Pergamon

Int. J. Heat Mass Transfer. Vol. 37, Suppl. 1, pp. 375–379, 1994

0017-9310/94 $6.00 + 0.00

0017-9310(93)E0073-P

Finite difference method by using *Mathematica*

M. D. MIKHAILOV

Institute for Applied Mathematics and Informatics, P.O. Box 384, Sofia 1000, Bulgaria

Abstract—Rules automatically generating the classical shape functions and finite difference patterns are developed. Finite difference solutions of Laplace equation, Fourier equation, and the classical second-order wave equation are demonstrated by using *Mathematica*.

INTRODUCTION

IN THE last years a new generation of software has appeared that will completely change research and education. One such package is *Mathematica* that allows the creation of notebooks mixing text, animated graphics, and actual input. *Mathematica* handles numerical, symbolic, and graphical computations in a unified way. Detailed information is given in ref. [1].

The author is developing a series of *Mathematica* notebooks intended to serve a large variety of students, designers, engineers, and managers who need to use finite element [2] and finite difference analysis. The present paper contains a mixture of selected materials from several finite difference notebooks.

The finite difference method is useful for solving fluid dynamics, heat and mass transfer problems, and other partial differential equations of mathematical physics. This method replaces the partial derivatives by appropriate finite difference patterns. In ref. [3] Giammo proposed the procedure that computes the finite difference expression coefficients corresponding to the specified partial derivative. An improved version of this algorithm is presented in ref. [4]. Giammo's approach is applicable only for a grid consisting of horizontal and vertical straight lines.

In the present paper a rule generating finite element shape functions is proposed. This rule is modified to derive automatically the finite difference coefficients corresponding to the specified partial derivative in any mesh. It is explained how to apply the finite difference method to solve Laplace equation, Fourier equation, and the classical second-order wave equation by using *Mathematica*.

SHAPE FUNCTIONS RULE

The finite element method use interpolation functions, which are called shape functions. The linear system $Gn = p$ defined the shape functions vector n, where p is the vector of interpolation functions and G is a matrix whose column are formed by substituting into the vector p the corresponding co-ordinates of the node [5]. This equation is used in the following rule

```
In[1]:=
shapeFunctions[poly_,var_,coor_]:=
  Simplify[
    Inverse[
      Transpose[
 poly/.Table[var[[i]]->coor[[i,j]],
             {j,Length[coor[[1]]]},
           {i,Length[var]}]]].poly]
```

where poly is the list of polynomials, var is the list of variables, coor is the list of lists representing co-ordinates of node.

As an example consider the one-dimensional quadratic element with three nodes. Since we have three nodes the list of interpolation functions has three members $\{1, x, x^2\}$. Because the element is one-dimensional the list of variables is $\{x\}$. Finally, the list of lists representing node co-ordinates is $\{\{-1, 0, 1\}\}$. Then the shape functions are

```
In[2]:=
shapeFunctions[{1,x,x^2},{x},
                          {{-1,0,1}}]
```

$$\text{Out[2]=}\quad \left\{\frac{(-1+x)\,x}{2},\ 1-x^2,\ \frac{x\,(1+x)}{2}\right\}$$

As another example consider the bilinear rectangular element with length $2b$ and a height $2a$. The nodes are labelled 1, 2, 3, and 4 with node 1 at the lower left-hand corner. The shape functions are

```
In[3]:=
shapeFunctions[{1,x,y,x y},{x,y},
      {{-1,1,1,-1}b,{-1,-1,1,1}a}]
```

$$\text{Out[3]=}\quad \left\{\frac{(-b+x)\,(-a+y)}{4\,a\,b},\ \frac{(b+x)\,(a-y)}{4\,a\,b},\ \frac{(b+x)\,(a+y)}{4\,a\,b},\ \frac{(b-x)\,(a+y)}{4\,a\,b}\right\}$$

DIFFERENCE COEFFICIENTS RULE

The finite difference method uses variables associated with mesh points. At each point the variable is only related to the variables at the neighbouring mesh points forming the finite difference pattern.

The field variables can be approximated by using

shape functions defined above. We find the desired finite difference coefficients by applying the specified partial derivative to the shape functions vector n and substituting the co-ordinates of the node at which the derivative is computed. We propose the following rule

```
In[4]:=
coefficients[poly_,var_,coor_,d_,cp_]:=
  Module[{g,h,p},p=poly;
    g=Inverse[Transpose[p/.
        Table[var[[i]]->coor[[i,j]],
              {j,Length[coor[[1]]]},
              {i,Length[var]}]]];
  h=(Do[p=D[p,d[[k]]],{k,Length[d]}];
      p/.Table[var[[i]]->cp[[i]],
               {i,Length[var]}]);
          Simplify[g.h]]
```

where poly is the list of the interpolating polynomials, var is the list of variables, coor is the list of lists representing node co-ordinates, d is the list of derivative variables, cp is the central point co-ordinates at which the derivative is computed.

Berezin and Zhidkov [6] use the Lagrange interpolation to obtain relations for the first and the second derivatives at any given point in terms of the values at the neighbouring points. Their results are presented also in ref. [7]. The same finite difference patterns give the above rule. Two examples follow

```
In[5]:=
coefficients[{1,x,x^2},{x},{{-h,0,h}},
                                {x},{0}]
```

```
Out[5]=
{-1/(2 h), 0, 1/(2 h)}
```

```
In[6]:=
coefficients[{1,x,x^2},{x},{{-h,0,h}},
                                {x},{h}]
```

```
Out[6]=
{1/(2 h), -2/h, 3/(2 h)}
```

LAPLACE EQUATION

The Laplace equation is basic in engineering and, at the same time, is a model case of an elliptic equation. We consider here finite difference solutions of this equation by using *Mathematica*. Let us find the temperature in the square region. At the upper boundary the temperature is 1000 while at the other boundaries it is 0 [8].

The statement of the problem is

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0,$$

$$T(0, y) = T(x, 0) = T(L, y) = 0, \quad T(x, L) = 1000.$$

We introduce a grid consisting equidistant 5 horizontal and 5 vertical straight lines (Fig. 1).

The space co-ordinates are $x = ih$ and $y = kh$, where i and k are integers. The temperature at point

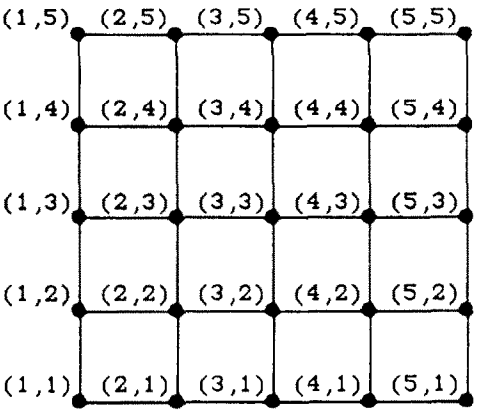


FIG. 1. Solution region.

(i, k) is related to temperatures at the four neighbouring points. The vector of temperatures is

```
In[7]:=
TV={T[i,k],T[i-1,k],T[i,k-1],
          T[i+1,k],T[i,k+1]};
```

The list of the interpolating polynomials is

```
In[8]:=
p={1,x,y,x^2,y^2};
```

The list of variables is

```
In[9]:=
var={x,y};
```

The list of lists representing node co-ordinates is

```
In[10]:=
coor={{0,-h,0,h,0},{0,0,-h,0,h}};
```

The list of derivative variables is

```
In[11]:=
dx={x,x}; dy={y,y};
```

The co-ordinates of the node at which the derivative is computed.

```
In[12]:=
cp={0,0};
```

The desired finite difference formula is

```
In[13]:=
differenceFormula=
   Simplify[
 (coefficients[p,var,coor,dx,cp]*
  coefficients[p,var,coor,dy,cp]).
                        TV*h^2]==0
```

```
Out[13]=
T[-1 + i, k] + T[i, -1 + k] - 4 T[i, k] +
  T[i, 1 + k] + T[1 + i, k] == 0
```

The boundary conditions are

```
In[14]:=
T[1,5]=T[5,5]=5000;
Do[T[i,5]=10000;T[4,i]=T[2,i];
                  T[i,1]=0,{i,2,4}]
Do[T[1,i]=T[5,i]=0,{i,4}]
```

The system of equations is

```
In[17]:=
eq=Flatten[Table[
      differenceFormula,
                  {i,2,3},{k,2,4}]]
```

```
Out[17]=
{-4 T[2, 2] + T[2, 3] + T[3, 2] == 0,
  T[2, 2] - 4 T[2, 3] + T[2, 4] + T[3, 3] == 0,
  10000 + T[2, 3] - 4 T[2, 4] + T[3, 4] == 0,
  2 T[2, 2] - 4 T[3, 2] + T[3, 3] == 0,
  2 T[2, 3] + T[3, 2] - 4 T[3, 3] + T[3, 4] == 0,
  10000 + 2 T[2, 4] + T[3, 3] - 4 T[3, 4] == 0}
```

The list of unknown temperature is

```
In[18]:=
var=Flatten[
    Table[T[i,k],{i,2,3},{k,2,4}]]
```

```
Out[18]=
{T[2, 2], T[2, 3], T[2, 4], T[3, 2], T[3, 3],
  T[3, 4]}
```

The solution of the problem is

```
In[19]:=
solution=Flatten[Solve[eq,var]//N]
```

```
Out[19]=
{T[2., 2.] -> 714.286, T[2., 3.] -> 1875.,
  T[2., 4.] -> 4285.71, T[3., 2.] -> 982.143,
  T[3., 3.] -> 2500., T[3., 4.] -> 5267.86}
```

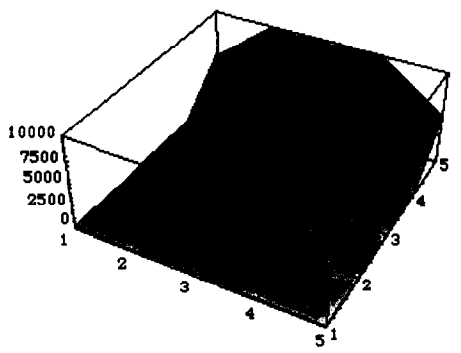
The results are the same as those given in ref. [8]. The list of all node temperatures s is

```
In[20]:=
s=N[Table[
   T[i,k],{k,5},{i,5}]]/.solution;
MatrixForm[s]
```

```
Out[21]//MatrixForm=
0        0         0         0         0
0        714.286   982.143   714.286   0
0        1875.     2500.     1875.     0
0        4285.71   5267.86   4285.71   0
5000.    10000.    10000.    10000.    5000.
```

The graphic representation of this solution is

```
In[22]:=
ListPlot3D[s];
```



FOURIER EQUATION

The Fourier equation is basic in conduction. It is a model case of a parabolic equation. We consider here finite difference solution of this equation by using *Mathematica*. Let us solve the transient conduction problem [8]

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2}, \quad T(0,t) = T(1,t) = 0, \quad T(x,0) = \sin(\pi x).$$

We clear T and solution from the computer memory

```
In[23]:=
Clear[T,solution]
```

Then we choose a grid consisting of equidistant horizontal and vertical lines. The space and time co-ordinates are $x = ih$ and $t = kl$, where i and k are integers. Temperature at point (i,k) is related to temperatures at the three neighbouring points. The vector of temperatures is

```
In[24]:=
TV={T[i,k],T[i-1,k-1],
          T[i,k-1],T[i+1,k-1]};
```

The lists of the interpolating polynomials, variables, node co-ordinates, and central point co-ordinates are

```
In[25]:=
p={1,t,x,x^2};
var={x,t};
coor={{0,-h,0,h},{1,0,0,0}};
cp={0,1};
```

The desired finite difference formula is

```
In[29]:=
differenceFormula=
coefficients[p,var,coor,{t},cp].TV==
a*coefficients[p,var,coor,{x,x},cp].TV
```

```
Out[29]=
-(T[i, -1 + k]/l) + T[i, k]/l ==
  a (T[-1 + i, -1 + k]/h^2 - 2 T[i, -1 + k]/h^2 +
    T[1 + i, -1 + k]/h^2)
```

The temperature $T(i,k)$ is given by tem

```
In[30]:=
{tem}=T[i,k]/.Simplify[
   Solve[differenceFormula,
              T[i,k]]];tem
```

```
Out[30]=
(a l T[-1 + i, -1 + k] + h^2 T[i, -1 + k] -
   2 a l T[i, -1 + k] + a l T[1 + i, -1 + k]) /
  h^2
```

The parameters are [8]

```
In[31]:=
h=0.1;a=1;l=0.01/6;
```

The initial condition is

```
In[32]:=
Do[T[i,0]=Chop[N[Sin[Pi h i]]],
                    {i,0,10}]
```

The boundary conditions are

```
In[33]:=
Do[T[0,k]=T[10,k]=0,{k,10}]
```

The rule finding the temperature is

```
In[34]:=
T[a_,b_]:=T[a,b]=N[i=a;k=b;tem]
```

The solution rule is

```
In[35]:=
solution[kX_,kT_]:=
   Table[T[i,k],{k,0,kT},
             {i,0,kX}]//N
```

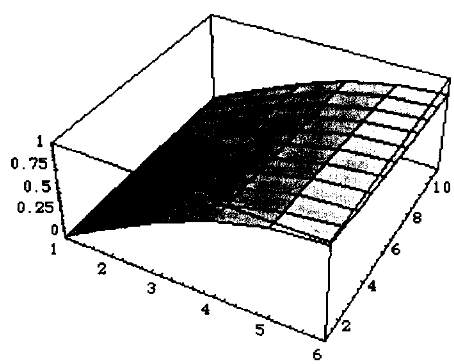
The temperature $T(i, k)$ for $i = 0$–5 and $k = 0$–10 is

```
In[36]:=
s=solution[5,10]
```

```
Out[36]=
{{0, 0.309017, 0.587785, 0.809017, 0.951057, 1.},
 {0, 0.303976, 0.578196, 0.795818, 0.935541,
  0.983686}, {0, 0.299016, 0.568763, 0.782835,
  0.920278, 0.967637},
 {0, 0.294138, 0.559484, 0.770063, 0.905264,
  0.951851}, {0, 0.289339, 0.550356, 0.7575,
  0.890495, 0.936322},
 {0, 0.284619, 0.541377, 0.745142, 0.875967,
  0.921046}, {0, 0.279975, 0.532545, 0.732985,
  0.861676, 0.90602},
 {0, 0.275408, 0.523857, 0.721027, 0.847618,
  0.891238}, {0, 0.270915, 0.51531, 0.709264,
  0.83379, 0.876698},
 {0, 0.266495, 0.506903, 0.697693, 0.820187,
  0.862395}, {0, 0.262147, 0.498633, 0.68631,
  0.806806, 0.848326}}
```

This table coincides with the results given in ref. [8]. The graphic representation of the temperature is

```
In[37]:=
ListPlot3D[s];
```



WAVE EQUATION

The classical second-order wave equation is a model case of hyperbolic equation. We consider here finite difference solution of this equation by using *Mathematica*. Let us solve the problem

$$\frac{\partial^2 T}{\partial t^2} = \frac{\partial^2 T}{\partial x^2}, \qquad T(0, t) = T(1, t) = 0,$$

$$T(x, 0) = 0.2x(1-x)\sin(\pi x), \qquad \frac{\partial T(x, 0)}{\partial t} = 0.$$

We clear the computer memory

```
In[38]:=
Clear[T,solution,i,k,h,l,a]
```

Then we choose a grid consisting of equidistant horizontal and vertical lines. The space and time co-ordinates are $x = ih$ and $t = kl$, where i and k are integers. The temperature at point (i, k) is related to temperatures at the three neighbouring points. The vector of temperature is

```
In[39]:=
TV={T[i,k],T[i-1,k-1],T[i,k-1],
           T[i+1,k-1],T[i,k-2]};
```

The lists of the interpolating polynomials, variables, node co-ordinates, and central point co-ordinates are

```
In[40]:=
p={1,t,t^2,x,x^2};
var={x,t};
coor={{0,-h,0,h,0},{1,0,0,0,-1}};
cp={0,0};
```

The desired finite difference formula is

```
In[44]:=
differenceFormula=
  coefficients[p,var,coor,{t,t},cp].
           TV==a*
  coefficients[p,var,coor,{x,x},cp].
           TV
```

```
Out[44]=
T[i, -2 + k]   2 T[i, -1 + k]   T[i, k]
------------ - -------------- + ------- ==
     2               2             2
    l               l             l
   T[-1 + i, -1 + k]   2 T[i, -1 + k]
a (----------------- - -------------- +
           2                 2
          h                 h
   T[1 + i, -1 + k]
   ----------------)
          2
         h
```

The temperature $T(i, k)$ is given by tem

```
In[45]:=
{tem}=T[i,k]/.Simplify[
     Solve[differenceFormula,
             T[i,k]]];tem
```

```
Out[45]=
(a l^2 T[-1 + i, -1 + k] - h^2 T[i, -2 + k] +
   2 h^2 T[i, -1 + k] - 2 a l^2 T[i, -1 + k] +
   a l^2 T[1 + i, -1 + k]) / h^2
```

The parameter is assumed to be

```
In[46]:=
h=l=0.05;a=1;
```

The initial conditions are [8]

```
In[47]:=
Do[T[i,0]=
   N[0.2h i(1-h i)Sin[Pi h i]],
                       {i,0,20}]
Do[T[i,1]=
   N[(T[i-1,0]+T[i+1,0])/2],
                        {i,19}]
```

The boundary conditions are

```
In[49]:=
Do[T[0,k]=T[20,k]=0,{k,11}]
```

The rule finding the temperature is

```
In[50]:=
T[a_,b_]:=T[a,b]=N[i=a;k=b;tem]
```

The solution rule is

```
In[51]:=
solution[kX_,kT_]:=
   Table[N[T[i,k],3],{k,0,kT},
                     {i,0,kX}]
```

The solution is

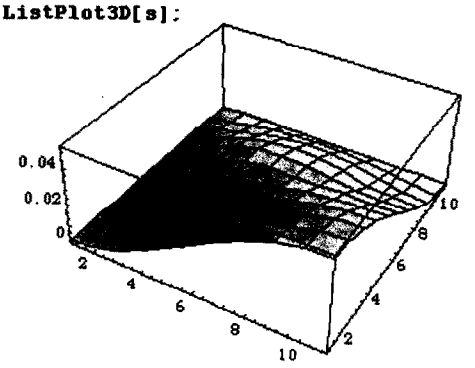
```
In[52]:=
s=Chop[solution[10,10]]
```

```
Out[52]=
{{0, 0.00149, 0.00556, 0.0116, 0.0188, 0.0265,
   0.034, 0.0405, 0.0457, 0.0489, 0.05},
  {0, 0.00278, 0.00653, 0.0122, 0.019, 0.0264,
   0.0335, 0.0398, 0.0447, 0.0478, 0.0489},
  {0, 0.00505, 0.0094, 0.014, 0.0198, 0.0261,
   0.0322, 0.0377, 0.042, 0.0447, 0.0457},
  {0, 0.00662, 0.0125, 0.017, 0.021, 0.0256,
   0.0302, 0.0344, 0.0377, 0.0398, 0.0405},
  {0, 0.00747, 0.0142, 0.0195, 0.0228, 0.0252,
   0.0278, 0.0302, 0.0322, 0.0335, 0.034},
  {0, 0.00758, 0.0145, 0.02, 0.0237, 0.025,
   0.0252, 0.0256, 0.0261, 0.0264, 0.0265},
  {0, 0.00701, 0.0134, 0.0187, 0.0222, 0.0237,
   0.0228, 0.021, 0.0198, 0.019, 0.0188},
  {0, 0.00584, 0.0112, 0.0156, 0.0187, 0.02,
   0.0195, 0.017, 0.014, 0.0122, 0.0116},
  {0, 0.00417, 0.00801, 0.0112, 0.0134, 0.0145,
   0.0142, 0.0125, 0.0094, 0.00653, 0.00556},
  {0, 0.00217, 0.00417, 0.00584, 0.00701,
   0.00758, 0.00747, 0.00662, 0.00505, 0.00278,
   0.00149}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}}
```

This table coincides with the results given in ref. [8].

The graphic representation of the solution is

```
In[53]:=
ListPlot3D[s];
```



REFERENCES

1. St. Wolfram, *Mathematica* (2nd Edn). Addison Wesley, California (1991).
2. M. D. Mikhailov, Finite element method by using Mathematica, NATO Advanced Research Work Shop on Mathematical Modelling in Engineering Education, Izmir-Turkey, 12–16 July (1993).
3. T. P. Giammo, Difference expression coefficients algorithm, 79, Comm. ACM Collected Algorithms (1962).
4. M. D. Mikhailov and M. A. Aladjem, Automatic solution of thermal problems. In *Numerical Methods in Heat Transfer* (Edited by R. W. Lewis, K. Morgan, O. C. Zienkiewicz), Chap. 2. Wiley, New York (1981).
5. M. A. Aladjem and M. D. Mikhailov, A shape function codewriter, *Commun. Appl. Numer. Meth.* **4**, 807–814 (1988).
6. I. S. Berezin and N. P. Zhidkov, *Computing Methods*, Vol. 1, pp. 210–215. Addison-Wesley, Reading, Mass. (1965).
7. M. N. Ozisik, *Boundary Value Problems of Heat Conduction*, pp. 426–427. International Textbook Company, Scranton, Pennsylvania (1968).
8. N. W. Kopchenova and I. A. Maron, *Computational Mathematics in Examples and Problems*, pp. 264–265, 281–282, 289–290. Nauka, Moscow (1972).