

## الحلول العددية للمعادلات التفاضلية الجزئية

### NUMERICAL SOLUTIONS TO PARTIAL DIFFERENTIAL EQUATIONS

(٩,١) مقدمة

#### Introduction

إن الصورة العامة للمعادلة التفاضلية الجزئية من الرتبة الثانية تكون كالتالي :

$$au_{xx} + bu_{xy} + cu_{yy} + du_x + eu_y + fu = g$$

حيث إن  $a, b, c, d, e, f, g$  تكون دوال في  $x, y$  ، وعادة ما تعطى شروط عن الدالة  $u(x, y)$  عند منحنيات معينة تحيط بنطاق الدالة ، وتسمى الشروط الحدية إذا كان كل من  $x, y$  هي الإحداثيات الفراغية ، وتسمى الشروط الابتدائية إذا احتوت على الزمن . ويتم تصنيف المعادلة إلى ثلاثة أنواع :

١- نوع ناقصي Elliptic : إذا كان  $b^2 - 4ac < 0$  ومثال على ذلك معادلة بواسون Poisson equation والتي لها الصورة التالية :

$$u_{xx} + u_{yy} = f(x, y)$$

٢- نوع مكافئ Elliptic : إذا كان  $b^2 - 4ac = 0$  ومثال على ذلك معادلة الحرارة heat equation والتي لها الصورة التالية :

$$u_t = u_{xx}$$

٣- نوع زائدي Hyperbolic : إذا كان  $b^2 - 4ac > 0$  ومثال على ذلك معادلة الموجة wave equation والتي لها الصورة التالية :

$$u_{tt} = u_{xx}$$

صور القيم الحدية المختلفة وشروطها

إذا كان المطلوب إيجاد قيم الدالة  $u(x, y)$  داخل النطاق  $D(x, y)$  والمنحنى المحيط بهذا النطاق يسمى  $\partial D(x, y)$  فإن هناك صورة مختلفة منها :

١- مسألة ديرشلت Dirichlet problem :

إذا كانت  $u(x, y)$  معطاة على المحيط  $\partial D(x, y)$ .

٢- مسألة نيومان Neumann problem :

إذا كانت  $\frac{\partial u}{\partial n} = f(x, y)$  معطاة عند المحيط  $\partial D(x, y)$  حيث إن  $n$  هو المتجه العمود على  $\partial D$ .

٣- مسألة مختلطة Mixed problem :

إذا كانت  $\frac{\partial u_p}{\partial n} = f(u_p)$  معطاة عند  $\partial D(x, y)$ .

(٩،٢) معادلات تفاضلية جزئية من النوع الناقصي

Elliptic Partial Differential Equations

نأخذ معادلة بواسون كمثال لهذا النوع :

$$u_{xx}(x, y) + u_{yy}(x, y) = f(x, y), \quad x, y \in D$$

$$u(x, y) = g(x, y), \quad x, y \in \partial D$$

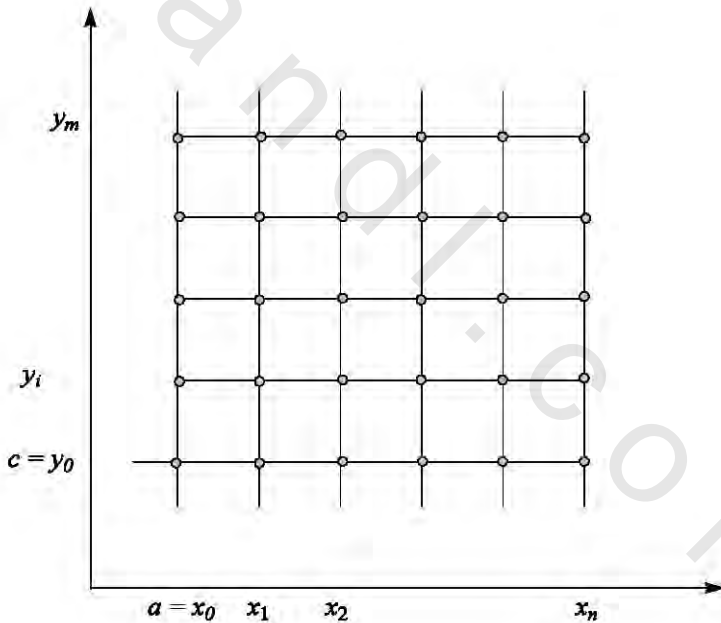
$$D = \{(x, y) : a \leq x \leq b, \quad c \leq y \leq d\}$$

حيث إن  $D$  يتم تقسيم المجال  $D$  إلى مستطيلات حيث إن :

$$h = \frac{b-a}{n}, \quad k = \frac{d-c}{m}, \quad m, n \in \mathbb{Z}^+$$

$$x_i = a + ih, \quad i = 0, 1, \dots, n$$

$$y_j = c + jh, \quad j = 0, 1, \dots, m$$



حيث إن نقاط التقاطع  $(x_i, y_j)$  تسمى نقاط الشبكة mesh points

: باستخدام مفكوك تايلور حول  $x_i$  نجد أن :  $j = 0, 1, \dots, m$  و  $i = 0, 1, 2, \dots, n$

$$\frac{\partial^2 u(x_i, y_j)}{\partial x^2} = \frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j))}{h^2} - \frac{h^2}{12} \frac{\partial^4 u(\xi_i, y_j)}{\partial x^4},$$

$$x_{i-1} \leq \xi_i \leq x_{i+1}$$

وحول  $y_j$  نجد أن:

$$\frac{\partial^2 u(x_i, y_j)}{\partial y^2} = \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1}))}{k^2} - \frac{k^2}{12} \frac{\partial^4 u(x_i, \eta_j)}{\partial y^4}$$

$$y_{i-1} \leq \mu_i \leq y_{i+1}$$

وبالتعويض في معادلة بواسون نحصل على التالي:

$$\frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j))}{h^2} + \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1}))}{k^2}$$

$$= f(x_i, y_j) + \frac{h^2}{12} \frac{\partial^4 u(\xi_i, y_j)}{\partial x^4} + \frac{k^2}{12} \frac{\partial^4 u(x_i, \eta_j)}{\partial y^4},$$

$$i = 1, 2, \dots, n-1, j = 1, 2, \dots, m-1$$

ومعالجة الشروط الحدية كما يلي:

$$u(x_0, y_j) = g(x_0, y_j), \quad j = 0, 1, \dots, m$$

$$u(x_n, y_j) = g(x_n, y_j), \quad j = 0, 1, \dots, m$$

$$u(x_i, y_0) = g(x_i, y_0), \quad i = 1, 2, \dots, n-1$$

$$u(x_i, y_m) = g(x_i, y_m), \quad i = 1, 2, \dots, n-1$$

وبذلك نحصل على طريقة الفرق المركزي Central difference method والتي لها خطأ الاقتطاع المحلي Local truncation error من الرتبة  $O(k^2 + h^2)$  على الصورة التالية :

$$2(r^2 + 1)w_{i,j} - (w_{i+1,j} + w_{i-1,j}) - r^2(w_{i,j+1} + w_{i,j-1}) = -h^2 f(x_i, y_j),$$

$$i = 1, 2, \dots, n-1, \quad j = 1, 2, \dots, m-1, \quad r = \frac{h}{k},$$

$$w_{0,j} = g(x_0, y_j), \quad j = 0, 1, \dots, m$$

$$w_{n,j} = g(x_n, y_j), \quad j = 0, 1, \dots, m$$

$$w_{i,0} = g(x_i, y_0), \quad i = 1, 2, \dots, n-1$$

$$w_{i,m} = g(x_i, y_m), \quad i = 1, 2, \dots, n-1$$

ويحل هذا النظام مع الشروط الحدية نحصل على  $w_{i,j} = u(x_i, y_j)$ .

مثال (٩،١)

بأخذ معادلة بواسون على الصورة التالية :

$$u_{xx} + u_{yy} = 0, \quad 0 \leq x \leq \frac{1}{2}, \quad 0 \leq y \leq \frac{1}{2}$$

مع الشروط الحدية :

$$u(0, y) = 0, \quad u(x, 0) = 0, \quad u(x, \frac{1}{2}) = 200x, \quad u(\frac{1}{2}, y) = 200y$$

وإذا أخذنا  $m = n = 4$  فسوف نحصل على التالي :

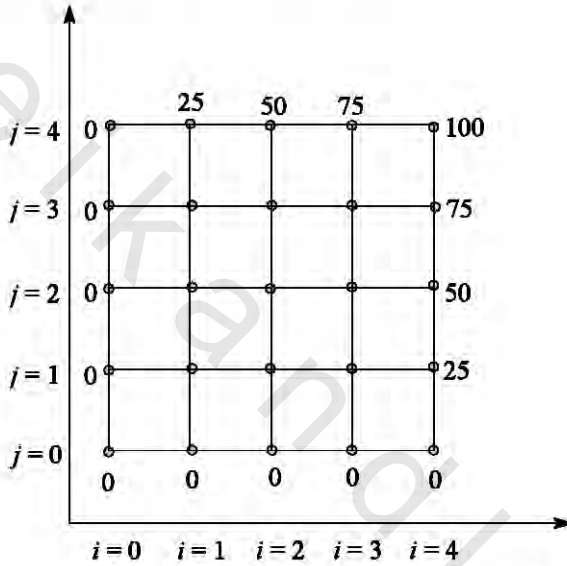
$$-w_{i,j} + 4w_{i,j} - w_{i-1,j} - w_{i,j-1} + w_{i,j+1} = 0$$

$$i = 1, 2, 3, \quad j = 1, 2, 3$$

ومن الشروط الحدية نجد أن :

$$w_{1,0} = w_{2,0} = w_{3,0} = 0, \quad w_{0,1} = w_{0,2} = w_{0,3} = 0,$$

$$w_{1,4} = w_{4,1} = 25, \quad w_{2,4} = w_{4,2} = 50, \quad w_{3,4} = w_{4,3} = 75,$$



ويكون النظام على الصورة التالية :

$$-w_{2,1} + 4w_{1,1} - w_{0,1} - w_{1,0} - w_{1,2} = 0,$$

$$i = 1, \quad j = 1$$

$$-w_{2,2} + 4w_{1,2} - w_{0,2} - w_{1,1} - w_{1,3} = 0,$$

$$i = 1, \quad j = 2$$

$$-w_{2,3} + 4w_{1,3} - w_{0,3} - w_{1,2} - w_{1,4} = 0,$$

$$i = 1, \quad j = 3$$

$$-w_{3,1} + 4w_{2,1} - w_{1,1} - w_{2,0} - w_{2,2} = 0,$$

$$i = 2, \quad j = 1$$

$$-w_{3,2} + 4w_{2,2} - w_{1,2} - w_{2,1} - w_{2,3} = 0,$$

$$i = 2, \quad j = 2$$

$$-w_{3,3} + 4w_{2,3} - w_{1,3} - w_{2,2} - w_{2,4} = 0,$$

$$i = 2, \quad j = 3$$

$$-w_{4,1} + 4w_{3,1} - w_{2,1} - w_{3,0} - w_{3,2} = 0,$$

$$i = 3, \quad j = 1$$

$$-w_{4,2} + 4w_{3,2} - w_{2,2} - w_{3,1} - w_{3,3} = 0,$$

$$i = 3, \quad j = 2$$

$$-w_{4,3} + 4w_{3,3} - w_{2,3} - w_{3,2} - w_{3,4} = 0,$$

$$i = 3, \quad j = 3$$

ويمكن كتابة هذا النظام على الصورة التالية :

$$\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} w_{1,1} \\ w_{1,2} \\ w_{1,3} \\ w_{2,1} \\ w_{2,2} \\ w_{2,3} \\ w_{3,1} \\ w_{3,2} \\ w_{3,3} \end{bmatrix} = \begin{bmatrix} 25 \\ 50 \\ 150 \\ 0 \\ 0 \\ 50 \\ 0 \\ 0 \\ 25 \end{bmatrix}$$

ويحل النظام الخطي السابق بأى طريقة مباشرة أو تكرارية نحصل على التالي :

$$\begin{aligned} w_{1,1} &= 18.75, & w_{1,2} &= 37.5, & w_{1,3} &= 56.25 \\ w_{2,1} &= 12.5, & w_{2,2} &= 25, & w_{2,3} &= 37.5 \\ w_{3,1} &= 6.25, & w_{3,2} &= 12.5, & w_{3,3} &= 18.75 \end{aligned}$$

### البرنامج رقم (٩.١)

```
C*****
**C                                POISSON FINITE-DIFFERENCE
C*****
**C  TO APPROXIMATE THE SOLUTION TO THE POISSON EQUATION
C  DEL(U) = F(X,Y), A <= X <= B, C <= Y <= D,
C  SUBJECT TO BOUNDARY CONDITIONS:
C      U(X,Y)=G(X,Y),
C  IF X = A OR X = B, FOR C <= Y <= D
C  IF Y = C OR Y = D, FOR A <= X <= B
C  INPUT:  ENDPOINTS A, B, C, D; INTEGERS m, n; TOLERANCE TOL;
C  MAXIMUM NUMBER OF ITERATIONS M.
```

```

C  OUTPUT: APPROXIMATIONS W(I,J) TO U(X(I),Y(J)) FOR EACH
C  I=1,...,n-1 AND J=1,...,m-1 OR A MESSAGE THAT THE
C  MAXIMUM NUMBER OF ITERATIONS WAS EXCEEDED.
C  INITIALIZATION
C  DIMENSION W(N-1,M-1),X(N-1),Y(M-1)
DIMENSION W(5,4),X(5),Y(4)
CHARACTER NAME1*30,AA*1
INTEGER OUP,FLAG,LBOUND
LOGICAL OK
C  BOUNDARY VALUES
F(XZ,YZ)= 0.0
G(XZ,YZ)= 400*XZ*YZ
OPEN(UNIT=5,FILE='CON',ACCESS='SEQUENTIAL')
OPEN(UNIT=6,FILE='CON',ACCESS='SEQUENTIAL')
WRITE(6,*) 'This is the Linear Finite Difference Method'
WRITE(6,*) 'for Elliptic Equations.'
WRITE(6,*) 'Have the functions F(x,y) and G(x,y) been created?'
WRITE(6,*) 'Enter Y or N '
WRITE(6,*) ''
READ(5,*) AA
IF(( AA .EQ. 'Y' ) .OR. ( AA .EQ. 'y' )) THEN
  OK = .FALSE.
19  IF (OK) GOTO 11
  WRITE(6,*) 'Input endpoints [A,B] on X-axis'
  WRITE(6,*) 'separated by blank.'
  WRITE(6,*) ''
  READ(5,*) A, B
  WRITE(6,*) 'Input endpoints [C,D] on X-axis'
  WRITE(6,*) 'separated by blank.'
  WRITE(6,*) ''
  READ(5,*) C, D
  IF ((A.GE.B).OR.(C.GE.D)) THEN
    WRITE(6,*) 'Left endpoint must be less'
    WRITE(6,*) 'than right endpoint'
  ELSE
    OK = .TRUE.
  ENDIF
  GOTO 19
11  OK = .FALSE.
12  IF (OK) GOTO 13
  WRITE(6,*) 'Input number of intervals n on the X-axis'
  WRITE(6,*) 'and m on the Y-axis separated by a blank.'
  WRITE(6,*) 'Note: both n and m should be larger than 2.'
  WRITE(6,*) ''

```

```

READ(5,*) N, M
IF ((N.LE.2).OR.(M.LE.2)) THEN
WRITE(6,*) 'Must be integers exceeding 2.'
ELSE
OK = .TRUE.
ENDIF
GOTO 12
13 OK = .FALSE.
14 IF (OK) GOTO 15
WRITE(6,*) 'Input tolerance '
WRITE(6,*) ''
READ(5,*) TOL
IF (TOL.LE.0.0) THEN
WRITE(6,*) 'Tolerance must be positive '
ELSE
OK = .TRUE.
ENDIF
GOTO 14
15 OK = .FALSE.
16 IF(OK) GOTO 17
WRITE(6,*) 'Input the maximum number of iterations.'
WRITE(6,*) ''
READ(5,*) LBOUND
IF(LBOUND.LE.0) THEN
WRITE(6,*) 'Must be a positive integer.'
ELSE
OK = .TRUE.
ENDIF
GO TO 16
17 CONTINUE
ELSE
WRITE(6,*) 'The program will end so that the functions'
WRITE(6,*) 'F and G can be created.'
OK = .FALSE.
ENDIF
IF(.NOT.OK) GOTO 400
WRITE(6,*) 'Select output destination: '
WRITE(6,*) '1. Screen '
WRITE(6,*) '2. Text file '
WRITE(6,*) 'Enter 1 or 2 '
WRITE(6,*) ''
READ(5,*) FLAG
IF ( FLAG .EQ. 2 ) THEN
WRITE(6,*) 'Input the file name in the form - '

```

```

WRITE(6,*) 'drive:name.ext'
WRITE(6,*) 'with the name contained within quotes'
WRITE(6,*) 'as example: "A:OUTPUT.DTA"'
WRITE(6,*) ''
READ(5,*) NAME1
OUP = 3
OPEN(UNIT=OUP,FILE=NAME1,STATUS='NEW')
ELSE
OUP = 6
ENDIF
WRITE(OUP,*) 'POISSON EQUATION FINITE-DIFFERENCE METHOD'
WRITE(OUP,*) 'A= ',A,' B= ',B
WRITE(OUP,*) 'C= ',C,' D= ',D
WRITE(OUP,*) 'TOL= ',TOL,' M= ',M,' N= ',N
WRITE(OUP,*) 'MAX. # ITERATIONS = ',LBOUND
MM=M-1
MMM=M-2
NN=N-1
NNN=N-2
C  XK WILL BE USED FOR K
C  STEP 1
H=(B-A)/N
XK=(D-C)/M
C  A,B,C,D WILL BE USED FOR X(0),X(N),Y(0),Y(M)
C  STEPS 2, 3 CONSTRUCT MESH POINTS
C  STEP 2
DO 10 I=1,NN
10 X(I)=A+I*H
C  STEP 3
DO 20 J=1,MM
20 Y(J)=C+J*XK
C  STEP 4
DO 30 I=1,NN
DO 30 J=1,MM
30 W(I,J)=0
C  STEP 5
C  USE V FOR LAMBDA, VV FOR MU
V=H*H/(XK*XK)
VV=2*(1+V)
L=1
CZ IS A NEW VALUE OF W(I,J) TO BE USED IN COMPUTING THE
ERROR
C  USE E INSTEAD OF NORM
C  STEP 6

```

```

100 IF (L.GT.LBOUND) GOTO 200
C   STEPS 7 THROUGH 20 PERFORM GAUSS-SEIDEL ITERATIONS
C   STEP 7
    Z=(-H*H*F(X(1),Y(MM))+G(A,Y(MM))+V*G(X(1),D)+W(1,MM-1)*V+W(2,MM))/VV
    E=ABS(W(1,MM)-Z)
    W(1,MM)=Z
C   STEP 8
    DO 40 I=2,NNN
    Z=(-H*H*F(X(I),Y(MM))+V*G(X(I),D)+W(I-1,MM)+W(I+1,MM)+V*
*   W(I,MM-1))/VV
    IF(ABS(W(I,MM)-Z).GT.E) E=ABS(W(I,MM)-Z)
40  W(I,MM)=Z
C   STEP 9
    Z=(-H*H*F(X(NN),Y(MM))+G(B,Y(MM))+V*G(X(NN),D)+W(NN-1,MM)+V*
*   W(NN,MM-1))/VV
    IF(ABS(W(NN,MM)-Z).GT.E) E=ABS(W(NN,MM)-Z)
    W(NN,MM)=Z
C   STEP 10
    DO 50 LL=2,MMM
    J=MMM-LL+2
C   STEP 11
    Z=(-H*H*F(X(1),Y(J))+G(A,Y(J))+V*W(1,J+1)+V*W(1,J-
1)+W(2,J))/VV
    IF(ABS(W(1,J)-Z).GT.E) E=ABS(W(1,J)-Z)
    W(1,J)=Z
C   STEP 12
    DO 60 I=2,NNN
    Z=(-H*H*F(X(I),Y(J))+W(I-1,J)+V*W(I,J+1)+V*W(I,J-
1)+W(I+1,J))/VV
    IF(ABS(W(I,J)-Z).GT.E) E=ABS(W(I,J)-Z)
60  W(I,J)=Z
C   STEP 13
    Z=(-H*H*F(X(NN),Y(J))+G(B,Y(J))+W(NN-1,J)+V*W(NN,J+1)+V
*   *W(NN,J-1))/VV
    IF(ABS(W(NN,J)-Z).GT.E) E=ABS(W(NN,J)-Z)
50  W(NN,J)=Z
C   STEP 14
    Z=(-
H*H*F(X(1),Y(1))+V*G(X(1),C)+G(A,Y(1))+V*W(1,2)+W(2,1))/VV
    IF(ABS(W(1,1)-Z).GT.E) E=ABS(W(1,1)-Z)
    W(1,1)=Z
C   STEP 15
    DO 70 I=2,NNN
    Z=(-H*H*F(X(I),Y(1))+V*G(X(I),C)+W(I+1,1)+W(I-1,1)+V*W(I,2))/VV

```

```

IF(ABS(W(I,1)-Z).GT.E) E=ABS(W(I,1)-Z)
70  W(I,1)=Z
C    STEP 16
Z=(-H*H*F(X(NN),Y(1))+V*G(X(NN),C)+G(B,Y(1))+W(NN-1,1)+V*W(NN,2))/VV
IF(ABS(W(NN,1)-Z).GT.E) E=ABS(W(NN,1)-Z)
W(NN,1)=Z
C    STEP 17
IF(E.LE.TOL) THEN
WRITE(OUT,2) L
C    STEP 18
DO 80 I=1,NN
DO 80 J=1,MM
80  WRITE(OUT,3) I,J,X(I),Y(J),W(I,J)
C    STEP 19
GOTO 400
END IF
C    STEP 20
L=L+1
GOTO 100
C    STEP 21
200  WRITE(OUT,1) LBOUND
400  CLOSE(UNIT=5)
CLOSE(UNIT=OUT)
IF(OUT.NE.6) CLOSE(UNIT=6)
STOP
1    FORMAT(1X,'METHOD FAILS AFTER ITERATION NO. ',I4)
2    FORMAT(1X,'NO. OF ITERATIONS IS',I4,3X,'ORDER OF OUTPUT IS
*I,J,X(I),Y(J),W(I,J)')
3    FORMAT(1X,2(I3,2X),3(E15.8,3X))
END

```

## تمارين (٩، ١)

باستخدام طريقة الفروق المنتهية حل معادلة بواسون على الصورة التالية :

$$u_{xx} + u_{yy} = xe^y, \quad 0 \leq x \leq 2, \quad 0 \leq y \leq 1$$

مع الشروط الحدية :

$$u(0, y) = 0, \quad u(2, y) = 2e^y,$$

$$u(x, 0) = x, \quad u(x, 1) = xe$$

وبأخذ  $n=6, m=5$ .

### (٩.٣) معادلات تفاضلية جزئية من النوع المكافئ

#### Parabolic Partial Differential Equations

نأخذ من هذا النمط معادلة الحرارة heat equation والتي لها الصورة:

$$u_t = \alpha^2 u_{xx}, \quad 0 < x < \ell, \quad t > 0$$

$$u(0, t) = u(\ell, t) = 0, \quad t > 0 \quad \text{بالشروط الحدية:}$$

$$u(x, 0) = f(x), \quad 0 \leq x \leq \ell \quad \text{والشروط الابتدائية}$$

باستخدام مفكوك تايلور نحصل على التالي:

$$\frac{\partial u(x_i, t_j)}{\partial t} = \frac{u(x_i, t_{j+1}) - u(x_i, t_j)}{k} - \frac{k}{2} u_{tt}(x_i, \mu_j), \quad \mu_j \in (t_j, t_{j+1})$$

وكذلك:

$$u_{xx}(x_i, t_j) = \frac{u(x_{i+1}, t_j) - 2u(x_i, t_j) + u(x_{i-1}, t_j))}{h^2} - \frac{h^2}{12} u_{4x}(\xi_i, t_j), \quad \xi_i \in (x_{i-1}, x_{i+1})$$

وبالتعويض في المعادلة نحصل على معادلة الفروق:

$$\frac{w_{i,j+1} - w_{i,j}}{k} - \alpha^2 \frac{w_{i+1,j} - 2w_{i,j} + w_{i-1,j}}{h^2} = 0, \quad i = 1, 2, \dots, m-1, \quad j = 1, 2, \dots$$

$$\text{وخطأ الاقتطاع لها هو } \tau_{i,j} = \frac{k}{2} u_{tt}(x_i, \mu_j) - \frac{\alpha^2 h^2}{12} u_{4x}(\xi_i, t_j) \text{ وتسمى}$$

بطريقة الفروق الأمامية Forward difference method ويمكن صياغتها على الصورة

التالية:

$$w_{i,i+1} = (1 - \frac{2\alpha^2 k}{h^2})w_{i,j} + \frac{\alpha^2 k}{h^2}(w_{i+1,j} + w_{i-1,j}), \quad i = 1, 2, \dots, m-1, \quad j = 1, 2, \dots$$

$$w_{0,i} = w_{m,j} = 0, \quad w_{i,0} = f(x_i)$$

ويمكن وضع النظام على الصورة التالية :

$$w^{(j)} = Aw^{(j-1)}, \quad j = 1, 2, \dots$$

حيث إن :

$$A = \begin{pmatrix} 1-2\lambda & \lambda & 0 & & \\ \lambda & 1-2\lambda & \lambda & & \\ & & \lambda & 1-2\lambda & \lambda \\ & & 0 & \lambda & 1-2\lambda \end{pmatrix}, \quad \lambda = \frac{\alpha^2 k}{h^2}$$

وهذه الطريقة من الرتبة  $O(k + h^2)$ . ومن أجل دراسة استقرار طريقة الفروق الأمامية لابد أن يكون :

$$\rho(A) \leq 1$$

وبحساب القيم الذاتية للمصفوفة نجد أن :

$$\rho(A) = \max_{1 \leq i \leq m-1} \left| 1 - 4\lambda \left( \sin\left(\frac{i\pi}{2m}\right) \right)^2 \right| \leq 1$$

ومنها نحصل على شرط الاستقرار التالي :

$$\alpha^2 \frac{k}{h^2} \leq \frac{1}{2}$$

مثال (٩.٢)

أوجد حل معادلة الحرارة التي لها الصورة التالية :

$$u_t - u_{xx} = 0, \quad 0 < x < 1, \quad t \geq 0$$

$$u(0, t) = u(1, t) = 0, \quad t \geq 0$$

$$u(x, 0) = \sin \pi x, \quad 0 \leq x \leq 1$$

والحل الحقيقي هو :  $u(x, t) = e^{-\pi^2 t} \sin \pi x$ بوضع  $h = 0.1$ ,  $k = 0.0005$  واستخدام معادلات الفروق السابقة سوف

نحصل على النتائج المسجلة في الجدول رقم (٩.١).

الجدول رقم (٩.١).

| $x$ | $u(x, 0.5)$ | $w_{i,1000}$ | $ u(x, 0.5) - w_{i,1000} $ |
|-----|-------------|--------------|----------------------------|
| 0.0 | 0           | 0            | 0                          |
| 0.2 | 0.00422728  | 0.00434922   | 1.219E-4                   |
| 0.4 | 0.00383989  | 0.00703719   | 1.973E-4                   |
| 0.6 | 0.00383989  | 0.00703719   | 1.973E-4                   |
| 0.8 | 0.00422728  | 0.00434922   | 1.219E-4                   |
| 1.0 | 0           | 0            | 0                          |

البرنامج رقم (٩.٢)

```

C*****
C  HEAT EQUATION BACKWARD-DIFFERENCE
C*****
C  TO APPROXIMATE THE SOLUTION TO THE PARABOLIC PARTIAL-DIFFERENTIAL
C  EQUATION SUBJECT TO THE BOUNDARY CONDITIONS
C    U(0,T) = U(L,T) = 0, 0 < t < T = MAX t
C  AND THE INITIAL CONDITIONS
C    U(X,0) = F(X), 0 <= X <= L:
C  INPUT:  ENDPOINT L; MAXIMUM TIME T; CONSTANT ALPHA;
c INTEGERS
C  OUTPUT: APPROXIMATIONS W(I,J) TO U(X(I),T(J)) FOR EACH

```

```

C      I=1,...,M-1 AND J=1,...,N.
C  INITIALIZATION
C  DIMENSION W(M), XL(M-1), XU(M-1), Z(M-1)
C  DIMENSION W(10),XL( 9),XU( 9),Z( 9)
C  FT IS CAPITAL T AND FX IS L
C  CHARACTER NAME1*30,AA*1
C  INTEGER OUP,FLAG
C  LOGICAL OK
C  F(XZ)=SIN(PI*XZ)
C  PI=4*ATAN(1.0)
C  OPEN(UNIT=5,FILE='CON',ACCESS='SEQUENTIAL')
C  OPEN(UNIT=6,FILE='CON',ACCESS='SEQUENTIAL')
C  WRITE(6,*) 'This is the Backward-Difference Method'
C  WRITE(6,*) 'for the Heat Equation.'
C  WRITE(6,*) 'Has the function F been created in the program? '
C  WRITE(6,*) 'Enter Y or N '
C  WRITE(6,*) ''
C  READ(5,*) AA
C  IF(( AA .EQ. 'Y' ) .OR. ( AA .EQ. 'y' )) THEN
C  OK = .FALSE.
C  WRITE(6,*) 'The lefthand endpoint on the X-axis is 0.'
19  IF (OK) GOTO 11
C  WRITE(6,*) 'Input righthand endpoint on the X-axis.'
C  WRITE(6,*) ''
C  READ(5,*) FX
C  IF (FX.LE.0.0) THEN
C  WRITE(6,*) 'Must be a positive number.'
C  ELSE
C  OK = .TRUE.
C  ENDIF
C  GOTO 19
11  OK = .FALSE.
12  IF (OK) GOTO 13
C  WRITE(6,*) 'Input the maximum value of the time'
C  WRITE(6,*) 'variable T.'
C  WRITE(6,*) ''
C  READ(5,*) FT
C  IF ( FT .LE. 0.0 ) THEN
C  WRITE(6,*) 'Must be positive number.'
C  ELSE
C  OK = .TRUE.
C  ENDIF
C  GOTO 12
13  WRITE(6,*) 'Input the constant alpha.'

```

```

WRITE(6,*) ''
READ(5,*) ALPHA
15 OK = .FALSE.
14 IF (OK) GOTO 16
WRITE(6,*) 'Input integer m = number of intervals'
WRITE(6,*) 'on X-axis and N = number of time intervals.'
WRITE(6,*) 'Note: m must be 3 or larger.'
WRITE(6,*) 'Separate by blank. '
READ(5,*) M,N
IF ((M.LE.2).OR.(N.LE.0)) THEN
WRITE(6,*) 'Numbers not within correct range.'
ELSE
OK = .TRUE.
ENDIF
GOTO 14
16 CONTINUE
ELSE
WRITE(6,*) 'The program will end so that the function F '
WRITE(6,*) 'can be created '
OK = .FALSE.
ENDIF
IF(.NOT.OK) GOTO 400
WRITE(6,*) 'Select output destination: '
WRITE(6,*) '1. Screen '
WRITE(6,*) '2. Text file '
WRITE(6,*) 'Enter 1 or 2 '
WRITE(6,*) ''
READ(5,*) FLAG
IF ( FLAG.EQ. 2 ) THEN
WRITE(6,*) 'Input the file name in the form - '
WRITE(6,*) 'drive:name.ext'
WRITE(6,*) 'with the name contained within quotes'
WRITE(6,*) 'as example: "A:OUTPUT.DTA"'
WRITE(6,*) ''
READ(5,*) NAME1
OUP = 3
OPEN(UNIT=OUP,FILE=NAME1,STATUS='NEW')
ELSE
OUP = 6
ENDIF
WRITE(OUP,*) 'BACKWARD-DIFFERENCE METHOD FOR HEAT EQUATION'
MM=M-1
MMM=MM-1
NN=N-1

```

```

C  STEP 1
  H=FX/M
C  USE TK FOR K
  TK=FT/N
C  USE VV FOR LAMBDA
  VV=ALPHA*ALPHA*TK/(H*H)
C  STEP 2
  DO 10 I=1,MM
C  INITIAL VALUES
10  W(I)=F(I*H)
C  STEP 3
C  STEPS 3 THROUGH 11 SOLVE A TRIDIAGONAL LINEAR SYSTEM
C  USING ALGORITHM 6.7
C  USE XL FOR L, XU FOR U
  XL(1)=1+2*VV
  XU(1)=-VV/XL(1)
C  STEP 4
  DO 20 I=2,MMM
  XL(I)=1+2*VV+VV*XU(I-1)
20  XU(I)=-VV/XL(I)
C  STEP 5
  XL(MM)=1+2*VV+VV*XU(MMM)
C  STEP 6
  DO 30 J=1,N
C  STEP 7
C  CURRENT T(J)
  T=J*TK
  Z(1)=W(1)/XL(1)
C  STEP 8
  DO 40 I=2,MM
40  Z(I)=(W(I)+VV*Z(I-1))/XL(I)
C  STEP 9
  W(MM)=Z(MM)
C  STEP 10
  DO 50 II=1,MMM
  I=MMM-II+1
50  W(I)=Z(I) -XU(I)*W(I+1)
30  CONTINUE
C  STEP 11 IS CHANGED TO OUTPUT ONLY THE FINAL TIME VALUES
C  OUTPUT FINAL TIME VALUE
  WRITE(OUT,1) FT
  WRITE(OUT,3)
  DO 60 I=1,MM
  X=I*H

```

```

60  WRITE(OUT,2) I,X,W(I)
C    STEP 12
400  CLOSE(UNIT=5)
      CLOSE(UNIT=OUT)
      IF(OUT.NE.6) CLOSE(UNIT=6)
      STOP
1    FORMAT(1X,'FINAL TIME VALUE IS',1X,E15.8)
2    FORMAT(1X,I3,2(1X,E15.8))
3    FORMAT(3X,T,12X,T(I),12X,'W(I)')
      END
    
```

ولدراسة استقرار stability معادلات الفروق المنتهية نعوض عن

$$u_{i,j} = \varepsilon^j e^{ikx_i} \text{ ونحصل على التالي:}$$

$$\varepsilon^{j+1} = \varepsilon^j [(1 - 2\lambda) + \lambda(e^{ikh} + e^{-ikh})]$$

$$\text{وحيث إن } \varepsilon^{j+1} = g \varepsilon^j \text{ إذن:}$$

$$g = 1 - 2\lambda(1 - \cos kh)$$

وشرط التقارب هو:  $|g| \leq 1$

$$-1 \leq 1 - 2\lambda(1 - \cos kh) \leq 1$$

$$0 \leq \lambda \leq \frac{1}{2}$$

$$\frac{k}{h^2} \leq \frac{1}{2} \quad \text{أي أن:}$$

وهو شرط الاستقرار لحل نظام معادلات الفروق المنتهية، ويقال أن هذا النظام مستقر

شرطياً Conditionally stable. ولايجاد نظام مستقر دون شرط Unconditionally stable

فإننا نستخدم طريقة الفروق الخلفية بالنسبة للزمن أي أن:

$$\frac{\partial u(x_i, t_j)}{\partial t} = \frac{u(x_i, t_j) - u(x_i, t_{j-1})}{k} + \frac{k}{2} u_{xx}(x_i, \mu_j), \mu_j \in (t_{j-1}, t_j)$$

ونحصل على معاملات الفروق المنتهية :

$$(1+2\lambda)w_{i,j} - \lambda w_{i+1,j} - \lambda w_{i-1,j} = w_{i,j-1},$$

$$i = 1, 2, \dots, m-1, \quad j = 1, 2, \dots$$

$$w_{i,0} = f(x_i), \quad w_{i,0} = w_{m,j} = 0 \quad \text{حيث إن :}$$

نجد أن هذا النظام مستقر استقراراً غير مشروط. أي أن  $|g| \leq 1$  متحقق وهذه الطريقة من الرتبة  $O(k+h^2)$  ونتائج المثال السابق مدونة في الجدول رقم (٩,٢).

الجدول رقم (٩,٢).

| $x$ | $u(x_0, 0.5)$ | $w_{i,50}$ | $ u(x_0, 0.5) - w_{i,50} $ |
|-----|---------------|------------|----------------------------|
| 0.0 | 0             | 0          | 0                          |
| 0.2 | 0.00422728    | 0.00551236 | 1.285E-3                   |
| 0.4 | 0.00683989    | 0.00891918 | 2.079E-3                   |
| 0.6 | 0.00683989    | 0.00891918 | 2.079E-3                   |
| 0.8 | 0.00422728    | 0.00551236 | 1.285E-3                   |
| 1.0 | 0             | 0          | 0                          |

ونلاحظ أن النتائج مأخوذة عند  $k = 0.01$ ,  $h = 0.1$ . ونذكر الآن طريقة كرانك نيكلسون Crank-Nicolson ومن أهم خواص هذه الطريقة أنها من الرتبة  $O(k^2 + h^2)$  وفيها يتم استبدال التفاضلات كما يلي :

$$(u_t)_{i,j} = \frac{u_{i,j+1} - u_{i,j}}{k}$$

$$(u_{xx})_{i,j} = \frac{1}{2}[(u_{xx})_{i,j+1} + (u_{xx})_{i,j}]$$

وتصبح المعادلة على الصورة التالية :

$$Aw^{(j+1)} = Bw^{(j)}, \quad j = 1, 2, \dots$$

حيث إن كلاً من A, B مصفوفة قطرية و  $\lambda = \frac{\alpha^2 k}{h^2}$

$$A = \begin{bmatrix} 1+2\lambda & -\frac{\lambda}{2} & 0 & 0 \\ -\frac{\lambda}{2} & 1+2\lambda & 0 & 0 \\ 0 & 0 & 1+2\lambda & -\frac{\lambda}{2} \\ 0 & 0 & -\frac{\lambda}{2} & 1+2\lambda \end{bmatrix}, \quad B = \begin{bmatrix} 1-2\lambda & \frac{\lambda}{2} & 0 & 0 \\ \frac{\lambda}{2} & 1-2\lambda & 0 & 0 \\ 0 & 0 & 1-2\lambda & \frac{\lambda}{2} \\ 0 & 0 & \frac{\lambda}{2} & 1-2\lambda \end{bmatrix}$$

وهذه الطريقة غير مشروطة الاستقرار ونتائج طريقة كرانك نيكلسون معروضة في الجدول رقم (٩,٣).

الجدول رقم (٩,٣).

| $x$ | $u(x_0, 0.5)$ | $w_{i,50}$ | $ u(x_0, 0.5) - w_{i,50} $ |
|-----|---------------|------------|----------------------------|
| 0.0 | 0             | 0          | 0                          |
| 0.2 | 0.00422728    | 0.00438461 | 1.573E-4                   |
| 0.4 | 0.00683989    | 0.00709444 | 2.546E-4                   |
| 0.6 | 0.00683989    | 0.00709444 | 2.546E-4                   |
| 0.8 | 0.00422728    | 0.00438461 | 1.573E-4                   |
| 1.0 | 0             | 0          | 0                          |

البرنامج رقم (٩.٣)

```

C*****
C      CRANK-NICOLSON
C*****
C      TO APPROXIMATE THE SOLUTION TO THE PARABOLIC PARTIAL-
CDIFFERENTIAL
C      EQUATION SUBJECT TO THE BOUNDARY CONDITIONS
C       $U(0,T) = U(L,T) = 0, 0 < t < T = \text{MAX } t$ 
C      AND THE INITIAL CONDITIONS
C       $U(X,0) = F(X), 0 \leq X \leq L:$ 
C      INPUT:  ENDPPOINT L; MAXIMUM TIME T; CONSTANT ALPHA;
cINTEGERS
C      M,N:
C      OUTPUT: APPROXIMATIONS W(I,J) TO U(X(I),T(J)) FOR EACH
C      I=1,...,M-1 AND J=1,...,N.
C      INITIALIZATION
C      DIMENSION V(M),XL(M-1),XU(M-1),Z(M-1)
DIMENSION V( 10),XL( 9),XU( 9),Z( 9)
C      V IS USED FOR W, FT FOR CAPITAL T, FX FOR L
      CHARACTER NAME1*30,AA*1
      INTEGER OUP,FLAG
      LOGICAL OK
      F(XZ)=SIN(PI*XZ)
      PI=4*ATAN(1.0)
      OPEN(UNIT=5,FILE='CON',ACCESS='SEQUENTIAL')
      OPEN(UNIT=6,FILE='CON',ACCESS='SEQUENTIAL')
      WRITE(6,*) 'This is the Crank-Nicolson Method'
      WRITE(6,*) 'for the Heat Equation.'
      WRITE(6,*) 'Has the function F been created in the program? '
      WRITE(6,*) 'Enter Y or N '
      WRITE(6,*) ''
      READ(5,*) AA
      IF(( AA .EQ. 'Y') .OR. ( AA .EQ. 'y' )) THEN
      OK = .FALSE.
      WRITE(6,*) 'The lefthand endpoint on the X-axis is 0.'
19  IF (OK) GOTO 11
      WRITE(6,*) 'Input righthand endpoint on the X-axis.'
      WRITE(6,*) ''
      READ(5,*) FX
      IF (FX.IE.0.0) THEN
      WRITE(6,*) 'Must be a positive number.'
      ELSE
      OK = .TRUE.

```

```

ENDIF
GOTO 19
11 OK = .FALSE.
12 IF (OK) GOTO 13
WRITE(6,*) 'Input the maximum value of the time'
WRITE(6,*) 'variable T.'
WRITE(6,*) ''
READ(5,*) FT
IF ( FT .LE. 0.0 ) THEN
WRITE(6,*) 'Must be positive number.'
ELSE
OK = .TRUE.
ENDIF
GOTO 12
13 WRITE(6,*) 'Input the constant alpha.'
WRITE(6,*) ''
READ(5,*) ALPHA
15 OK = .FALSE.
14 IF (OK) GOTO 16
WRITE(6,*) 'Input integer m = number of intervals'
WRITE(6,*) 'on X-axis and N = number of time intervals.'
WRITE(6,*) 'Note: m must be 3 or larger.'
WRITE(6,*) 'Separate by blank. '
READ(5,*) M,N
IF ((M.LE.2).OR.(N.LE.0)) THEN
WRITE(6,*) 'Numbers not within correct range.'
ELSE
OK = .TRUE.
ENDIF
GOTO 14
16 CONTINUE
ELSE
WRITE(6,*) 'The program will end so that the function F '
WRITE(6,*) 'can be created '
OK = .FALSE.
ENDIF
IF(.NOT.OK) GOTO 400
WRITE(6,*) 'Select output destination: '
WRITE(6,*) '1. Screen '
WRITE(6,*) '2. Text file '
WRITE(6,*) 'Enter 1 or 2 '
WRITE(6,*) ''
READ(5,*) FLAG
IF ( FLAG .EQ. 2 ) THEN

```

```

WRITE(6,*) 'Input the file name in the form - '
WRITE(6,*) 'drive:name.ext'
WRITE(6,*) 'with the name contained within quotes'
WRITE(6,*) 'as example: "A:OUTPUT.DTA"'
WRITE(6,*) ''
READ(5,*) NAME1
OUP = 3
OPEN(UNIT=OUP,FILE=NAME1,STATUS='NEW')
ELSE
OUP = 6
ENDIF
WRITE(OUP,*) 'CRANK-NICOLSON METHOD FOR HEAT EQUATION'
MM=M-1
MMM=MM-1
NN=N-1
C STEP 1
H=FX/M
C TK IS USED FOR K
TK=FT/N
C VV IS USED FOR LAMBDA
VV=ALPHA*ALPHA*TK/(H*H)
C SET V(M)=0
V(M)=0
C STEP2
C INITIAL VALUES
DO 10 I=1,MM
10 V(I)=F(I*H)
C STEP 3
C STEPS 3 THROUGH 11 SOLVE A TRIDIAGONAL LINEAR SYSTEM
C USING ALGORITHM 6.7
C USE XL FOR L, XU FOR U
XL(1)=1+VV
XU(1)=-VV/(2*XL(1))
C STEP 4
DO 20 I=2,MMM
XL(I)=1+VV+VV*XU(I-1)/2
20 XU(I)=-VV/(2*XL(I))
C STEP 5
XL(MM)=1+VV+VV*XU(MMM)/2
C STEP 6
DO 30 J=1,N
C STEP 7
C CURRENT T(J)
T=J*TK

```

```

      Z(1)=((1-VV)*V(1)+VV*V(2)/2)/XL(1)
C     STEP 8
      DO 40 I=2,MM
40    Z(I)=((1-VV)*V(I)+VV/2*(V(I+1)+V(I-1)+Z(I-1)))/XL(I)
C     STEP 9
      V(MM)=Z(MM)
C     STEP 10
      DO 50 II=1,MMM
      I=MMM-II+1
50    V(I)=Z(I)-XU(I)*V(I+1)
30    CONTINUE
C     STEP 11--OUTPUT WILL BE ONLY FOR T=FT
      WRITE(OUT,1) FT
      WRITE(OUT,3)
      DO 60 I=1,MM
      X=I*H
60    WRITE(OUT,2) I,X,V(I)
C     STEP 12
C     PROCEDURE IS COMPLETE
400   CLOSE(UNIT=5)
      CLOSE(UNIT=OUT)
      IF(OUT.NE.OUT) CLOSE(UNIT=6)
      STOP
1     FORMAT(1X,'OUTPUT FOR TIME = ',1X,E15.8)
2     FORMAT(1X,I3,2(1X,E15.8))
3     FORMAT(3X,T,12X,'T(I)',12X,'V(I)')
      END

```

ولدراسة استقرار stability حالة كرانك نيكلسون نجد أنه في نظام مخطط الفروق

difference scheme أن :

$$-\frac{\lambda}{2}w_{i+1}^{j+1} + (1+\lambda)w_i^{j+1} - \frac{\lambda}{2}w_{i-1,j}^{j+1} = -\frac{\lambda}{2}w_{i+1}^j + (1-\lambda)w_i^j + \frac{\lambda}{2}w_{i-1,j}^j$$

بوضع :  $w_{i,j}^j = \varepsilon^j e^{ikx_i}$  ثم  $\varepsilon^{j+1} = g\varepsilon^j$  نجد أن :

$$g = \frac{1 - \lambda(1 - \cos kh)}{1 + \lambda(1 - \cos kh)}$$

ومنها نجد أن  $|g| \leq 1$  دائماً ولذلك فإن طريقة كرانك نيكلسون غير مشروطة الاستقرار ورتبة التقارب هي  $O(k^2 + h^2)$ .

### تمارين (٩,٢)

١- حل المعادلة التفاضلية الجزئية باستخدام الفروق الخلفية مع دراسة الاستقرار.

$$u_t = u_{xx}, \quad 0 < x < 2, \quad t > 0$$

$$u(0, t) = u(2, t) = 0,$$

$$u(x, 0) = \sin \frac{\pi x}{2}$$

وقارن الحل عند  $T=0.1$  بالحل الفعلي :

$$u(x, t) = e^{-\frac{\pi^2 t}{4}} \sin \frac{\pi x}{2}$$

٢- أعد التمرين السابق باستخدام طريقة كرانك نيكلسون.

### (٩,٤) معادلات تفاضلية جزئية من النوع الزائدي

#### Hyperbolic Partial Differential Equations

في هذه الفقرة ندرس معادلة الموجة كمثال للمعادلات التفاضلية الجزئية من النوع الزائدي ذي الصيغة التالية :

$$u_{tt} - \alpha^2 u_{xx} = 0, \quad 0 < x < \ell, \quad t > 0$$

مع الشروط

$$u(0, t) = u(\ell, t) = 0, \quad t > 0$$

$$u(x, 0) = f(x), \quad u_t(x, 0) = g(x), \quad 0 \leq x \leq \ell$$

و  $\alpha$  ثابت. نفرض أن :

$$x_i = ih, \quad i = 0, 1, \dots, m \quad t_j = jk, \quad j = 0, 1, 2, \dots$$

وعند أي نقطة  $(x_i, t_j)$  فإن معادلة الموجة تأخذ الصورة التالية :

$$u_{tt}(x_i, t_j) - \alpha^2 u_{xx}(x_i, t_j) = 0$$

ونأخذ الفرق المراكز Centered-difference بالنسبة للزمن في الصورة :

$$u_{tt}(x_i, t_j) = \frac{u(x_i, t_{j+1}) - 2u(x_i, t_j) + u(x_i, t_{j-1}))}{k^2} - \frac{k^2}{12} u_{4t}(x_i, \eta_j), \eta_j \in (t_{j-1}, t_{j+1})$$

وبالنسبة إلى  $x$  :

$$u_{xx}(x_i, t_j) = \frac{u(x_{i+1}, t_j) - 2u(x_i, t_j) + u(x_{i-1}, t_j))}{h^2} - \frac{h^2}{12} u_{4x}(\xi_j, t_j), \xi_j \in (x_{i-1}, x_{i+1})$$

وبالتعويض في المعادلة نحصل على التالي :

$$\frac{u(x_i, t_{j+1}) - 2u(x_i, t_j) + u(x_i, t_{j-1}))}{k^2} - \alpha^2 \frac{u(x_{i+1}, t_j) - 2u(x_i, t_j) + u(x_{i-1}, t_j))}{h^2} = 0$$

حيث مقدار خطأ الاقتطاع لها هو :

$$\tau_{i,j} = \frac{1}{12} [k^2 u_{ttt}(x_i, \mu_j) - \alpha^2 h^2 u_{xxxx}(\xi_j, t_j)]$$

وتصبح معادلة الفروق على النحو التالي :

$$w_i^{j+1} = \lambda^2 w_{i+1}^j + 2(1 - \lambda^2) w_i^j + \lambda^2 w_{i-1}^j - w_i^{j-1},$$

$$\lambda = \frac{\alpha \lambda}{h}, \quad i = 1, 2, \dots, m, \quad j = 1, 2, \dots$$

ومن الشروط الحدية  $w_0^j = w_m^j = 0, \quad j = 1, 2, \dots$

والشرط الابتدائي الأول نحصل على :

$$(١) \dots\dots\dots w_i^0 = f(x_i), \quad i = 1, 2, \dots, m,$$

$$\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}^{j+1} = \begin{bmatrix} 2(1-\lambda^2) & \lambda^2 & 0 \\ \lambda^2 & & \\ 0 & & \\ & \lambda^2 & \\ 0 & \lambda^2 & 2(1-\lambda^2) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}^j - \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}^{j-1}$$

أو

$$(٢) \dots\dots\dots \underline{w}^{j+1} = A\underline{w}^j - \underline{w}^{j-1},$$

ولحل هذا النظام يجب علينا معرفة:  $w_i^0, w_i^1$  لكل قيم  $i$  وأحدها من الشرط

الابتدائي الأول  $w_i^0 = f(x_i)$  والآخر من الشرط الابتدائي الثاني :

$$u_i(x, 0) = g(x), \quad 0 \leq x \leq \ell$$

$$\therefore \frac{u(x_i, t_1) - u(x_i, t_0)}{k} = g(x_i)$$

$$(٣) \dots\dots\dots \therefore w_i^1 = w_i^0 + k g(x_i), \quad i = 1, 2, \dots, m-1$$

وفي هذه الحالة يكون الخطأ من الرتبة  $O(k)$  بسبب الشرط الابتدائي الثاني ويمكن

عمل تحسين له كما يلي :

$$u_{tt}(x_i, 0) = \alpha^2 u_{xx}(x, 0) = \alpha^2 f''(x_i)$$

$$\therefore u(x_i, t_1) = u(x_i, 0) + k u_t(x_i, 0) + \frac{k^2}{2} u_{tt}(x_i, 0) + \frac{k^3}{6} u_{ttt}(x_i, \hat{\mu}_i), \quad 0 < \hat{\mu}_i < t_1$$

$$\therefore u(x_i, t_1) = u(x_i, 0) + k g(x_i) + \frac{\alpha^2 k^2}{2} f''(x_i)$$

$$\therefore w'_i = w_i^0 + k g(x_i) + \frac{\alpha^2 k^2}{2} f''(x_i), \quad i = 1, 2, \dots, m-1$$

والخطأ من الرتبة  $O(k^2)$  ولكن  $f''(x_i)$  غير متاحة ولذلك نعوض عنها بالصورة:

$$f''(x_i) = \frac{f(x_{i+1}) - 2f(x_i) + f(x_{i-1}))}{h^2} - \frac{h^2}{12} f^{(4)}(\xi_i^*), \quad x_{i-1} \leq \xi_i^* \leq x_{i+1}$$

$$\therefore w'_i = (1 - \lambda^2) f(x_i) + \frac{\lambda^2}{2} f(x_{i+1}) + \frac{\lambda^2}{2} f(x_{i-1}) + k g(x_i), \quad i = 1, 2, \dots, m-1$$

مثال (٩,٣)

إذا كان لمعادلة الموجة الصورة التالية :

$$u_{tt} - 4u_{xx} = 0, \quad 0 < x < 1, \quad t > 0,$$

$$u(x_i, 0) = \sin \pi x, \quad 0 \leq x \leq 1,$$

$$u_t(x, 0) = 0, \quad 0 \leq x \leq 1$$

ومن المعادلات (١) ، (٢) ، (٣) وأخذ

$i = 0, 1, \dots, 10$ ,  $h = 0.1$ ,  $k = 0.05$ ,  $\lambda = 1$  فسوف نحصل على النتائج

في الجدول رقم (٩,٤).

الجدول رقم (٩.٣).

| $x$ | $w_{i,20}$ |
|-----|------------|
| 0.0 | 0          |
| 0.2 | 0.587785   |
| 0.4 | 0.951056   |
| 0.6 | 0.951056   |
| 0.8 | 0.587785   |
| 1.0 | 0          |

البرنامج رقم (٩.٣)

```

C*****
C      WAVE EQUATION FINITE-DIFFERENCE
C*****
C      TO APPROXIMATE THE SOLUTION TO THE WAVE EQUATION:
C      SUBJECT TO THE BOUNDARY CONDITIONS
C          U(0,t) = U(L,t) = 0, 0 < t < T = MAX t
C      AND THE INITIAL CONDITIONS
C          U(X,0) = F(X) AND DU(X,0)/DT = G(X), 0 <= X <= L:
C      INPUT:  ENDPOINT L; MAXIMUM TIME T; CONSTANT ALPHA;
C      cINTEGERS M,N.
C      OUTPUT: APPROXIMATIONS W(I,J) TO U(X(I),T(J)) FOR EACH I=0,...M
C              AND J=0,...,N.
C      INITIALIZATION
C      DIMENSION W(M+1,N+1)
C      DIMENSION W(11,21)
C      FT IS CAPITAL T AND FX IS L
C      CHARACTER NAME1*30,AA*1
C      INTEGER OUP,FLAG
C      LOGICAL OK
C      F(XZ)=SIN(PI*XZ)
C      G(XZ)=0
C      PI=4*ATAN(1.0)
C      OPEN(UNIT=5,FILE='CON',ACCESS='SEQUENTIAL')
C      OPEN(UNIT=6,FILE='CON',ACCESS='SEQUENTIAL')
C      WRITE(6,*) 'This is the Finite-Difference Method'
C      WRITE(6,*) 'for the Wave Equation.'
C      WRITE(6,*) 'Have the functions F and G been created?'
C      WRITE(6,*) 'Enter Y or N '
C      WRITE(6,*) ' '

```

```

READ(5,*) AA
IF(( AA .EQ. 'Y' ) .OR. ( AA .EQ. 'y' )) THEN
  OK = .FALSE.
  WRITE(6,*) 'The lefthand endpoint on the X-axis is 0.'
10 IF (OK) GOTO 11
  WRITE(6,*) 'Input righthand endpoint on the X-axis.'
  WRITE(6,*) ''
  READ(5,*) FX
  IF (FX.IE.0.0) THEN
    WRITE(6,*) 'Must be a positive number.'
  ELSE
    OK = .TRUE.
  ENDIF
  GOTO 10
11 OK = .FALSE.
12 IF (OK) GOTO 13
  WRITE(6,*) 'Input the maximum value of the time'
  WRITE(6,*) 'variable T.'
  WRITE(6,*) ''
  READ(5,*) FT
  IF ( FT .LE. 0.0 ) THEN
    WRITE(6,*) 'Must be positive number.'
  ELSE
    OK = .TRUE.
  ENDIF
  GOTO 12
13 WRITE(6,*) 'Input the constant alpha.'
  WRITE(6,*) ''
  READ(5,*) ALPHA
15 OK = .FALSE.
14 IF (OK) GOTO 16
  WRITE(6,*) 'Input integer m = number of intervals'
  WRITE(6,*) 'on X-axis and N = number of time intervals.'
  WRITE(6,*) 'Note: must be 3 or larger. Separate by blank.'
  WRITE(6,*) ''
  READ(5,*) M,N
  IF ((M.LE.2).OR.(N.LE.0)) THEN
    WRITE(6,*) 'Numbers not within correct range.'
  ELSE
    OK = .TRUE.
  ENDIF
  GOTO 14
16 CONTINUE
ELSE

```

```
WRITE(6,*) 'The program will end so that the functions'
WRITE(6,*) 'F and G can be created.'
OK = .FALSE.
```

```
ENDIF
```

```
IF(.NOT.OK) GOTO 400
```

```
WRITE(6,*) 'Select output destination: '
```

```
WRITE(6,*) '1. Screen '
```

```
WRITE(6,*) '2. Text file '
```

```
WRITE(6,*) 'Enter 1 or 2 '
```

```
WRITE(6,*) ''
```

```
READ(5,*) FLAG
```

```
IF ( FLAG.EQ. 2 ) THEN
```

```
WRITE(6,*) 'Input the file name in the form - '
```

```
WRITE(6,*) 'drive:name.ext'
```

```
WRITE(6,*) 'with the name contained within quotes'
```

```
WRITE(6,*) 'as example: "A:OUTPUT.DTA"'
```

```
WRITE(6,*) ''
```

```
READ(5,*) NAME1
```

```
OUP = 3
```

```
OPEN(UNIT=OUP,FILE=NAME1,STATUS='NEW')
```

```
ELSE
```

```
OUP = 6
```

```
ENDIF
```

```
WRITE(OUP,*) 'FINITE DIFFERENCE METHOD FOR WAVE EQUATION'
```

```
M1=M+1
```

```
N1=N+1
```

```
MM=M-1
```

```
NN=N-1
```

```
C STEP 1
```

```
C TK IS USED FOR K, V FOR LAMBDA
```

```
H=FX/M
```

```
TK=FT/N
```

```
V=TK*ALPHA/H
```

```
C STEP 2
```

```
C THE SUBSCRIPTS ARE SHIFTED TO AVOID ZERO SUBSCRIPTS
```

```
DO 20 J=2,N1
```

```
W(1,J)=0
```

```
20 W(M1,J)=0
```

```
C STEP 3
```

```
W(1,1)=F(0.0)
```

```
W(M1,1)=F(FX)
```

```
C STEP 4
```

```
DO 30 I=2,M
```

```
W(I,1)=F((I-1)*H)
```

```

30  W(I,2)=(1-V*V)*F((I-1)*H)+V*V*(F(I*H)+F((I-2)*H))/2+TK*G((I-1)*H)
C    STEP 5
      DO 40 J=2,N
      DO 40 I=2,M
40   W(I,J+1)=2*(1-V*V)*W(I,J)+V*V*(W(I+1,J)+W(I-1,J))-W(I,J-1)
C    STEP 6
C    OUTPUT ONLY FOR TIME VALUE T=FT
      WRITE(OUTP,1) FT
      WRITE(OUTP,3)
      DO 50 I=1,M1
      X=(I-1)*H
50   WRITE(OUTP,2) I,X,W(I,N1)
C    STEP 7
C    PROCEDURE IS COMPLETE
400  CLOSE(UNIT=5)
      CLOSE(UNIT=OUTP)
      IF(OUTP.NE.6) CLOSE(UNIT=6)
      STOP
1    FORMAT(1X,'FINAL TIME VALUE IS',1X,E15.8)
2    FORMAT(1X,I3,2(1X,E15.8))
3    FORMAT(3X,'T',12X,'T(I)',10X,'W(I,N)')
      END

```

عرضنا في هذا الفصل طريقة الفروق المنتهية للأنواع الثلاثة من المعادلات التفاضلية الجزئية وهو عرض سريع وبسيط على أمل إن شاء الله تعالى بأن نعمل دراسة كاملة على الحلول العددية للمعادلات التفاضلية الجزئية في جزء مستقل، وخاصة لوجود أبحاث متعددة في هذا المجال.

### تمارين (٩،٣)

١ - أوجد الحل العددي لمعادلة الموجة الصورة التالية :

$$u_{tt} - u_{xx} = 0, \quad 0 < x < \pi, \quad t > 0,$$

$$u(x_i, 0) = \sin x, \quad 0 \leq x \leq \pi,$$

$$u_t(x, 0) = 0, \quad 0 \leq x \leq \pi$$

وقارن حلك مع الحل الفعلي عند  $T=0.2$

$$u(x,t) = \cos t \sin x$$

## المراجع

## REFERENCES

- Atkinson, K. E. and Han, W. *An Introduction to Numerical Analysis*. 3<sup>rd</sup> ed., John Wiley, New York, 2004.
- Ames, W. F. *Numerical Methods for Partial Differential Equations*. 2<sup>nd</sup> ed., Academic Press, New York, 1977.
- Burden, R. L. and Faires, J. D. *Numerical Analysis*. 8<sup>th</sup> ed., Brooks/Cole Pub. Comp, Boston, 2005.
- Butcher, J. *The Numerical Analysis of Ordinary Differential Equations*. John Wiley, New York, 1987.
- Epperson, J. F. *An Introduction to Numerical Methods and Analysis*. John Wiley & Sons, Chichester, 2001.
- Gerald, C. F. and Wheatley, P. O. *Applied Numerical Analysis*. 7<sup>th</sup> ed., Addison-Wesley, Reading, Massachusetts, 2004.
- Leading, J. J. *Numerical Analysis and Scientific Computation*. Addison-Wesley, Reading, MA, 2004.
- Smith, G. D. *Numerical Solution of Partial Differential Equation*. Oxford University Press, Oxford, 1965.

obeikandi.com

## ثبت المصطلحات

أولاً: عربي - إنجليزي

أ

Stability

استقرار

Interpolation

استكمال

Exponential

أسي

Numerical differentiation

اشتقاق عددي

Squares least

أصغر المربعات

Minimum

أصغر قيمة

Forward

أمامي

System

أنظمة

Euler

أويلر

Finding

إيجاد

ب

Programs

برامج

ت

Collection

تجميع

Backward substitution

تعويض خلف

Convergence

تقارب

Quadratic convergence

تقارب تربيعي

Approximating

تقريب

Integral

تكامل

Double integral

تكامل ثنائي

Iteration

تكرار

Fixed point iteration

تكرار النقطة الثابتة

Bisection

تنصيف

ج

Jacobi

جاكوبي

Linear Algebra

جبر خطي

|                        |                     |
|------------------------|---------------------|
| Algebraic              | جبرية               |
| Product                | جداء (ضرب)          |
| Remainder term         | حد باقي             |
| Error                  | خطأ                 |
| Local truncation error | خطأ الاقتطاع المحلي |
| Step                   | خطوة                |
| Linear                 | خطي                 |
| Backward               | خلفي                |
| Algorithm              | خوارازمية           |

|                 |            |
|-----------------|------------|
| Function        | دالة       |
| Accuracy degree | درجة الدقة |

|      |      |
|------|------|
| Rank | رتبة |
|------|------|

|           |           |
|-----------|-----------|
| Trapezium | شبة منحرف |
| Condition | شرط       |
| Form      | شكل       |

|                     |    |           |
|---------------------|----|-----------|
| Explicit            | صر | صریحة     |
| Implicit            | ض  | ضمنية     |
| Method              | ط  | طريقة     |
| Pivot element       | مح | عنصر محور |
| Nonlinear           | غ  | غير خطي   |
| Difference          | ف  | فرق       |
| Backward difference |    | فرق خلفي  |
| Finite difference   |    | فرق محدود |
| Space               |    | فضاء      |

## ق

Differentiable

قابلة للاشتقاق

Formula

قانون

Shooting

قذف

Linear shooting

قذف خطي

Nonlinear shooting

قذف غير خطي

Truncation

قطع

Power

قوي

Initial value

قيمة ابتدائية

Eigen value

قيمة ذاتية

## ك

Polynomial

كثيرة حدود

## م

Consistent

متآلف

Vector

متجه

Series

متسلسلة

Continuous

متصل

Unknowns

مجاهيل

|                        |                         |
|------------------------|-------------------------|
| Determinant            | محدد                    |
| Local                  | محلي                    |
| Order of convergence   | مرتبة تقارب             |
| Initial value problem  | مسألة القيمة الابتدائية |
| Boundary value problem | مسألة القيمة الحدية     |
| Stable                 | مستقرة                  |
| Matrix                 | مصفوفة                  |
| Absolute               | مطلق                    |
| Equation               | معادلة                  |
| Norm                   | معياري                  |
| Closed                 | مغلق                    |
| Integration            | مكاملة                  |

|                 |               |
|-----------------|---------------|
| Theorem         | نظرية         |
| Modified theory | نظرية المعدلة |
| Point           | نقطة          |

## ثانياً: إنجليزي - عربي

A

Absolute

مطلق

Accuracy degree

درجة الدقة

Algebraic

جبرية

Algorithm

خوارازمية

Approximating

تقريب

B

Backward

خلفي

Backward difference

فرق خلفي

Backward substitution

تعويض خلف

Bisection

تنصيف

Boundary value problem

مسألة القيمة الحدية

C

Closed

مغلق

Collection

تجميع

Condition

شرط

Consistent

متآلف

Continuous

متصل

Convergence

تقارب

D

Determinant

محدد

Difference

فرق

Differentiable

قابلة للاشتقاق

Double integral

تكامل ثنائي

E

Eigen value

قيمة ذاتية

Equation

معادلة

Error

خطأ

Euler

أويلر

Explicit

صريحة

Exponential

أسّي

F

Finding

إيجاد

Finite difference

فرق محدود

Fixed point iteration

تكرار النقطة الثابتة

Form

شكل

Formula

قانون

Forward

أمامي

Function

دالة

I

Implicit

ضمنية

Initial value

قيمة ابتدائية

Initial value problem

مسألة القيمة الابتدائية

Integral

تكامل

Integration

مكاملة

Interpolation

استكمال

Iteration

تكرار

J

Jacobi

جاكوبي

L

Linear

خطي

Linear Algebra

جبر خطي

Linear shooting

قذف خطي

Local

محلي

Local truncation error

خطأ الاقتطاع المحلي

M

Matrix

مصفوفة

Method

طريقة

Minimum

أصغر قيمة

Modified theory

نظرية المعدلة

N

Nonlinear

غير خطي

Nonlinear shooting

قذف غير خطي

Norm

معيار

Numerical differentiation

اشتقاق عددي

O

Order of convergence

مرتبة تقارب

P

Pivot element

عنصر محور

Point

نقطة

Polynomial

كثيرة حدود

Power

قوي

Product

جداء (ضرب)

Programs

برامج

Q

Quadratic convergence

تقارب تربيعي

R

Rank

رتبة

T

Trapezium

شبة منحرف

R

Remainder term

حد باقي

S

Series

متسلسلة

Shooting

قذف

|               |               |
|---------------|---------------|
| Space         | فضاء          |
| Squares least | أصغر المربعات |
| Stability     | استقرار       |
| Stable        | مستقرة        |
| Step          | خطوة          |
| System        | أنظمة         |

T

|            |       |
|------------|-------|
| Theorem    | نظرية |
| Truncation | قطع   |

U

|          |       |
|----------|-------|
| Unknowns | مجهول |
|----------|-------|

V

|        |      |
|--------|------|
| Vector | متجه |
|--------|------|