratsimp(diff(v,theta));
if ur=ratsimp(1/r*vt) and ut=ratsimp(-r*vr) then
disp("the function is analytic")
else
disp("the function is not analytic");
```

Examples:

Check the analyticity of the following functions:

1. $(r^2\cos 2\theta - r\sin\theta)+i(r^2\sin 2\theta + r\cos\theta)$,
2. $\log r + i\theta$,
3. $\sin z$, where $z=r(\cos\theta+i\sin\theta)$,
4. $\frac{1}{r}\cos\theta-\frac{i}{r}\sin\theta$.

2.Construction of an analytic function using Milne-Thomson method.

Suppose we require an analytic function $f(z)=u+iv$, when u is given then we construct the analytic function as follows:

Since $f(z)$ is analytic, $f^{'}(z)$ exists and $f^{'}(z)=\frac{\partial u}{\partial x}+i\frac{\partial v}{\partial x}$.

By C-R equations, $f^{'}(z)=\frac{\partial u}{\partial x}-i\frac{\partial u}{\partial y}$. Let $\frac{\partial u}{\partial x}=\phi_1(x,y)$, $\frac{\partial u}{\partial y}=\phi_2(x,y)$ then $f^{'}(z)=\phi_1(x,y)-i\phi_2(x,y)=\phi_1(z,0)-i\phi_2(z,0)$. On integration, we get the required analytic function $f(z)=\int\phi_1(z,0)\,dz-i\int\phi_2(z,0)\,dz+c$ where c is a constant of integration.

Similarly when v is given, we have $f(z)=\int\psi_1(z,0)\,dz-i\int\psi_2(z,0)\,dz+c_1$ where $\frac{\partial v}{\partial y}=\psi_1(x,y)$, $\frac{\partial v}{\partial x}=\psi_2(x,y)$.

a).Maxima program to construct the analytic function when the real part is given, $u=e^x(x\cos y-y\sin y)$.

```
kill(all)$
u:%e^x*(x*cos(y)-y*sin(y));
f1:diff(u,x);
f2:diff(u,y);
f3:subst([x=z,y=0],f1);
f4:subst([x=z,y=0],f2);
W:integrate(f3,z)-%i*integrate(f4,z)+c;
```

Exercise:

Find the analytic function whose real parts are

1. $2x-x^3+3xy^2$, 2. $\frac{\sin 2x}{\cosh 2y+\cos 2x}$, 3. $\frac{1}{2}\log\left(x^2+y^2\right)$.

b).Maxima program to construct analytic function when the imaginary part is given, $v=x\sin x\sinh y-y\cos x\cosh y$.

```
kill(all)$
v:x*sin(x)*sinh(y)-y*cos(x)*cosh(y);
f1:diff(v,x);
f2:diff(v,y);
f3:subst([x=z,y=0],f2);
```

```
f4:subst([x=z,y=0],f1);
W:integrate(f3,z)+%i*integrate(f4,z)+c;
```

Exercise:
Find the analytic function whose imaginary parts are

$$1.\ \frac{x-y}{x^2+y^2},\quad 2.\ y+e^x\cos y,\quad 3.\ \frac{\sin 2x}{\cosh 2y-\cos 2x}.$$

3. Orthogonal surfaces

Two surfaces obtained from the real and imaginary parts of an analytic function, i. e., $u=c_1,\ \ v=c_2$, are orthogonal to each other if the product of their slopes is -1.

From $u=c_1$, we get, $\dfrac{dy}{dx}=-\dfrac{\partial u/\partial x}{\partial u/\partial y}=m_1$ and $v=c_2$ we get $\dfrac{dy}{dx}=-\dfrac{\partial v/\partial x}{\partial v/\partial y}=m_2.$

Using C-R equations we get $m_1 \square m_2=-1$. Hence the surfaces are orthogonal.

Maxima code to check the orthogonality of the surfaces

```
kill(all)$
z:x+%i*y$
W:sin(z)$
u:realpart(W)$
v:imagpart(W)$
ux:diff(u,x)$
uy:diff(u,y)$
vx:diff(v,x)$
vy:diff(v,y)$
y1:-ux/uy$
y2:-vx/vy$
if (y1*y2=-1) then
disp("surfaces are orthogonal")
else
disp("surfaces are not orthogonal");
```

Exercise:
Check whether the real and imaginary parts of the following analytic functions are orthogonal are not

$$1.\ x^3-3xy^2+i(3x^2y-y^3),\quad 2.\ ze^z,\quad 3.\ e^{-y}(\cos x+i\sin x),\quad 4.\ \cosh z.$$

4.Harmonic function

The real and imaginary parts of an analytic function will be harmonic if they satisfy Laplace equation i.e., if u and v are the real and imaginary parts of an analytic function then they are harmonic if

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \text{ and } \nabla^2 v = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} = 0 .$$

When the function is in polar coordinates, then

$$\nabla^2 u = \frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2} = 0 \text{ and} \nabla^2 v = \frac{\partial^2 v}{\partial r^2} + \frac{1}{r}\frac{\partial v}{\partial r} + \frac{1}{r^2}\frac{\partial^2 v}{\partial \theta^2} = 0 .$$

Maxima code for harmonic function (in polar coordinates)

```
kill(all)$
z:r*(cos(theta)+%i*sin(theta))$
W:z+1/z$
u:realpart(W)$
v:imagpart(W)$
ur:diff(u,r)$
vr:diff(v,r)$
urr:diff(u,r,2)$
utt:diff(u,theta,2)$
vrr:diff(v,r,2)$
vtt:diff(v,theta,2)$
u1:trigsimp(urr+1/r*ur+1/r^2*utt);
v1:trigsimp(vrr+1/r*vr+1/r^2*vtt);
if (u1=0) and (v1=0) then
disp("u and v are harmonic")
else
disp("u and v are not harmonic");
```

Exercise:

Verify whether real and imaginary parts of the following functions are harmonic

1. $\frac{\cos 2\theta - i\sin 2\theta}{r^2}(r \neq 0),$
2. $r^2 \cos 2\theta - r\sin\theta + i(r^2 \sin 2\theta + r\cos\theta),$
3. $\tan z,$ 4. e^{iz}, where $z = r(\cos\theta + i\sin\theta).$

5. Preservance of cross-ratio under bilinear transformation

Cross ratio: If z_1, z_2, z_3, z_4 are four distinct points, then the ratio $\frac{(z_1 - z_2)(z_3 - z_4)}{(z_2 - z_3)(z_4 - z_1)}$ is called the cross ratio of these points and is denoted by (z_1, z_2, z_3, z_4).

Preservance of cross ratio under bilinear transformation

A bilinear transformation preserves the cross ratio of four points i.e., If w_1, w_2, w_3, w_4 be the images of four distinct points z_1, z_2, z_3, z_4 in the z-plane under a bilinear transformation then $(w_1, w_2, w_3, w_4) = (z_1, z_2, z_3, z_4)$.

Maxima code to illustrate the preservance of cross ratio under bilinear transformation

```
kill(all)$
z:x+%i*y$
W(z):=(a*Z(z)+b)/(c*Z(z)+d);
c1:(W(1)-W(2))$
c2:(W(3)-W(4))$
c3:(W(2)-W(3))$
c4:(W(4)-W(1))$
cr1:factor(c1*c2)$
cr2:factor(c3*c4)$
cr:fullratsimp(cr1/cr2)$
if (cr=(fullratsimp((Z(1)-Z(2))*(Z(3)-Z(4)))/((Z(2)-Z(3))*(Z(4)-Z(1)))))
then
disp("Bilinear transformation preserves the cross-ratio of four points")
else
disp("Bilinear transformation does not preserve the cross-ratio of four points");
```

6. A bilinear transformation transforms circles to circles

One of the property of the bilinear transformation is that the image of a circle in z-plane will be transformed to a circle in w-plane.

Maxima code to illustrate that the bilinear transformation transforms circles to circles for $w = \dfrac{z - 2i}{z + 1}$.

```
kill(all)$
z:x+%i*y$
z1:2-%i$
z2:-1-%i$
a:1$b:-%i$c:1$d:-1+%i$
A:cabs((z-z1)/(z-z2))^2-2$
A1:fullratsimp(A)$
A2:num(A1)-denom(A1);
disp("The figure in Z-plane is:");
load(draw);wxdraw2d(implicit(A2,x,-8,2,y,-5,5));
kill(z)$
w1:solve((a*z+b)/(c*z+d)-w,z)$
w1[1]$
z:rhs(w1[1])$
subst(w1[1],(z-z1)/(z-z2))$
factor(%)$
exp:fullratsimp(%,w)$
w2:fullratsimp(num(exp)/coeff(num(exp),w))$
w3:denom(exp)/coeff(denom(exp),w)$
exp1:fullratsimp(w2/w3,w)$
p1:subst(w=u+%i*v,exp1)$
exp2:cabs(p1)^2-2$
exp3:fullratsimp(exp2)$
exp4:num(exp3)-denom(exp3);
disp("The figure in W-plane is:");
load(draw);wxdraw2d(implicit(exp4,u,-8,5,v,-2,7));
disp("Conclusion: Bilinear transformation transforms circles
in Z-plane to circles in W-plane");
```

Exercise

Show that the following transformations transform circles to circles

1. $w = \dfrac{2z - i}{z + i}$; $z_1 = i, z_2 = 2i + 3$, 2. $w = \dfrac{(1 - i)z - 2}{z + 3i - 2}$; $z_1 = 2, z_2 = -1$.

7.Cauchy’s integral theorem

Statement: If a function $f(z)$ is analytic at all points within and on a closed contour C then $\int\limits_C f(z)\,dz = 0.$

Generalized Cauchy’s integral formula

Statement: If $f(z)$ is analytic inside and on a simple closed curve C and if ‘a’ is any point within C then $f^n(a) = \frac{n!}{2\pi i}\int\limits_C \frac{f(z)}{(z-a)^{n+1}}dz.$

Scilab code for evaluating Cauchy’s integrals

Ex: Evaluate $\boldsymbol{\int\limits_C \frac{e^z}{(z\text{-}2)(z\text{-}5)^3}dz}$ **where** $\boldsymbol{C:|z| = 8.}$

```
clc;
clear;
funcprot(0);
function y=f(z)
 y =%e^z/((z-5)^3*(z-2));
endfunction
```

Note: For evaluating the integral call the function using the command *intl* i.e., *intl(0,2*%pi,0,8,f)*.

Exercise:

Evaluate

1. $\int\limits_C \frac{2z+2}{z^2+2z+2}dz$ where $C:|z+i| = 2,$

2. $\int\limits_C \frac{e^{2z}}{z+i\pi}dz$ where $C:$Unit circle with center at origin,

3. $\int\limits_C \frac{z}{(z^2+1)(z^2-9)}dz$ where $C:|z| = 2,$

4. $\int\limits_C \frac{dz}{(z^2+4)^2}$ where $C:|z-i| = 2,$

5. $\int\limits_C \frac{\sin\pi z^2+\cos\pi z^2}{(z-1)^2(z-2)}dz$ where $C:|z| = \frac{1}{2}.$

8. Bisection method

The method consists of locating the root of the equation *f(x)*=0 between '*a*' &'*b*' (*a*<*b*) If f(x) is continuous in the interval [*a,b*] and *f(a)* & *f(b)* are of opposite signs then there is a root between '*a*' & '*b*'. For definiteness, *f(a)* be negative and *f(b)* be positive. Then the first approximation to the root is $x_1 = \frac{1}{2}(a+b)$. If $f(x_1)$ =0 then x_1 is a root of *f(x)*=0. Otherwise, the root lies between '*a*'& 'x_1' or 'x_1' and '*b*' accordingly as $f(x_1)$ is positive or negative. Then we bisect the interval as before and continue the process until the root is found to the desired accuracy.

Scilab code for finding the root of $x^3 - 9x + 1 = 0$, using bisection method

```
clc;
clear;
funcprot(0);
function y=f(x)
   y = x^3 - 9*x + 1;
endfunction
function bisection(a, b)
   for i = 1 : 5000
      c = (a+b)/2;
      if abs (f(c)) <= 0.00000001 then
         disp(c);
         break;
      else
         if (f(a)*f(c)) < 0 then
            b = c;
         else
            a = c;
         end
      end
   end
endfunction
```

Exercise:

Find the root of the following equations by bisection method

1. $2x = 3 + \cos x$, 2. $xe^x - 1 = 0$,

3. $x \log_e x = 12$, 4. $\cos x - xe^x = 0$.

9(i). Regula Falsi method

The Regula –Falsi method is based on replacing the part of the curve between the points (x_1, $f(x_1)$) and ($x_{2,}$ $f(x_2)$) by the chord joining these two points and then taking the point of intersection of the chord with x- axis as an approximation to the root .We obtain $x = \dfrac{x_1 f(x_2) - x_2 f(x_1)}{f(x_2) - f(x_1)}$, which gives the first approximation. Using this equation we get a sequence of approximations till we get the root to the desired accuracy.

Scilab code to find the root of $x^3 + 4x^2 - 10 = 0,$ using Regula-Falsi method

```
clc;
clear;
funcprot(0);
function y=f(x)
  y = x^3 + 4*x^2 - 10;
endfunction
function regulafalsi(a, b)
  for i = 1 : 5000
    c = (a * f(b) –b* f(a))/(f(b) - f(a));
    if abs (f(c)) <= 0.00001 then
      disp(c);
      break;
    else
      if (f(a)*f(c)) < 0 then
        b = c;
      else
        a = c;
      end
    end
  end
endfunction
```

Exercise:

Find the root of the following equations by Regula-Falsi method

1. $x^4 - x - 10 = 0,$ 2. $\cos x = 3x - 1,$

3. $\tan x + \tanh x = 0,$ 4. $x e^x - x^2 = 4.$

(ii).Newton-Raphson method

Assuming that x_0 is an approximate value of a real root of the equation *f*(*x*)=0, let x_1 be the exact root and $x_1 = x_0 + h$, where *h* is small correction.

Using Taylor's expansion and neglecting higher powers of *h* $(h^2, h^3, ...)$, we get $x_1 = x_0 - \dfrac{f(x_0)}{f'(x_0)}$.

In general $x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}, \quad n = 0,1,2,...$

This is called Newton - Raphson iterative formula.

Scilab code to find the root of $x^3 + 4x^2 - 10 = 0$, using Regula-falsi method

```
clc;
clear;
function [y]=f(x)
   y = cos(x) - (x*exp(x));
endfunction

function y=df(x)
   y = -sin(x)-x*exp(x)-exp(x);
endfunction

x0 = input('Enter the initial condition   ');
for i = 1 : 1000
   xn = x0 - (f(x0)/df(x0));
   if (abs(f(xn))) <= 0.00001 then
      mprintf('Root is   ');
      disp(xn);
      disp(i);
      abort;
   else
      x0 = xn;
   end
end
```

Exercise:

Find the root of the following equations by Newton-Raphson method

2. $x\sin x + \cos x = 0,$ 2. $x^2 - 4\sin x = 0,$

3. $\tan x + x = 0,$ 4. $x^2 \log x = 2,$

10(i). Jacobi iteration method (Gauss-Jacobi method)

Consider the system of equations:

$a_1x+b_1y+c_1z=d_1;\ a_2x+b_2y+c_2z=d_2;\ a_3x+b_3y+c_3z=d_3;$

From the given system we have,

$$x=\frac{1}{a_1}\left[d_1-b_1y-c_1z\right], \tag{1}$$

$$y=\frac{1}{a_2}\left[d_2-a_2x-c_2z\right], \tag{2}$$

$$z=\frac{1}{a_3}\left[d_3-a_3x-b_3y\right]. \tag{3}$$

Initially we give the values $x=x_0, y=y_0, z=z_0$ using these values in the equations (1), (2), (3) we have,

$$x_1=\frac{1}{a_1}\left[d_1-b_1y_0-c_1z_0\right]; y_1=\frac{1}{a_2}\left[d_2-a_2x_0-c_2z_0\right];\ z_1=\frac{1}{a_3}\left[d_3-a_3x_0-b_3y_0\right];$$

Similarly the second iterative values of the unknowns can be calculated by substituting first iterative values in equations (1), (2) and (3).

The above process is repeated until two consecutive iterative values are same.

Scilab code to find the solution of system of equations using Jacobi iteration method

Ex: ***10x + y + z = 12, 2x + 10y + z = 13, 2x + 2y + 10z = 14.***

```
clc;
clear;
funcprot(0);
n = 3;
A = [10 1 1; 2 10 1; 2 2 10];
B = [12; 13; 14];
xold = [0; 0; 0];
x = xold;
for itr = 1 : 500
   for i = 1 : n
      sum = 0;
      for j = 1 : n
         if i <> j then
            sum = sum + A(i,j) * xold(j);
         end
      end
      x(i) = (B(i) - sum)/A(i,i);
```

```
    end
    if abs (max(x - xold)) <= 0.00001 then
        mprintf('The required solution is ');
        disp(x);
        break;
    else
        xold = x;
    end
end
```

Exercise:
Solve the system of equations by Gauss-Jacobi method

1. $5x-2y+z=-4,\ x+6y-2z=-1,\ 3x+y+5z=13.$
2. $8x+y+z=8,\ 2x+4y+z=4,\ x+3y+5z=5.$
3. $9x+2y+2z=20,\ x+10y+4z=6,\ 2x-4y+10z=-15.$
4. $20x+2y+6z=28,\ x+20y+9z=-23,\ 2x-7y-20z=-57.$

(ii). Gauss-Seidel method

Consider the system of equations:

$a_1x+b_1y+c_1z=d_1;\ a_2x+b_2y+c_2z=d_2;\ a_3x+b_3y+c_3z=d_3;$

From the given system we have

$$x=\frac{1}{a_1}\left[d_1-b_1y-c_1z\right], \tag{1}$$

$$y=\frac{1}{a_2}\left[d_2-a_2x-c_2z\right], \tag{2}$$

$$z=\frac{1}{a_3}\left[d_3-a_3x-b_3y\right]. \tag{3}$$

Initially we give the values $x=x_0, y=y_0, z=z_0$ using these values in the equations (1), (2), (3) we have,

$x_1=\frac{1}{a_1}\left[d_1-b_1y_0-c_1z_0\right]; y_1=\frac{1}{a_2}\left[d_2-a_2x_1-c_2z_0\right];\ z_1=\frac{1}{a_3}\left[d_3-a_3x_1-b_3y_1\right];$

Similarly the second iterative values of the unknowns can be calculated by substituting first iterative values in equations (1), (2) and (3).

The above process is repeated until two consecutive iterative values are same.

Scilab code to find the solution of system of equations using Gauss-Seidel method

Ex: $\boldsymbol{10x + 2y + z = 9,\ x + 10y - z = -22,\ -2x + 3y + 10z = 22.}$

```
clc;
clear;
funcprot(0);
n = 3;
A = [10 2 1; 1 10 -1; -2 3 10];
B = [9; -22; 22];
xold = [0; 0; 0];
x = xold;
for itr = 1 : 500
   for i = 1 : n
      sum = 0;
      for j = 1 : n
         if i <> j then
            sum = sum + A(i,j) * x(j);
         end
      end
      x(i) = (B(i) - sum)/A(i,i);
   end
   if abs (max(x - xold)) <= 0.00001 then
      mprintf('The required solution is ');
      disp(x);
      break;
   else
      xold = x;
   end
end
```

Exercise:

Solve the system of equations by Gauss-Seidel method

1. $x + y + 54z = 110,\ 27x + 6y - z = 85,\ 6x + 15y + 2z = 72.$
2. $5x + 2y + z = 12,\ x + 4y + 2z = 15,\ x + 2y + 5z = 0.$
3. $28x + 4y - z = 32,\ 2x + 77y + 4z = 35,\ x + 3y + 10z = 24.$
4. $30x - 2y + 3z = 75,\ 2x + 2y + 18z = 30,\ 3x + 17y - 2z = 48.$

11. Power method to compute the largest eigen value of a square matrix

Suppose A is the given square matrix, we assume initially an eigen vector X_0 in a simple form like $\begin{bmatrix}1\\0\\0\end{bmatrix}$ or $\begin{bmatrix}0\\1\\0\end{bmatrix}$ or $\begin{bmatrix}0\\0\\1\end{bmatrix}$ or $\begin{bmatrix}1\\1\\1\end{bmatrix}$ and find the matrix AX_0 which will also be a column matrix. Take of the largest element as the common factor to obtain $AX_0 = \lambda^{(1)}X^{(1)}$, we then compute $AX^{(1)}$ and again put it in the form $AX^{(1)} = \lambda^{(2)}X^{(2)}$ by normalization .This iterative process is continued till two consecutive iterative values of λ and X are same up to a desired accuracy. The values so obtained are respectively the largest eigen value and the corresponding eigen vector of the given square matrix A.

Scilab code to compute the largest eigen value by power method

Ex: $\begin{bmatrix}-15 & 4 & 3\\ 10 & -12 & 6\\ 20 & -4 & 2\end{bmatrix}$

```
clc;
clear;
funcprot(0);
A = [-15 4 3; 10 -12 6; 20 -4 2];
X = [1; 0; 0];
small = 0;
for  i = 1 : 400
   X1 = A*X
   big = max(X1);
   lambda = max(X1)
   X1 = X1/lambda;
   if abs(big - small ) <= 0.00001 then
      mprintf("The required eigenvectors are ");
      disp(X1);
      mprintf("The required eigevnvalue is %f", lambda);
     break;
   else
      X = X1;
      small = big;
   end
   end
```

Exercise:

Find the largest eigen value of the following matrices, by power method

$$1.\begin{bmatrix}25 & 1 & 2\\ 1 & 3 & 0\\ 2 & 0 & -4\end{bmatrix},\quad 2.\begin{bmatrix}6 & -2 & 2\\ -2 & 3 & -1\\ 2 & -1 & 3\end{bmatrix},\quad 3.\begin{bmatrix}2 & 0 & 1\\ 0 & 2 & 0\\ 1 & 0 & 2\end{bmatrix},\ 4.\begin{bmatrix}1 & 6 & 1\\ 1 & 2 & 0\\ 0 & 0 & 2\end{bmatrix}.$$

12. Modified Euler's method

Consider the initial value problem $\frac{dy}{dx} = f(x,y);\ y(x_0) = y_0,$ we need to find y at $x_1 = x_0 + h.$ First approximation $y(x_1) = y_1$ is obtained by Euler's formula $y_1 = y_0 + h\,f(x_0, y_0).$ For the accuracy, this value is modified and is given by $y_1^{(1)} = y_0 + h\left[f(x_0,y_0) + f(x_1,y_1)\right].$ The second modified value of y_1 is given by $y_1^{(2)} = y_0 + h\left[f(x_0,y_0) + f(x_1,y_1^{(1)})\right].$ The third modified value of y_1 is given by $y_1^{(3)} = y_0 + h\left[f(x_0,y_0) + f(x_1,y_1^{(2)})\right]$ and so on. The procedure is repeated till two consecutive values of y are equal to the desired degree of accuracy.

Scilab code to solve the initial value problem by modified Euler's method

Ex: $\boldsymbol{\frac{dy}{dx} = 2x;\ y(0) = 0,}$ **compute** $\boldsymbol{y(0.2)}$ **by taking** $\boldsymbol{h=0.1.}$

```
funcprot()
clc
clear
function [y]=f(x, y)
  z=2*x;
endfunction
function eulermodified(xi, yi, h, xf)
   x(1)=xi
   y(1)=yi
  n=(xf-xi)/h;
   for i=2:n+1
      x(i)=x(i-1)+h
      y(i)=y(i-1)+h*f(x(i-1),y(i-1));
      y(i,1)=y(i);
      for j=2:20
         y(i,j)=y(i-1)+h*(f(x(i-1),y(i-1))+f(x(i),y(i,j-1)))/2;
         if abs(y(i,j)-y(i,j-1))<10^(-4) then
```

```
            y(i)=y(i,j);
            break;
         end
      end
       mprintf("y(%2f)=%f \n " ,x(i),y(i));
   end
endfunction
```

Exercise:
Solve the following initial value problems by applying modified Euler's method

1. $\frac{dy}{dx}=x^2+y;\ y(0)=1,$ in the range $0\le x\le 0.06$ by taking $h=0.02.$
2. $\frac{dy}{dx}=1+\frac{y}{x};\ y(1)=2,$ compute y at x=1.4 by taking h=0.2.
3. $\frac{dy}{dx}=x+y;\ y(0)=1,$ compute y at x=0.2 by taking h=0.1.
4. $\frac{dy}{dx}=\frac{y-x}{y+x};\ y(0)=1,$ compute $y(0.1).$

13. Runge-Kutta method of 4th order

Consider the initial value problem $\frac{dy}{dx}=f(x,y);\ y(x_0)=y_0,$ The approximate solution of the equation at $x_1=x_0+h,$ is given by

$$y_1=y(x_0+h)=y_0+\tfrac{1}{6}(k_1+2k_2+2k_3+k_4),$$

where
$$k_1=h\,f(x_0,y_0),\ k_2=h\,f(x_0+\tfrac{h}{2},y_1+\tfrac{k_1}{2}),$$
$$k_3=h\,f(x_1+\tfrac{h}{2},y_1+\tfrac{k_2}{2}),\ k_4=h\,f(x_1+h,y_1+k_3).$$

A more generalised formula is as follows

$$y_1=y(x_0+h)=y_0+\tfrac{1}{6}(k_1+2k_2+2k_3+k_4),$$

where
$$k_1=h\,f(x_n,y_n),\ k_2=h\,f(x_n+\tfrac{h}{2},y_n+\tfrac{k_1}{2}),$$
$$k_3=h\,f(x_n+\tfrac{h}{2},y_n+\tfrac{k_2}{2}),\ k_4=h\,f(x_n+h,y_n+k_3).$$

Scilab code to solve the initial value problem by Runge-Kutta 4th order method

Ex: $\frac{dy}{dx} = 2x;\ y(1) = 2,$ **find the approximate solution at** x**=1.2.**

```
funcprot()
clc
clear
function [z]=f(x, y)
  z=x*y;
endfunction
function rk4(xi, yi, h, xf)
   x(1)=xi
   y(1)=yi
     n=(xf-xi)/h;
   for i=2:n+1
      x(i)=x(i-1)+h;
      k1=h*f(x(i-1),y(i-1));
      k2=h*f(x(i-1)+h/2,y(i-1)+k1/2);
      k3=h*f(x(i-1)+h/2,y(i-1)+k2/2);
      k4=h*f(x(i-1)+h,y(i-1)+k3);
      y(i)=y(i-1)+(k1+2*k2+2*k3+k4)/6;
      mprintf("y(%2f)=%f \n " ,x(i),y(i));
   end
endfunction
```

Exercise:

Solve the following initial value problems by Runge-Kutta 4th order method

1. $\frac{dy}{dx} = \frac{1}{x+y};\ y(0.4) = 1,$ at x=0.5,
2. $\frac{dy}{dx} = 3e^{x} + 2y;\ y(0) = 0,$ at x=0.1,
3. $\frac{dy}{dx} = 1 + y^{2};\ y(0) = 0,$ for x=0.2(0.2)0.4,
4. $\frac{dy}{dx} = x^{2}y + x;\ y(0) = 1,$ compute $y(0.1)$ and $y(0.2)$.