The Iteration Method for non-linear equation

Theory

Let the equations be given by $f(x,y) = 0$, $g(x,y) = 0$ ……………………(1)

Whose real roots are required within a specified accuracy. As in the method of iteration for a single equation, it is assumed that the equations in (1) can be written in the following form,

$$x = F(x,y) \text{ and } y = G(x,y) \text{ ………………(2)}$$

where the functions F and G satisfy the conditions,

$$\left|\frac{\partial F}{\partial x}\right| + \left|\frac{\partial F}{\partial y}\right| < 1 \quad \text{and} \quad \left|\frac{\partial G}{\partial x}\right| + \left|\frac{\partial G}{\partial y}\right| < 1 \text{…….(3)}$$

In the neighborhood of the root. Let (x0,y0) be the initial approximation to a root (ζ , η) of the system (1). Then it is constructed the successive approximation according to the following formulae,

$$\left.\begin{array}{ll} x_1 = F(x_0,y_0), & y_1 = G(x_0,y_0) \\ x_2 = F(x_1,y_1), & y_2 = G(x_1,y_1) \\ x_3 = F(x_2,y_2), & y_3 = G(x_2,y_2) \\ \dots\dots & \dots\dots \\ x_{n+1} = F(x_n,y_n), & y_{n+1} = G(x_n,y_n) \end{array}\right| \text{…………… (4)}$$

This process continues until the required accuracy is obtained.

Code

```
#include<bits/stdc++.h>
using namespace std;
double F(double x,double y)
{
   return y + 4.00/(x+y);
}
double G(double x,double y)
{
   return sqrt(16.00 + 2*x*y)-x;
}
double diff_Fx(double x,double y)
{
   return -4/((x+y)*(x+y));
}
```

```
double diff_Fy(double x,double y)
{
    return 1-4/((x+y)*(x+y));
}
double diff_Gx(double x,double y)
{
    return (y/sqrt(16.00 + 2*x*y))-1;
}
double diff_Gy(double x,double y)
{
    return x/(sqrt(16.00 + 2*x*y));
}
int Check(double diff_Fx,double diff_Fy,double diff_Gx,double diff_Gy)
{
    if((fabs(diff_Fx)+fabs(diff_Fy))<1.00 && (fabs(diff_Gx)+fabs(diff_Gy))<1.00 )
        return 1;
    else
        return 0;
}
void Iteration(double x, double y)
{
    double X,Y,m,n;
    do
    {
        X = F(x,y);
        Y = G(x,y);
        m = x;
        n = y;
        x = X;
        y = Y;
        printf("x: %0.6lf\ty: %0.6lf\tError rate (x) : %0.6lf\tError rate (y) : %0.6lf\n",X,Y,fabs(X-
m),fabs(Y-n));
    }while(fabs(X-m)>=0.0001 && fabs(Y-n)>=0.0001);
    cout<<"\nFinal x: "<<X<<"\tFinal y: "<<Y<<endl;
}

int main()
{
    double x,y;
    int z;
    cout<<" Assume the x0 : ";
    cin>>x;
    cout<<" Assume the y0 : ";
    cin>>y;
    cout<<endl;
```

```
    z = Check(diff_Fx(x,y),diff_Fy(x,y),diff_Gx(x,y),diff_Gy(x,y));
    while(z!=1)
    {
        x += 0.5;
        y += 0.5;
        z = Check(diff_Fx(x,y),diff_Fy(x,y),diff_Gx(x,y),diff_Gy(x,y));
    }
    Iteration(x,y);
    return 0;
}
```

Output

```
"E:\Study\My C\Lab\2-1\CSE 2104\Lab 4\IterationNonLinear.exe"
 Assume the x0 : 2
 Assume the y0 : 2

x: 3.300000     y: 2.838539     Error rate (x) : 0.800000       Error rate (y) : 0.338539
x: 3.490160     y: 2.593586     Error rate (x) : 0.190160       Error rate (y) : 0.244953
x: 3.251076     y: 2.349708     Error rate (x) : 0.239084       Error rate (y) : 0.243878
x: 3.063894     y: 2.341612     Error rate (x) : 0.187182       Error rate (y) : 0.008096
x: 3.081598     y: 2.445090     Error rate (x) : 0.017704       Error rate (y) : 0.103478
x: 3.168851     y: 2.492410     Error rate (x) : 0.087252       Error rate (y) : 0.047320
x: 3.198966     y: 2.469957     Error rate (x) : 0.030116       Error rate (y) : 0.022453
x: 3.175558     y: 2.440415     Error rate (x) : 0.023408       Error rate (y) : 0.029542
x: 3.152669     y: 2.436871     Error rate (x) : 0.022889       Error rate (y) : 0.003544
x: 3.152493     y: 2.447804     Error rate (x) : 0.000176       Error rate (y) : 0.010933
x: 3.162052     y: 2.454054     Error rate (x) : 0.009558       Error rate (y) : 0.006250
x: 3.166291     y: 2.452188     Error rate (x) : 0.004240       Error rate (y) : 0.001866
x: 3.164125     y: 2.448750     Error rate (x) : 0.002166       Error rate (y) : 0.003439
x: 3.161397     y: 2.448032     Error rate (x) : 0.002728       Error rate (y) : 0.000718
x: 3.161117     y: 2.449164     Error rate (x) : 0.000280       Error rate (y) : 0.001133
x: 3.162141     y: 2.449961     Error rate (x) : 0.001024       Error rate (y) : 0.000796
x: 3.162706     y: 2.449832     Error rate (x) : 0.000565       Error rate (y) : 0.000129
x: 3.162522     y: 2.449441     Error rate (x) : 0.000184       Error rate (y) : 0.000391
x: 3.162204     y: 2.449325     Error rate (x) : 0.000318       Error rate (y) : 0.000117
x: 3.162143     y: 2.449438     Error rate (x) : 0.000061       Error rate (y) : 0.000113

Final x: 3.16214        Final y: 2.44944
```

Discussion

In the above code, the iteration method of finding the roots of non-linear equations was performed. Here firstly the value of x_0 and y_0 were taken from the user input. After that the Check() function was called to check the condition of iteration method whether this assumption is suitable for further procedure or not. If the Check() returns 1 then the condition satisfies and Iteration() function is called where the equations no (4) from the theory were used with the help of a do while loop. The loop continues until the required accuracy level is reached. When the loop terminates , the actual value of x and y is obtained. Thus the iteration method for non-linear equations was successfully implemented.