Bachelor of Science
(B.Sc.- PCM)

PARTIAL DIFFERENTIATION EQUATIONS - LAB
(DBSPDS303P24)

Self-Learning Material
(SEM-III)

Jaipur National University
Centre for Distance and Online Education

Established by Government of Rajasthan
Approved by UGC under Sec 2(f) of UGC ACT 1956
&
NAAC A+ Accredited

TABLE OF CONTENT

EXPERT COMMITTEE

Dr. Vikas Gupta
Dean,
Department of Mathematics
LNMIIT, Jaipur

Dr. Nawal Kishor Jangid
Department of Mathematics
SKIT, Jaipur

COURSE COORDINATOR

Prof. (Dr.) Hoshiyar Singh
Dept. of Basic Science
JNU, Jaipur

UNIT PREPARATION

Unit Writers	**Assisting & Proof Reading**	**Unit Editor**
Dr. Sanju Jangid Dept. of Basic Science JNU, Jaipur Unit: 1-4	Mr. Mohhamad Asif Dept. of Basic Science JNU, Jaipur	Dr, Yogesh Khandelwal Dept. of Basic Science JNU, Jaipur

Secretarial Assistance:

Mr. Suresh Sharma
Senior office Assistant, JNU

COURSE INTRODUCTION

Partial Differential Equations (PDEs) are a fundamental class of equations that describe a wide array of physical phenomena, including heat conduction, wave propagation, fluid dynamics, and quantum mechanics. They are equations that involve rates of change with respect to continuous variables. Understanding and solving PDEs is crucial for students and professionals in fields such as mathematics, physics, engineering, and computer science.

This laboratory manual is designed to provide a hands-on approach to learning PDEs, focusing on both the theoretical aspects and practical applications. The lab sessions will guide you through the process of formulating, solving, and interpreting the results of various PDEs. You will explore classical methods of solution, including separation of variables, Fourier series, and transform methods, as well as numerical techniques for approximating solutions to more complex equations.

Course Outcomes: After the completion of the course, the students will be able to:

1. Recall the Understanding of PDE Basics.
2. Explain Solution Techniques of PDE.
3. Understand and apply methods for solving non-linear PDEs in special cases.
4. Analyze and Implement numerical methods for solving PDEs, including finite difference methods, finite element methods, and other discretization techniques.
5. Utilize software tools and programming languages to implement and solve PDEs numerically uniform continuity, differentiation, integration and uniform convergence.
6. Create a practical understanding of the existence, uniqueness, and regularity of solutions to PDEs.

Acknowledgements:

The content we have utilized is solely educational in nature. The copyright proprietors of the materials reproduced in this book have been tracked down as much as possible. The editors apologize for any violation that may have happened, and they will be happy to rectify any such material in later versions of this book.

Practical -1

Approximation Solution of R-k Method for first order PDE.

The most commonly used Runge-Kutta method to find the solution of a differential equation is the RK4 method, i.e., the fourth-order Runge-Kutta method. The Runge-Kutta method provides the approximate value of y for a given point x. Only the first order ODEs can be solved using the Runge Kutta RK4 method.

Runge-Kutta Fourth Order Method Formula

The formula for the fourth-order Runge-Kutta method is given by:

$y_1 = y_0 + (⅙) (k_1 + 2k_2 + 2k_3 + k_4)$
Here,
$k_1 = hf(x_0, y_0)$
$k_2 = hf[x_0 + (½)h, y_0 + (½)k_1]$
$k_3 = hf[x_0 + (½)h, y_0 + (½)k_2]$
$k_4 = hf(x_0 + h, y_0 + k_3)$
Example 1: Consider an ordinary differential equation $dy/dx = x^2 + y^2$, $y(1) = 1.2$. Find $y(1.05)$ using the fourth order Runge-Kutta method.
Solution:

Given,
$dy/dx = x^2 + y^2$, $y(1) = 1.2$

So, $f(x, y) = x^2 + y^2$

$x0 = 1$ and $y0 = 1.2$

Also, $h = 0.05$

Let us calculate the values of k_1, k_2, k_3 and k_4.

$k_1 = hf(x_0, y_0)$

$= (0.05) [x_0^2 + y_0^2]$

$= (0.05) [(1)^2 + (1.2)^2]$

$= (0.05) (1 + 1.44)$

$= (0.05)(2.44)$

$= 0.122$

$k_2 = hf[x_0 + (½)h, y_0 + (½)k_1]$

$= (0.05) [f(1 + 0.025, 1.2 + 0.061)]$ {since $h/2 = 0.05/2 = 0.025$ and $k_1/2 = 0.122/2 = 0.061$}

$= (0.05)\ [f(1.025, 1.261)]$

$= (0.05)\ [(1.025)^2 + (1.261)^2]$

$= (0.05)\ (1.051 + 1.590)$

$= (0.05)(2.641)$

$= 0.1320$

$k_3 = hf[x_0 + (½)h, y_0 + (½)k_2]$

$= (0.05)\ [f(1 + 0.025, 1.2 + 0.066)]$ {since $h/2 = 0.05/2 = 0.025$ and $k_2/2 = 0.132/2 = 0.066$}

$= (0.05)\ [f(1.025, 1.266)]$

$= (0.05)\ [(1.025)^2 + (1.266)^2]$

$= (0.05)\ (1.051 + 1.602)$

$= (0.05)(2.653)$

$= 0.1326$

$k_4 = hf(x_0 + h, y_0 + k_3)$

$= (0.05)\ [f(1 + 0.05, 1.2 + 0.1326)]$

$= (0.05)\ [f(1.05, 1.3326)]$

$= (0.05)\ [(1.05)^2 + (1.3326)^2]$

$= (0.05)\ (1.1025 + 1.7758)$

$= (0.05)(2.8783)$

$= 0.1439$

By RK4 method, we have;

$y_1 = y_0 + (⅙)\ (k_1 + 2k_2 + 2k_3 + k_4)$

$y_1 = y(1.05) = y_0 + (⅙)\ (k_1 + 2k_2 + 2k_3 + k_4)$

By substituting the values of y_0, k_1, k_2, k_3 and k_4, we get;

$y(1.05) = 1.2 + (⅙)\ [0.122 + 2(0.1320) + 2(0.1326) + 0.1439]$

$= 1.2 + (⅙)\ (0.122 + 0.264 + 0.2652 + 0.1439)$

$= 1.2 + (⅙)\ (0.7951)$

$= 1.2 + 0.1325$

$= 1.3325$

SCILAB Code

```
//Runge Kutta 4rth order

deff('g=f(x,y)','g=2*y+x')
x0=input("Enter initial value of x0: ")
y0=input("Enter the value of y0: ")
h=input("Enter value of h: ")
xn=input("Enter Final value of xn: ")
n=(xn-x0)/h
for i=1:n
   k1=h*f(x0,y0)
   k2=h*f(x0+(h/2),y0+(k1/2))
   k3=h*f(x0+(h/2),y0+(k2/2))
   k4=h*f(x0+h,y0+k3)
   y1=y0+(1/6)*(k1+2*k2+2*k3+k4)
   x0=x0+h
   disp([x0 y1])
   y0=y1
end
```

Output

```
-->exec('D:\Scilab prog by me\Runge Kutta fourth order.sce', -1)
Enter initial value of x0: 0
Enter the value of y0: 1
Enter value of h: 0.1
Enter Final value of xn: 0.3
```

```
   0.1    1.22675

   0.2    1.5147724

   0.3    1.8776331
```

Practical -2

Solution of Wave Equations.

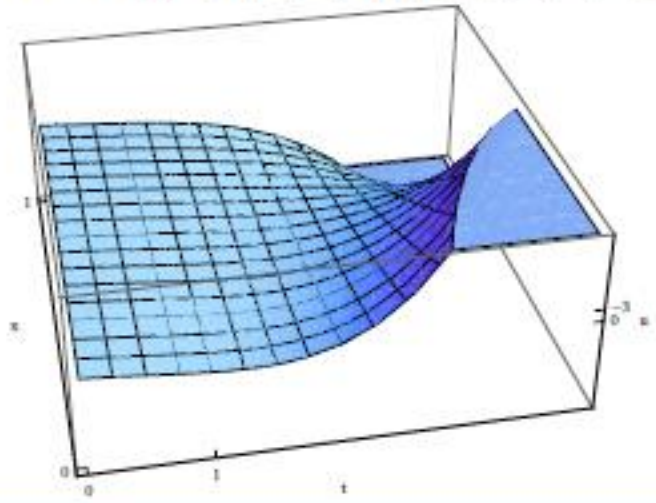
Problem-:1(a) $u_{tt} - u_{xx} = 0, \quad 0 < x < a, \quad t > 0,$
$u(x, 0) = \text{Log}\left(1 + x^2\right), \quad 0 \leq x \leq a,$
$u_t(x, 0) = 2, \quad 0 \leq x \leq a.$

```
eqn1a = {∂t,t u[x, t] - ∂x,x u[x, t] == 0,
   u[x, 0] == Log[1 + x^2], Derivative[0, 1][u][x, 0] == 2}
{u^(0,2)[x, t] - u^(2,0)[x, t] == 0,
 u[x, 0] == Log[1 + x^2], u^(0,1)[x, 0] == 2}
sol1a = u[x, t] /. NDSolve[eqn1a, u[x, t],
     {x, 0, 1}, {t, 0, 4}, PrecisionGoal → 3]  [[1]]
```

NDSolve::bcart :
Warning: an insufficient number of boundary conditions have been specified for the direction of independent variable x. Artificial boundary effects may be present in the solution. »

```
InterpolatingFunction[{{0., 1.}, {0., 4.}}, <>][x, t]
```

```
Plot3D[sol1a, {x, 0, 1}, {t, 0, 4}, AxesLabel → {"x", "t", "u"},
 Ticks → {{0, 1, 2, 3, 4}, {0, 1}, {-3, 0}}]
```



Problem-:1(b) $u_{tt} - u_{xx} = 0, \quad 0 < x < a, \quad t > 0,$
$u(x, 0) = \text{Sin}(x), \quad 0 \leq x \leq a,$
$u_t(x, 0) = \text{Cos}(x), \quad 0 \leq x \leq a.$

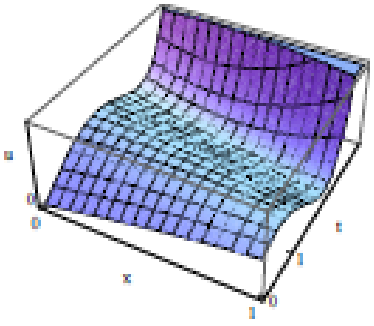
```
eqn1b = {∂t,t u[x, t] - ∂x,x u[x, t] == 0,
   u[x, 0] == Sin[x], Derivative[0, 1][u][x, 0] == Cos[x]}
sol1b = u[x, t] /. NDSolve[eqn1b, u[x, t],
     {x, 0, 1}, {t, 0, 4}, PrecisionGoal → 3]  [[1]]
Plot3D[sol1b, {x, 0, 1}, {t, 0, 4}, AxesLabel → {"x", "t", "u"},
 Ticks → {{0, 1, 2, 3, 4}, {0, 1}, {-3, 0}}]
```

$$\left\{u^{(0,2)}[x, t] - u^{(2,0)}[x, t] == 0,\ u[x, 0] == \text{Sin}[x],\ u^{(0,1)}[x, 0] == \text{Cos}[x]\right\}$$

NDSolve:bcart:

Warning: an insufficient number of boundary conditions have been specified for the direction of independent variable x. Artificial boundary effects may be present in the solution. >>

```
InterpolatingFunction[{{0., 1.}, {0., 4.}}, <>][x, t]
```



Practical -3

Euler Method

CONCEPT:-

(a)(Differential Equation - First Order) In many problems, the direct functional relation between the dependent variable y and the independent variable x is not known. However, the rate of change in y with respect to x is known and is given by a function f(x, y). The idea is to deduce the function y(x) from the rate equation

$$dy/dx = f(x, y(x))$$

(b)(Initial Value Problem) Any DE represent a family of curves/surfaces, however most problems pertains to picking out a particular curve which satisfies the given conditions. In nutshell, we seek a solution y(x) which satisfies the initial conditions

$$y(x0) = y0$$

CODE/METHOD

(a) (Euler Method - Forward) The derivative is approximated as $y' = \{y(x + h) - y(x)\}/h$

and the ODE is represented by

$$\{y(x + h) - y(x)\}/h = f(x, y(x)) \Rightarrow y(x + h) = y(x) + hf(x, y(x))$$

The domain (a, b) is populated with N equispaced nodal points to get (a, x1, ..., xN , b) representative points where $xi = a + ih$ and $h = (b - a)/(N + 1)$. The corresponding yi is evaluated using the above recursive relation to yield (y0, y1, ..., yN , yN+1)

APPLICATION :-

(a) (Radioactive Decay) The time t and the population N bears the relation $dN\ dt = -\lambda N$.

Note $f(t, N) = -N/\tau$ where $\tau = 1/\lambda$.

(b)(RC Circuit) The time t and the voltage V bears the relation $dV/dt = -V/\tau$ where $\tau = RC$. Note $f(t, V) = -V/\tau$.

(c)(Stokes' Law) The time t and the speed v bears the relation $mdv/dt = -6\pi\eta av$ where all variables have their usual meaning. Note $f(t, v) = -v/\tau$ where $\tau = m/6\pi\eta a$. Plot the time evolution of the respective solutions and highlight the main features of the solution.

Make and display the tabulated output with time

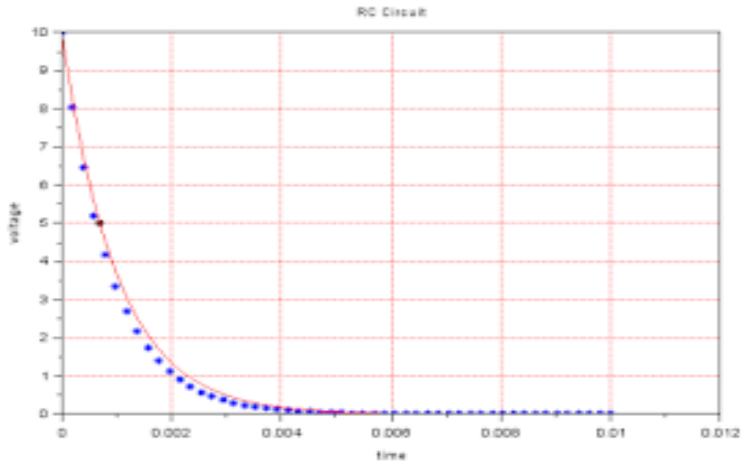
Note Initial Values will be Given In the Question.

SCILAB CODE FOR THE GIVEN PROBLEM

```
//***Euler MEthod****
/*****************************
clc, clear, clf
a=input("enter the value of left boundary condition for x of RC circuit= ")
b=input("enter the value of right boundary condition for xof RC circuit= ")
k=input("enter the value of left boundary condition for x of Radioactive
decay= ")
l=input("enter the value of right boundary condition for x of Radioactive
decay- ")
e=input("enter the value of left boundary condition for x of viscosity= ")
f=input("enter the value of right boundary condition for x of visxosity= ")
N=50
//RC Circuit
A= 5 //initial value of y in volts
Az=A/2 //half life
R= 1000 //value of R in ohms
C=0.000001 //value of C in Farads
u=1/(R*C) //u is a constant
```

```
hf1= log(0.5)/(-u)
h=(b-a)/(N+1) // calculating the value of step size for RC circuit
x(1)=a;y(1)=A; //initialising x and y
for i=1:N+1
m=i+1
y(m)= y(m-1)+h*(-u*y(i)) //forward difference formula
x(m)= x(m-1) + h
end
ex= A*exp(-u*x)
disp([x,y])
scf
plot(x,y,'*')
plot(hf1,Az,"*k")
plot(x,ex, 'r')
xlabel("time(sec))")
ylabel("voltage(v)")
set(gca(), 'grid',[1 1]*color('red'))
title ("RC Circuit")
```

note – With Limits 0 to 0.01 we will get the graph like this



Practical -4

Solution of Heat equation.

Problem-:1(a) $u_t - u_{xx} = 0, \quad 0 < x < 5, \quad t > 0,$

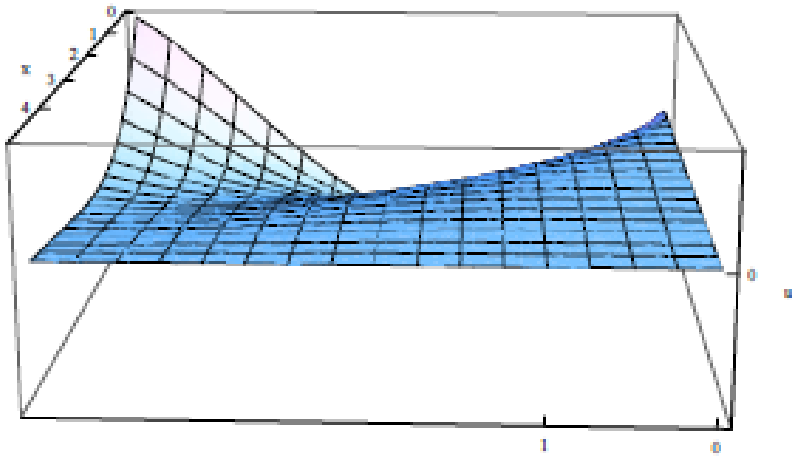
$u(x, 0) = 0, \quad 0 \leq x \leq 5,$

$u(0, t) = \sin(t) \quad t \geq 0$

$u(5, t) = 0, \quad t \geq 0.$

```
eqn1a = {∂t u[x, t] - ∂x,x u[x, t] == 0,
  u[x, 0] == 0, u[0, t] == Sin[t], u[5, t] == 0}
sol1a = u[x, t] /. NDSolve[eqn1a, u[x, t],
    {x, 0, 5}, {t, 0, 10}, PrecisionGoal → 3]  [[1]]
Plot3D[sol1a, {x, 0, 5}, {t, 0, 4}, AxesLabel → {"x", "t", "u"},
 Ticks → {{0, 1, 2, 3, 4}, {0, 1}, {-3, 0}}]
```

```
{u^(0,1)[x, t] - u^(2,0)[x, t] == 0, u[x, 0] == 0, u[0, t] == Sin[t], u[5, t] == 0}
InterpolatingFunction[{{0., 5.}, {0., 10.}}, <>][x, t]
```



Problem-:1(b) $u_t - u_{xx} = 0, \quad 0 < x < 20, \quad t > 0,$

$u(x, 0) = 0, \quad 0 \leq x \leq 20,$

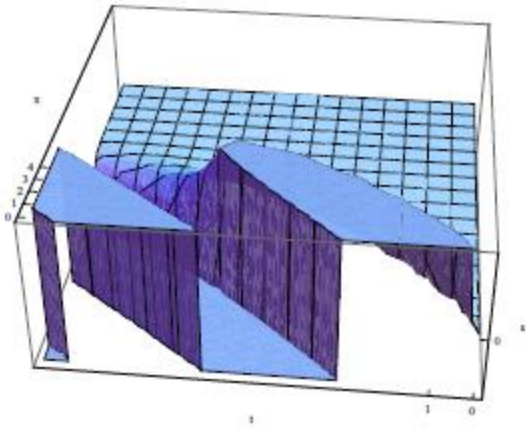
$u(0, t) = t^2 * \sin(t) \quad t \geq 0$

$u(20, t) = 0, \quad t \geq 0.$

```
eqn1b = {∂t u[x, t] - ∂x,x u[x, t] == 0,
   u[x, 0] == 0, u[0, t] == t^2 * Sin[t], u[20, t] == 0}
sol1b = u[x, t] /. NDSolve[eqn1b, u[x, t],
     {x, 0, 20}, {t, 0, 10}, PrecisionGoal → 3][[1]]
Plot3D[sol1b, {x, 0, 20}, {t, 0, 10}, AxesLabel →
   {"x", "t", "u"}, Ticks → {{0, 1, 2, 3, 4}, {0, 1}, {-3, 0}}]
```

$\{u^{(0,1)}[x, t] - u^{(2,0)}[x, t] == 0, u[x, 0] == 0, u[0, t] == t^2 \text{Sin}[t], u[20, t] == 0\}$

InterpolatingFunction[{{0., 20.}, {0., 10.}}, <>][x, t]



Problem-:1(c) $u_t - u_{xx} = 0, \quad 0 < x < 20, \quad t > 0,$

$u(x, 0) = 0, \quad 0 \leq x \leq 20,$

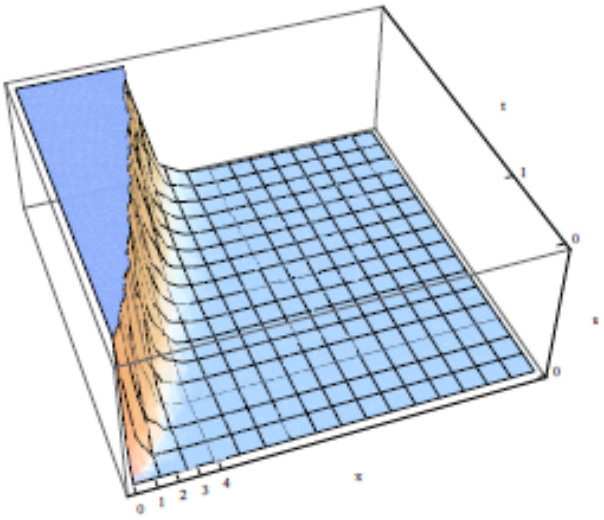
$u(0, t) = t^2 \quad t \geq 0$

$u(20, t) = 0, \quad t \geq 0.$

```
eqn1c = {∂t u[x, t] - ∂x,x u[x, t] == 0,
   u[x, 0] == 0, u[0, t] == t^2, u[20, t] == 0}
sol1c = u[x, t] /. NDSolve[eqn1c, u[x, t],
     {x, 0, 20}, {t, 0, 10}, PrecisionGoal → 3][[1]]
Plot3D[sol1c, {x, 0, 20}, {t, 0, 4}, AxesLabel →
   {"x", "t", "u"}, Ticks → {{0, 1, 2, 3, 4}, {0, 1}, {-3, 0}}]
```

$\left\{u^{(0,1)}[x, t] - u^{(2,0)}[x, t] == 0, u[x, 0] == 0, u[0, t] == t^2, u[20, t] == 0\right\}$

```
InterpolatingFunction[{{0., 20.}, {0., 10.}}, <>][x, t]
```



Reference book of practicals:

T1: Chetna Jain, Computing in Scilab, Cambridge university press, 3rd Edition, 2023.

T2: Sandeep Nagar, Introduction to Scilab: For Engineers and Scientists, Apress, 2017.

T3: Alain Vande Wouwer, Philippe Saucez, Carlos Vilas, Simulation of ODE/PDE Models with MATLAB®, OCTAVE and SCILAB, ·Springer 2014

T4: Jose Miguel David Baez-Lopez, David Alfredo Baez Villegas, MATLAB Handbook with Applications to Mathematics, Science, Engineering, and Finance, CRC Press, 2019