**Unit 5: Solution of Ordinary Differential Equations**

**Need and Scope**

Many of the laws in physics, chemistry, engineering, biology and economics are based on empirical observations that describe changes in the states of systems. Mathematical models that describe the state of such systems are often expressed in terms of not only certain system parameters but also their derivatives. Such mathematical models which use differential calculus to express relationship between variables are known as differential equations.

**Number of Independent Variables**

An equation which uses differential calculus to express relationship between variables is known as differential equations. Quantity being differentiated is called dependent variable and the quantity with respect to which the dependent variable is differentiated is called independent variable. The differential equations have applications in all areas of science and engineering. Differential equations are used to model problems in science and engineering that involve the change of some variables with respect to another. Mathematical formulation of most of the physical and engineering problems leads to differential equations of two types.

* **Ordinary Differential equations (ODE):** If there is only one independent variable the equation is called an ordinary differential equation.
* **Partial differential equations (PDE):** If it contains two or more independent variables, the derivatives will be partial and therefore, the equation is called a partial differential equation

**Ordinary and Partial differential equations**

An ordinary differential equation with single independent variable is called ordinary differential equation. Examples of ordinary differential equations include

..

A differential equation with more than one independent variable is called partial differential equation. An example of such an equation would be

…

Where u is the dependent variable and x and y are independent variables.

**Order of Equations**

Ordinary differential equations are classified in terms of order. Order of an ordinary differential equation is the same as the highest derivative that appears in the equation. When the equation contains only a first a first derivative, it is called a first order differential equation. On the other hand, if the highest derivative is second derivative, the equation is called a second order differential equation.

A first order equation can be expressed in the form

= f(x,y)

A second order equation can be expressed in the form

y’’ = f(x,y,y’)

where y’’ denotes the second derivative and t’ is the first derivative. Higher order equations can be reduced to a set of first order equations by suitable transformations. For example, the equation

y’’ = f(x,y,y’)

can be equivalently represented by

u’ = f(x,y,u)

y’ = u

**Degree of Equations**

The degree of a differential equation is the power of the highest order derivative. For example,

xy’’+y2y’ = 2y+y

is a first degree, second order equation while,

(y’’’)2+5y’ = 0

Is second degree, third order equation?

**Linear and Nonlinear Equations**

A differential equation is known as a linear equation when it does not contain terms involving the products of the dependent variable or its derivatives. For example,

y’’+3y’ = 2y+x2

is a second order, linear equation.

The equations

Y’’+(y’)2 =1

And y’ = -ay2

are non linear because the first one contains a product of y’ and the second contains a product of y

**General and Particular Solutions**

A solution to a differential equation is a relationship between the dependent and independent variables that satisfy the differential equation. For example,

y = 3x2+x

is the solution of y’ = 6x+1

Similarly,

y = ex

is the solution of y’’ = y

Note that each of the solutions given above is only one of an infinite number of solutions. For example,

y = 3x2+x+2

y = 3x2+x-10

are also solution of y’ = 6x+1.

In general, y’ = 6x+1 has a solution of the form

y’ = 3x2+x+c

where c is known as the constant of integration. Similarly y’ = y has a solution of the form y = aex

The solution that contains arbitrary constants is not unique and is therefore known as the general solution.

If the values of the constants are known then, on substitution of these values in the general solution, a unique solution known as particular solution can be obtained.

**Initial vs Boundary Values Problem**

In order to obtain the values of the integration constants, we need additional information. For example, consider the solution y = aex to the equation y’ = y. If we are given a value of y for some x, the constant a can be determined. Suppose y = 1 at x=0, then

y(0)= ae0 = 1

Therefore a=1

And the particular solution is

y = ex

If the order of the equation is n, we will have to obtain n constants and therefore, we need n conditions in order to obtain a unique solution. When all the conditions are specified at a particular value of the independent variable x, then the problem is called an initial value problem.

It is also possible to specify the conditions are different values of the independent variable. Such problems are called the boundary value problem. For example, if, instead of specifying only y(0) = 1, we also specify y(0)+y(1) = 2 then the problem will be a boundary value problem. In this case,

y(0)+y(1) = a(1+e) = 2

Giving a = 2(1+e)

**One Step and Multistep Methods**

All numerical techniques for solving differential equations involve a series of estimates of y(x) starting from the given conditions. There are two basic approaches that could be used to estimate the values of y(x). They are known as one step methods and multistep methods.

In one step methods, we use information from only one preceding point. That is, to estimate the value of yi, we need the conditions at the precious point yi-1. Multistep methods use information at two or more previous steps to estimate a value.

**Scope**

In this chapter, we mainly concentrate on the solutions of ordinary differential equations and discuss the following methods

1. Taylor series Method
2. Picard’s Method
3. Euler Method
4. Heun’s Method
5. Runge Kutta Method

**Solving Initial Value Problems**

**Taylors Series Method**

Taylor series expansion of function y(x) about a point x=x0 is given by the relation

y(x) = y(x0)+(x-x0)y’(x0)+(x-x0)2 + …. (1)

Value of the function y(x) can be calculated if we know the values of its derivatives at that point. Thus, if we are given the equation y’ = f(x,y), we can differentiate it repeatedly and calculate them at x = x0. Finally these values can be substituted in equation (1) to obtain y(x).

That is, if

y’ = f(x,y) then

y’’ = f(x,y) = [f(x,y)+[f(x,y)]

= fx+fyf

Where f denotes the function f(x,y), fx denotes partial derivatives of f(x,y) with respect to x and fy denotes partial derivative of f(x,y) with respect to y

Similarly, we can compute

y’’’ = fxx+2f fxy+f2yy+fxfy+ff2y

**Algorithm for Taylor’s Method**

1. Start
2. Read initial values say x0 and y0
3. Read the value at which function to be evaluated say x
4. Compute the value of y’, y’’ and y’’’…
5. Compute v = y(x) = y0+(x-x0)y’+(x-x0)2y’’/2!+….+(x-x0)3y’’’/3!+….
6. Display function value at x
7. Terminate

**C program to solve ODE by using Taylor’s series method**

#include<conio.h>

#include<stdio.h>

#include<math.h>

int fact(int n)

{

if(n<=1)

return 1;

else return n\*fact(n-1);

}

int main()

{

float xp, x0,y0,yx,fdy,sdy,tdy;

printf("Enter initial value of x and y\n");

scanf("%f%f",&x0,&y0);

printf("Enter x at which function is to be evaluated\n");

scanf("%f",&xp);

fdy = x0\*x0+y0\*y0;

sdy = 2\*x0+2\*y0\*fdy;

tdy = 2+2\*y0\*sdy+2\*fdy\*fdy;

yx = y0+ (xp-x0)\*fdy+(xp-x0)\*(xp-x0)\*sdy/fact(2)+(xp-x0)\*(xp-x0)\*(xp-x0)\*tdy/fact(3) ;

printf("Function value at x = %f is %f\n",xp,yx);

getch();

return 0;

}

Output:

Enter initial value of x and y

0 1

Enter x at which function is to be evaluated

0.5

Function value at x = 0.500000 is 1.916667

**Example 1 : Given y’ = x-y2 with the initial condition y = 1 when x=0. Find y for x=0.1 by using first four terms of the series.**

**Solution:** From Taylor’s series method, we know that

y’ = x-y2

y’’ = fx+fyf = 1-2y(x-y2) = 1-2yy’

y’’’ = fxx+2 f fxy +f2 fxy +fx fy +f fy2 = -2(y’)2-2yy’’

Now, values of derivatives can be calculated at x=0 as follows:

y’(0) = 0-(1)2 = -1

y’’ (0) = 1 – 2\*1\*(-1) = 3

y’’’(0) = -2\*(-1)2-2\*1\*3 = -2-6 =-8

Substituting above values in the Taylor series we get

y(x) = y(x0)+(x-x0)y’(x0)+(x-x0)2 + (x-x0)3 +……+(x-x0)n +

y(x) = 1-x+x2 - x3

This is the solution of given differential equation.

Now, put x=0.1 in above equation we get.

y(0.1) = 0.913

**Example 2: Solve the differential equation y’ = 3x2 such that y = 1 at x=1. Find y for x=2 using first four terms.**

**Solution:**

From the Taylor’s series we have

Y’ = 3x2

Y’’ = 6x

Y’’’ = 6

Now, values of derivatives can be calculated at x=1 as follows:

Y’(1) = 3

Y’’(1) = 6

Y’’’(1) = 6

Substituting above values in the Taylor series we get

y(x) = y(x0)+(x-x0)y’(x0)+(x-x0)2 + (x-x0)3 +……+(x-x0)n

= 1+3(x-1)+3(x-1)2+(x-1)3

This is the solution of given differential equation

Now, put x=2 in above equation we get

y(2) = 1+3+3+1 = 8

**Improving Accuracy**

The error in Taylor method is in the order of (x-x0)n+1. If |x-x0| is large, the error can also become large. Therefore, the result of this method in the interval (x0,b) is larger, is often found unsatisfactory. The accuracy can be improved by dividing the entire interval into subintervals (x0,x1),(x1,x2),(x2,x3),… of equal length and computing y(xi), i=1,2…n successively, using the Taylor series expansion. Here, y(xi) is used as an initial condition for computing y(xi+1). Thus,

y(xi+1) = y(xi) +(xi+1-xi) + (xi+1-xi)2 +.......................+(xi+1-xi)m

If we denote the size of each subinterval as h, then

xi+1-xi =h, for i=0,1,..n-1

Now above equation becomes

yi+1 = yi + \*h + \*h2 +…. …+ \*hm

Example: Use the Taylor method recursively to solve the equation

y’ = x2+y2 , y(0) =0

For the interval (0, 0.4) using two subinterval of size 0.2

Solution:

Y’ = y’ = x2+y2

Y’’ = 2x+2yy’

y’’’ = 2+2(y’)2+2yy’

y(4) = 6y’y’’+3yy’’’

Iteration 1:

y1 = y0 + \*h + \*h2 + \*h2+ \*h4

h=0.2, y0 = y(0) = 0

y’0 = y’(0) = 0+y(0)2 = 0

y’’0 = y’’(0) = 2\*0+2\*y(0)\*y’(0) = 0

Similarly,

Y’’’0 = 2

Y(4)0 = 0

Therefore

Y1 = 0+0+0+2/3! \*(0.2)3+0 = 0.002667

Iteration 2:

X1 = 0.2

Y1 = 0.002667

Y’1 = x12+y12 = (0.2)2+(0.006667)2 =0.04

……………….

That is,

y(0.4) = 0.0231352

If we use h = (b-x0)/4 = 0.4 (without subdividing), we obtain

y(0.4) = 0.021333

The correct answer to the accuracy shown as y(0.4) = 0.021359. it shows that the accuracy has been improved by using subintervals. The accuracy can be further improved by reducing h further say, h = 0.1

One major problem with the Taylor series method is the evaluation of higher order derivatives. They become very complicated. All these derivatives must be evaluated at (xi,yi) i=0.1…. This method is, therefore, generally impractical from a computational point of view. However, it illustrates the basic approach to numerical solution of differential equations.

**Picard Method**

Consider the differential equation

= f(x,y)

We can integrate this to obtain the solution in the interval (x0,x)

=

Or

y(x) = y(x0)+

Since y appears under the interval sign on the right, the integration cannot be formed. The dependent variable should be replaced by either a constant or a function of x. Since, we know the initial value of y at x=x0), we may use this as a first approximation to the solution and the result can be used on the right hand side to obtain the next approximation. The iterative equation can be written as

yi+1 = y0+ ………(1)

Equation (1) is known as Picard’s method. Since this method involves actual integration, sometimes it may not be possible to carry out the integration.

It can be seen that Picard’s method is not convenient for computer based solutions.

**Example:** Solve the following equations by Picard’s method

1. y’(x) = x2+y2, y(0) =0
2. y’(0) = xey, y(0) = 0

and estimate y(0.1), y(0.2) and y(1)

1. y’(x) = x2+y2

y0 = 0, x0 = 0

y(1) = y0+ (x2+(y0)2)dx

= 0+ (x2dx = x3/3

y(2) = y0+ (x2+(y1)2)dx = 0+ (x2+x6/9)dx = x3/3+x7/63

This process can be continues further although it may be a difficult task. If we stop at y(2), then

y(x) = x3/3+x7/63

Now, y(0.1) = 0.00003333

y(0.2) = 0.0026666

y(1) = 0.3492

**Algorithm for Picard Method**

1. start
2. Read initial values of x and y say x0 and y0
3. Read the value at which functional value is required, say x
4. Set y=y0
5. Compute estimates value of y as below

Compute my = y0+

Compute Error = |(ny-y)/ny)|

If Error<E

Then go to step 6

Else

Set y = ny and goto step 5

1. Display the functional value y
2. Stop

**C program to solve ODE by using Picard’s method**

#include<conio.h>

#include<stdio.h>

#include<math.h>

#define y1(x) 2+(x)-2/3\*pow(x,3)

#define y2(x) 2+(x)+pow(x,2)-2/3\*pow(x,3)+pow(x,4)/4

#define y3(x) 2+(x)+pow(x,2)-pow(x,4)/3-pow(x,5)/15

int main()

{

float x,x0,y0,y,ny,er;

printf ("Enter initial value of x and y\n");

scanf("%f%f",&x0,&y0);

printf("Enter x at which function is to be evaluated\n");

scanf("%f",&x);

y = y0;

y = y0+y1(x);

y = y0+y2(x);

y = y0+y3(x);

printf("Function value at x = %f is %f\n", x, y);

getch ();

return 0;

}

Output:

Enter initial value of x and y

0 2

Enter x at which function is to be evaluated

0.4

Function value at x = 0.400000 is 4.550784

**Euler’s Method:**

Euler’s method is the simplest one step method and has a limited application because of its low accuracy. However, it is discussed here as it serves as a starting point for all other advanced methods.

Consider the first two terms of the expansion of Taylor’s series

y(x) = y(x0)+ y’(x0)(x-x0)

Given the differential equation

y’(x0) = f(x,y) with y(x0) = y0

We have

y’(x0) = f(x0,y0)

and therefore

y(x) = y(x0)+(x-x0)f(x0,y0)

Then, the value of y(x) at x =x1 is given by

y(x1) = y(x0)+(x-x0)f(x0,y0)

Letting h = x1-x0, we obtain

y1 = y0+hf(x0,y0)

Similarly, y(x) at x =x2 is given by

y2 = y1+hf f(x1,y1)

In general, we obtain a recursive relation as

yi+1 = yi+h f(xi,yi)

This formula is known as Euler’s method and can be used recursively to evaluate y1, y2,… of y(x1), y(x2),…. Starting from the initial condition y0 = y(x0). Note that this does not involve any derivatives

A new value of y is estimates using the previous value of y as the initial condition. Note that the term h f(xi,yi) represents the incremental value of y and f(xi, yi) is the slope of y(x) at (xi,yi). That is, the new value is obtained by extrapolating linearly over the step size h using the slope at its previous value. That is,

New value = old value \*slope \*step size

**Example: Given the equation**

= 3x2+1 with y(1) = 2

Estimate y(2) by Euler’s method using (i) h=0.5 and (ii) h=0.25

Solution: h=0.5

y(1) = 2

y(1.5) = 2+ 0.5(2\*1.02+1) = 4.0

y(2.0) = 4.0+0.5(2\*1.52+1)= 7.875

1. h = 0.25

y(1) = 2

y(1.25) = 2+0.25\*(3\*12+1) = 3.0

y(1.50) = 3+0.25\*(3\*1.252+1) = 5.42188

y(1.75) = 5.42188+0.25\*(3\*1.52+1) = 7.3593

y(1.20) = 7.35593+0.25\*(3\*1.752+1) = 9.90626

Notice the difference in answers of y(2) in these two cases. The accuracy is improved considerably when h is reduced to 0.25. True answer is 10.0

**Algorithm for Euler ‘s Method**

1. start
2. Read initial values of x and y say x0 and y0
3. Read the value at which functional value is required, say xp
4. Read the step size say h
5. Set y=y0 and x = x0
6. Compute the value of y as below

For x=x0 to xp

y = y+f(x,y)

y = x+h

End for

1. Display the functional value y
2. Stop

**C program for Euler Method**

#include<stdio.h>

#include<conio.h>

#include<math.h>

#define f(x,y) 2\*y/x

int main()

{

float x,xp,x0,y1,y,h,y0;

printf("Enter initial values of x and y\n");

scanf("%f%f",&x0,&y0);

printf("Enter x at which function to be evaluated \n");

scanf("%f",&xp);

printf("Enter the step size\n");

scanf("%f",&h);

y = y0;

x= x0;

for(x=x0;x<xp;x = x+h)

{

y = y+f(x,y)\*h;

}

printf ("Function value at x = %f is %f\n",xp,y);

getch ();

return 0;

}

**Output**

Enter initial values of x and y

1 2

Enter x at which function to be evaluated

2

Enter the step size

0.25

Function value at x = 2.000000 is 7.200000

**Accuracy of Euler’s Method**

As usual, the accuracy is affected by two sources of error, namely round off error and truncation error. Rounding off error is always present in a computation and this can be minimized by increasing the precision of calculations.

The major cause for loss of accuracy is truncation error. This arises because of the use of a truncated Taylor Series. Since Euler’s method uses Taylor series iteratively, the truncation error introduced in iteration is propagated to the following iterations. This means the total truncation error in any iteration step will consists of two components. The propagated truncation error and the truncation error introduced by the step itself

The truncation introduced by the step itself is known as the local truncation error and the sum of the propagated error and the local error is called the global truncation error.

**Heun’s Method**

Euler’s method is the simplest of all one-step methods. It does not require any differentiation and is easy to implement on computers. However, its major weakness is large truncation errors. This is due to its linear characteristics. Recall that Euler’s method uses only the first two terms of the Taylor’s Series.

In Euler’s method, the slope at the beginning of the interval is used to extrapolate yi to yi+1 over the entire interval. Thus,

yi+1 = yi+m1h

Where m1 is the slope at (xi, yi). This is clearly an underestimate of y(xi+1)

An alternative is to use the line which is parallel to the tangent at the point (xi+1,y(xi+1)) to extrapolate from yi to yi+1. That is

yi+1 = yi+m2h

Where m2 is the slope at (xi+1, y(xi+1)). Note that the estimate appears to be overestimated.

A third approach is to use a line whose slope is the average of the slopes at the end points of the interval.

Then

yi+1 = yi + \*h…………(1)

This gives better approximation to yi+1. This approach is known as Heun’s method

The formula for implementing Heun’s method can be constructed easily.

Given the formula

y’(x) = f(x,y)

we can obtain

m1 = y’(xi) = f(xi, yi)

m2 = y’(xi+1) = f(xi+1,yi+1)

and therefore

m =

Now, the equation (1) becomes,

yi+1 = yi + \*h ………..(2)

Note that the term yi+1 appears on both sides of equation (2) and therefore yi+1 cannot be evaluated until the value of yi+1 inside the function f(xi+1,yi+1) is available. This value can be predicted using the Euler’s formula as

yi+1 = yi+h\* f(xi, yi)

Then, Heun’s formula becomes

yi+1 = yi + \*h …..(3)

Equation (3) is improved version of Euler’s method. Since it attempts to correct the value of yi+1 using the predicted value of yi+1 (by Euler’s method), it is classified as a one step predictor corrector method.

**Algorithm**

1. start
2. Read initial values x and y say x0 and y0
3. Read the value at which functional value is required say xp
4. Read step size say h
5. Set x =x0 y=y0
6. Compute values of y(xp) as below

For x= x0 to xp

m1 = f(x,y)

m2 = f(x+h,y+h\*m1)

y = y+h/2\*(m1+m2)

End for

1. Display functional value y
2. stop

**C Program**

#include<stdio.h>

#include<conio.h>

#define f(x,y) 2\*(y)/(x)

int main()

{

float x, xp, x0,y0, y,h,m1,m2;

printf("Enter initial value of x and y\n");

scanf("%f%f",&x0,&y0);

printf("Enter the value at which function is to be evaluated\n");

scanf("%f",&xp);

printf("Enter step size\n");

scanf("%f",&h);

y = y0;

x = x0;

for(x=x0;x<xp;x=x+h)

{

m1 = f(x,y);

m2 = f(x+h,y+h\*m1);

y = y+h/2\*(m1+m2);

}

printf("Function value at x %f =%f",x,y);

getch();

return 0;

}

Output:

Enter initial value of x and y

1 2

Enter the value at which function is to be evaluated

2

Enter step size

0.25

Function value at x 2.000000 =7.860846

**Example: Given the equation**

**y’(x) = 2y/x with y(1) = 2**

Estimate y(2) using (i) Euler’s method, and (ii) Heun’s method, using h=0.25 and compare the results with exact answers.

Solution:

Given y’ = f(x,y) = 2y/x

x0 = 1, y0 = 2, h=0.25

1. Euler’s method:

Y(1.25) = y1 = y0+h.f(x0,y0) = 2+0.25\*f(x0,y0) = 2+ 0.25\* 2\*2/1 = 3.0

Y(1.5) = 3.0+0.25\*2\*3.0/1.25 = 4.2

Y(1.75) = 4.2+0.25\*2\*4.2/1.5 = 5.6

Y(2.0) = 5.6+ 0.25\*2\*5.6/1.75 = 5.6

1. Heun’s Method

Iteration 1:

m1 = = 4.0

Ye(1.25) = 2+0.25\*(4.0) = 3.0

m2 = = 4.8

y(1.25) = 2+ (4.0+4.8) =3.1

Iteration 2:

m1 = = 4.96

ye(1.5) = 3.1+0.25\*4.96 = 4.24

m2 = = 6.79

y(1.5) = 3.1+ (4.96+5.79) =4.44

Iteration 3:

m1 = = 5.92

ye(1.75) = 4.44+0.25\*(5.92) = 5.92

y(1.75) = 4.44+ (5.92+6.77) = 6.89

Iteration 4:

m1 = = 6.89

ye(2.0) = 6.03+0.25\*(6.89) = 7.75

y(2.0) = 6.03+ (6.89+6.75) = 6.86

Exact solution of the equation

Y’(x) = 2y/x with y(1) = 2

Is obtained as

y(x) = 2x2

The exact values of y(x) and the estimated values by both the methods are tabulated below

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X | Y(x) | | | |
| Euler’s Method | Heun’s Method | Analytical Method | |
| 1.00 | 2.00 | 2.00 | 2.00 |  |
| 1.25 | 3.00 | 3.10 | 3.125 |  |
| 1.50 | 4.20 | 4.44 | 4.50 |  |
| 1.75 | 5.60 | 6.03 | 6.126 |  |
| 2.00 | 7.20 | 7.86 | 8.00 |  |

**Fourth Order Runge Kutta Methods**

Heun’s method can be further refined by replacing the average slope of two points with a slope that is the weighted average of f(x,y) at four points within the interval. This refinement in the Heun’s method improves the order of approximation from h2 to h4. This refinement was carried out by Two German mathematicians CDT Runge and M.W. Kutta using The Taylor series expansion with remainder of function y(x)

Given the initial value problem

= f(x,y) , y(x0) = y0

For a fixed constant of h, y(xn+h) can be approximated by

Y(xn+h) = yn+1 = yn+\*h(m1+2m2+2m3+m4)

m1 = f(xi,yi). { This is the slope at (xi,yi)

m2 = f(xi + ,yi+) { The slope at the midpoint of the interval along the line connecting (xi,yi) and (xi+h, yi+hm1) }

m3 = f(xi + , yi + ) { The slope at the midpoint of the interval along the line connecting (xi,yi) and (xi+h, yi+hm2)}

m4 = (xi+h, yi+m3h) { slope at (xi+h, yi+hm3)}

**Algorithm**

1. start
2. Read initial values x and y say x0 and y0
3. Read the value at which functional value is required say xp
4. Read step size say h
5. Set x =x0 y=y0
6. Compute values of y(xp) as below

For x= x0 to xp

m1 = f(x,y)

m2 = f(x+h/2,y+h/2\*m1)

m3 = f(x+h/2,y+h/2\*m2)

m4 = f(x+h, y+h\*m3)

y = y+\*h(m1+2m2+2m3+m4)End for

1. Display functional value y
2. stop

**C program**

#include<stdio.h>

#include<conio.h>

#define f(x,y) 2\*(y)/(x)

int main()

{

float x, xp, x0,y0, y,h,m1,m2,m3,m4;

printf("Enter initial value of x and y\n");

scanf("%f%f",&x0,&y0);

printf("Enter the value at which function is to be evaluated\n");

scanf("%f",&xp);

printf("Enter step size\n");

scanf("%f",&h);

y = y0;

x = x0;

for(x=x0;x<xp;x=x+h)

{

m1 = f(x,y);

m2 = f(x+1/2.0\*h,y+1/2.0\*h\*m1);

m3 = f(x+1/2.0\*h,y+1/2.0\*h\*m2);

m4 = f(x+h,y+h\*m3);

y = y+(m1+2\*m2+2\*m2+m4)\*h/6;

}

printf("Function value at x %f =%f",x,y);

getch();

return 0;

}

Output:

Enter initial value of x and y

1 2

Enter the value at which function is to be evaluated

2

Enter step size

0.25

Function value at x 2.000000 =7.952565

**Example 1: Use the Runge Kutta method to estimate y(0.4) if y’ = 2x+y with y(0) = 1**

**Solution:**

Here f(x,y) = 2x+y, x0 = 0 and y0 = 1, h=0.4

Now from Runge Kutta method, we have

m1 = f(x0,y0) = (2\*0+1} = 1

m2 = f(x0+h/2,y0+h/2\*m1) = (0+0.4/2,1+0.4/2\*1) = f(0.2,1.2) = 1.6

m3 = f(x+h/2,y+h/2\*m2) = 1.72

m4 = f(x+h, y+h\*m3) = 1.72

hence y(0.4) = 1+1/6\*0.4(1.0+2(1.6)+2(1.72)+2.488) = 1.675

**Example 2: Use the classical RK method to estimate y(0.4) when**

y’(x) = x2+y2 with y(0) = 0 assume h=0.2

**Solution:**

**Iteration 1:**

f(x,y) = x2+y2

m1 = f(x0,y0) = 0

m2 = f(xi + ,yi+) = f(x0+0.02/2, 0) = 0.01

m3 = f(xi + , yi + ) = f(0.2/2, (0.01\*0.2)/2) = 0.01

m4 = (xi+h, yi+m3h) = f(0.2/2, 0.01\*0.2) = 0.04

y(0.2) = 0 + )\*0.2 = 0.002667

Iteration 2:

x1 = 0.2

y1 = 0.002667

m1 = f(0.2,0.002667) = 0.04

m2 = f(0.3, 0.002667+) = 0.090044

m3 = f(0.3,0.002667+) = 0.090136

m4 = f(0.4,0.002667+(0.090136)\*0.2) = 0.160428

Y(0.4) = 0.002667+ \*0.2 = 0.021360 (Correct to six decimal places)

The exact answer is 0.021359. if we take h=0.1 then y(0.4) will be 0.021359

**Solving System of ordinary Differential Equations**

Many mathematical problems require solving the system of several first order differential equations represented as below

= f1(x,y1,y2…...yn) with y1(x0) = y10

= f1(x,y1,y2…...yn) with y2(x0) = y10

= f1(x,y1,y2…...yn) with y3(x0) = y10

…………………………………..

= f1(x,y1,y2…...yn) with yn(x0) = yn0

The solutions of such a system requires that n initial conditions be known at the stating value of x. single equation methods can be used to solve systems of ODE’s as well. We can use Euler’s method or Heun’s method or Runge Kutta method for solving system of ordinary differential equations.

**Algorithm for Solving System of Two ODE’s**

1. Start
2. Read initial values of x,y and z say x0, y0, and z
3. Read the value at which functional value is required say xp
4. Read step size say h
5. Set y = y0 and x = x0
6. Compute value of y as below

For x =x0 to xp

ny = y+f(x,y,z)\*h

nz = z+f(x,y,x)\*h

y = ny

z = nz

End for

1. Display functional value y
2. Stop

**C Program for solving system of ODE’s by using Euler’s method**

#include<stdio.h>

#include<conio.h>

#include<math.h>

#define f1(x,y,z) z

#define f2(x,y,z) exp(-x)-2\*(z)-(y)

int main()

{

float x, xp, x0,y0,z0,y,z,ny,nz,h;

printf("Enter initial value of x and y and z\n");

scanf("%f%f%f",&x0,&y0,&z0);

printf("Enter the value at which function is to be evaluated\n");

scanf("%f",&xp);

printf("Enter step size\n");

scanf("%f",&h);

y = y0;

x = x0;

z = z0;

for(x=x0;x<xp;x=x+h)

{

ny = y+(f1(x,y,z))\*h;

nz = z+(f2(x,y,z))\*h;

y = ny;

z = nz;

}

printf("Function value at x %f =%f",x,y);

getch();

return 0;

}

Output:

Enter initial value of x and y and z

0 1 2

Enter the value at which function is to be evaluated

0.75

Enter step size

0.25

Function value at x 0.750000 =1.829925

**Example 1: Solve the following two simultaneous first order differential equations with step size 0.25**

= z = f1(x,y,z), y(0) = 1

= e-x – 2z-y = f2(x,y,z), z(0) = 2

Use Euler method to find y(0.75)

Solution:

From Euler’s method, we have

yi+1 = yi+f1(xi,yi,zi)\*h

zi+1 = zi+f2(xi,yi,zi)\*h

here,

f1(x,y,z) =z and f2(x,y,z) = e-x – 2z-y

Iteration 1:

x0 = 0, y0=1, z0 =2

y1 = y0+f1(x0,y0,z0)\*h = 1+f1(0,1,2)\*0.25 = 1+2\*0.25 = 1.5

🡪y(0.25) = 1.5

z1 = z0+f2(x0,y0,z0)\*h = 2+ f2(0,1,2)\*0.25 = 2+ (e-0-2\*2-1)\*0.25 = 1

🡪z(0.25) =1

Iteration 2

x1 = 0.25, y1=1.5, z1 =1

Y2 = y1+f1(x0,y0,z0)\*h = 1.5+f1(0.25,1.5,1)\*0.25 = 1.5+1\*0.25 = 1.75

🡪y(0.50) = 1.75

z2 = z1+f2(x0,y0,z0)\*h = 1+ f2(0.25,1.5,1)\*0.25 = 2+ (e-0.25-2\*1-1.5)\*0.25 = 0.31970

🡪z(0.75) =0.

Iteration 3

x2 = 0.5, y2=1.75, z2 =0.31970

y3 = y2+f1(x0,y0,z0)\*h = 1.75+f1(0.50,1.75,0.31970)\*0.25 = 1.75+0.31970\*0.25 = 1.8299

🡪y(0.75) = 1.8299

z3 = z2+f2(x0,y0,z0)\*h = 1+ f2(0.50,1.75,0.31972)\*0.25 = 0.31972+ (e-0.50-2\*0.31972-1.75)\*0.25 =-0.126

🡪z(0.75 ) = -0.126

**Example 2: Solve the following two simultaneous first order differential equations with step size 0.1**

= x+y+z = f1(x,y,z), y(0) = 1

= 1+y+z = f2(x,y,z), z(0) = -1

Use Heun’s method to find y(0.75)

Solution:

………………

**Higher Order Differential Equations**

Many scientific and engineering problems are modeled as differential equations that are higher than first order. An nth order differential equation has the form

an+ an-1 +…….. a1+ a0y = f(x)

with n-1 initial conditions, it can be solved by assuming

y(x0) = a1, y’(x0) = a2, y’’(x0) = a3, ……..yn-1(x0) = an

Let us denote

y = z1

= z2

= z3

….

= zn

Then above equation represents n first order differential equations are follows:

= z2 , z1(x0) = z10 = a1

= z3 , z2(x0) = z20 = a2

…………………………………..

= zn , zn-1(x0) = zn-10 = an-1

= f(x,z1 , z2……zm) zn(x0) = zn0 = an

Each of the n first order ordinary differential equations are accompanied by one initial condition. These first order ordinary differential equations are simultaneous in nature and hence can be solved by the methods used for solving system of first order ordinary differential equations that we have already learned.

**Algorithm for solving second order differential equations**

1. Start
2. Read initial values of x and y say x0 and y0
3. Read the value at which functional value is required say xp
4. Read the step size say h
5. Reduce the equation into system of differential equations
6. Set x = x0 and y= y0
7. Compute value of y as below

For x=x0 to xp

ny = y+f(x,y,z)\*h

nz = z+ f(x,y,z)\*h

y = ny

z = nz

x = x+h

1. Display functional value, y
2. stop

**C program to solve second degree ODE using Euler’s method**

#include<stdio.h>

#include<conio.h>

#include<math.h>

#define f1(x, y, z) z

#define f2(x, y, z) 6\*(x)+3\*(y)-2\*(z)

int main()

{

float xp, x0,y0,z0,x,y,z,h,m1,m2;

printf("Enter initial value of x and y and z\n");

scanf("%f%f%f",&x0,&y0,&z0);

printf("Enter the value at which function is to be evaluated\n");

scanf("%f",&xp);

printf("Enter step size\n");

scanf("%f",&h);

y = y0;

x = x0;

z = z0;

for(x=x0;x<xp;x=x+h)

{

m1 = f1(x,y,z);

m2 = f2(x,y,z);

y = y+m1\*h;

z = z+m2\*h;

}

printf("Function value x at %f =%f", x, y);

getch();

return 0;

}

Output:

Enter initial value of x and y and z

0 0 1

Enter the value at which function is to be evaluated

0.2

Enter step size

0.1

Function value x at 0.200000 =0.180000

**Example 1: Rewrite the following differential equation as a set of first order differential equations**

**+ 2 - 3y = 6x with y(0) = 0, y’(0) = 1**

**Find y(0.2) by Euler’s method with step size 0.1**

**Solution:**

First, the second order differential equation is rewritten as two simultaneous first order differential equations as follows:

Assume

= z

Then

+2z-3y = 6x

= 6x+3y-2z

So, the two simultaneous first order differential equation are

= z , y(0)= 0

= 6x+3y-2z, z(0) = 1

From Euler’s method, we have

yi+1 =yi + f1(xi,yi,zi)\*h

zi+1 =zi + f2(xi,yi,zi)\*h

Here,

f1(x,y,z) = z

f2(x,y,z) = 6x+3y-2z

Iteration 1:

x0 = 0, y0 = 0, z0 = 1

y1 =y0 + f1(x0,y0,z0)\*h = 0 + f1(0,0,1)\*0.1 = 1\*0.1 = 0.1

* y(0.1) = 0.1

z1 =z0 + f2(x0,y0,z0)\*h = 1 + f2(0,0,1)\*0.1 = 1-2\*0.1 = 0.8

Iteration 2:

x1 = 0.1, y1 = 0.1, z0 = 0.8

y2 =y1 + f1(x0,y0,z0)\*h = 0.1 + f1(0.1,00.1,0.8)\*0.1 = 0.1+0.8\*0.1 = 0.18

* y(0.2) = 0.18

z2 =z1 + f2(x0,y0,z0)\*h = 0.8 + f2(0.1,0.1,0.8)\*0.1 = 0.8+6\*0.1+3\*0.1-2\*0.8 = 0.73

**Example 2:** Rewrite the following differential equation as a set of first order differential equations

+ 2 - y = e-x with y(0) = 0, y’(0) = 2

Find y(0.5) by Heun’s method with step size 0.25

**Solution:**

First, the second order differential equation is rewritten as two simultaneous first order differential equations as follows:

Assume

= z

Then

+2z+y = e-x

= e-x -y-2z

So, the two simultaneous first order differential equation are

= z , y(0)= 0

= e-x -y-2z, z(0) = 2

Now, using Heun’s method on equation (1) and (2), we get

**Solving Boundary Value Problem**

In this section we show how to approximate the solution to boundary value problems, differential equations with conditions imposed at different points. For first order differential equations, only one condition is specified so, there is no distinction between initial value and boundary value problems. We will consider second order equations with two boundary values

Ordinary differential equations are given either with initial conditions or with boundary conditions.

We have seen that we require m conditions to be specified in order to solve an m-order differential equations and all m conditions are specified at one point, x=x0 and therefore, we call this problem as an initial value problem. It is not always necessary to specify the conditions at one point of the independent variable. They can be specified at different points in the interval (a,b) and therefore such problems are called the boundary value problems. A large number of problems fall into this category.

In solving initial value problems, we move in steps from the given initial value of the x to the point where the condition is required. In case of boundary value problems, we seek solutions at specified points within the solutions at specified points within the domain of given boundaries, for instance, given

= f(x,y,y’) y(a) = ya ,y(b) = yb

We are interested in finding the values of y in the range a ≤ x ≤ b

There are two popular methods available for solving the boundary value problems. The first one is known as the shooting. This method makes use of the techniques of solving initial value problems. The second one is called the finite difference method which makes use of the finite difference equivalents of derivatives.

**Shooting Method**

In this method given boundary value problem is first transferred into equivalent initial value problem and then it is solved by using any of the method used for solving initial value problem.

The main steps involved in shooting methods are:

* Transform boundary value problem into equivalent initial value problem
* Get solution of initial value problem by using any existing method
* Get the solution of boundary value problem

Consider the boundary value problem

y’’ = f(x,y,y’) y(a) =u and y(b) = v

Let

Y’ = z, now we can obtain following set off two equations

y’ = z

z’ = f(x, y, z)

To solve above initial value problem, we need to have two conditions at x=a. We have given one condition y(a) = u. Let’s guess another condition is z(a) = g1. Here g1 represents slope of y(x) ay x=a. Thus the, problem can be written as system of two first order equations as below:

Y’ = z , y(a) =u

Z’ = f(x,y,z) , z(a) = g1 (1)

Now, equation (1) can be solved by using any method for solving initial value problem until the solution at x = b reaches to specified accuracy level. Suppose, first estimated value of y(x) at x = b is given by (b) = v1. If v1 = v then we are done and it is the required solution. Otherwise, we should repeat the same process by taking second guess say g2. Suppose v2 is the estimates value at y(b) for second guess. If solution is not achieved from second guess, we can obtain better approximation by using linear interpolation as below:

=

This gives

g3 = g2 - (g2-g1)

Now, with z(a) = g3, solution of y(x) can be obtained. We have to repeat this process until solution with desired level of accuracy is achieved

**Algorithm**

1. Start
2. Read boundary conditions say xa, xb, ya and yb
3. Read the point at which solution is needed say xp
4. Read the accuracy limit say E
5. Convert higher order differential equation to system of differential equations
6. Read the value of h
7. Approximate first approximation as below

Set x = xa , y = ya

g1 = (yb-ya)/(xb-xa)

Calculate y(xb) by using Euler’s method

Set v1 = y

If(y<yb)

g2 = 2g1

else

g2 = g1/2

Calculate y(xb) by using Euler’s method

Set v2 = y

1. Compute new values of y(xb) as below

Compute g3 = g2 - (g2-g1)

Find y(xb) using Euler’s method

Compute error

If(error<E)

Display solution

Go to step 9

Else

Set v1 = v2

v2 = y(xb)

Set g1 = g2

g2 = g3

Go to step 8

1. Stop

**C program for solving boundary value problem using shooting method**

#include<stdio.h>

#include<conio.h>

#include<math.h>

#define f1(x,y,z) z

#define f2(x,y,z) 6\*(x)

int main()

{

float xa,xb,ya,yb,x0,y0,x,y,z,xp,h,sol,ny,nz,error,E, g[3],v[3],gs;

int i;

printf("Enter boundary conditions\n");

scanf("%f%f%f%f",&xa,&ya,&xb,&yb);

printf("Enter x at which value is required\n");

scanf("%f",&xp);

printf("Enter step size\n");

scanf("%f",&h);

printf("Enter accuracy level\n");

scanf("%f",&E);

x = xa;

y = ya;

g[1] = (yb-ya)/(xb-xa);

printf("g1 = %f\n",g[1]);

while(x<xb)

{

ny = y+(f1(x,y,z))\*h;

nz = z+(f2(x,y,z))\*h;

x = x+h;

y = ny;

z = nz;

if(x==xp)

sol = y;

}

v[1] = y;

if(y<yb)

g[2]=z = 2\*g[1];

else

g[2] = z = g[1]/2;

printf("g = %f\n",g[2]);

while(x<xb)

{

ny = y+(f1(x,y,z))\*h;

nz = z+(f2(x,y,z))\*h;

x = x+h;

y = ny;

z = nz;

if(x==xp)

sol = y;

}

while(1)

{

x = xa;

y = ya;

gs = z = g[2]-(v[2]-yb)/(v[2]-v[1])\*(g[2]-g[1]);

while(x<xb)

{

ny = y+(f1(x,y,z))\*h;

nz = z+(f2(x,y,z))\*h;

x = x+h;

y = ny;

z = nz;

if(x==xp)

sol = y;

}

error = fabs(y-yb)/y;

v[1] = v[2];

v[2] = y;

g[1] = g[2];

g[2] = gs;

if(error<E)

{

printf("y(%f) %f\n",xp,sol);

break;

}

}

getch();

return 0;

}

**Example:** Solve the differential equation given below by using shooting method with Euler’s method. And calculate the value of y(1.5) by using h = 0.5

= 6x, y(1) = 2 , y(2) = 9

Solution:

Let =z

Then

= 6x

This gives us two first order differential equations

=z y(1) = 2

= 6x z(1) = unknown

Let us assume that

Z(1) = = 7

Now, set up the initial value problem as

=z y(1) = 2

= 6x z(1) = 6

Where

f1(x,y,z) = z

f2(x,y,z) = 6x

From Euler’s method, we know that

yi+1 = yi+ f1(x, y, z)\*h

zi+1 = zi+ f2(x, y, z)\*h

**Calculate first approximation**

**Iteration 1**:

x0 = 1, y0 = 2, z0 = 7

y1 = y0+f1(x0, y0, z0)\*h = 2+f1(1,2,7)\*0.5 = 2+3.5 = 5.5

z1 = z0+f2(x0,y0,z0)\*h = 7+f2(1,2,7)\*h = 7+6\*1\*0.5 = 10

**Iteration 2**:

x1 = 1.5, y1 = 5.5, z1 = 10

y2 = y1+f1(x0,y0,z0)\*h = 5.5+f1(1.5,.5.5,10)\*0.5 = 5.5+10\*0.5 =10.5

z2 = z1+f2(x0,y0,z0)\*h = 10+f2(1.5,5.5,10)\*h = 10+6\*1.5\*0.5 = 14.5

Thus y(2) = 10.5

The given value of this boundary condition is :y(2) =9

Since, predicted value of y(2) is higher than actual value

Let us assume that

Z(1) =1/2\* = 3.5

**Calculate second approximation**

**Iteration 1**:

x0 = 1, y0 = 2, z0 = 3.5

y1 = y0+f1(x0,y0,z0)\*h = 2+f1(1,2,3.5)\*0.5 = 2+3.5\*0.5 = 3.75

z1 = z0+f2(x0,y0,z0)\*h = 3.5+f2(1,2,3.5)\*h = 3.5+6\*1\*0.5 = 6.5

**Iteration 2**:

x1 = 1.5, y1 = 3.75, z1 = 6.5

y2 = y1+f1(x0,y0,z0)\*h = 3.75+f1(1.5,.3.75,6.5)\*0.5 = 3.75+3.75\*0.5 =7

z2 = z1+f2(x0,y0,z0)\*h = 6.5+f2(1.5,3.75,6.5)\*h = 6.5+6\*1.5\*0.5 = 11

Thus y(2) = 7

Since predicted value y(2) is less than actual value

We use linear interpolation on the previous guesses to obtain new guess as below

g3 = g2 - (g2-g1) = 3.5 – (3.5-7) = 5.5

Thus new guess is g3 = 5.5

Iteration 1;

**Calculate third approximation**

**Iteration 1**:

x0 = 1, y0 = 2, z0 = 5.5

y1 = y0+f1(x0,y0,z0)\*h = 2+f1(1,2,5.5)\*0.5 = 2+5.5\*0.5 = 4.75

z1 = z0+f2(x0,y0,z0)\*h = 5.5+f2(1,2,5.5)\*h = 5.5+6\*1\*0.5 = 8.5

**Iteration 2**:

x1 = 1.5, y1 = 4.75, z1 = 8.5

y2 = y1+f1(x0,y0,z0)\*h = 3.75+f1(1.5,.3.75,6.5)\*0.5 = 3.75+3.75\*0.5 =7

z2 = z1+f2(x0,y0,z0)\*h = 8.5+f2(1.5,4.75,8.5)\*h = 8.5+6\*1.5\*0.5 = 13

Thus y(2) = 9

And the given value of this boundary condition is y(2) = 9

Thus, we can use third approximation to obtain value y(2)

**Example 2:** Solve the ordinary differential equation given below by using shooting method with Euler’s method. And calculate the value of y(3) and y(6) by using h=3

-2y = 72x-8x2, y(0) = 0 , y(9) = 0

Solution:

Let = z

Then

-2y = 72x-8x2

This gives us two first order differential equations

= z y(0) = 0

Then

- 2y = 72x-8x2  z(0) = unknown

Let us assume that

z(0) =