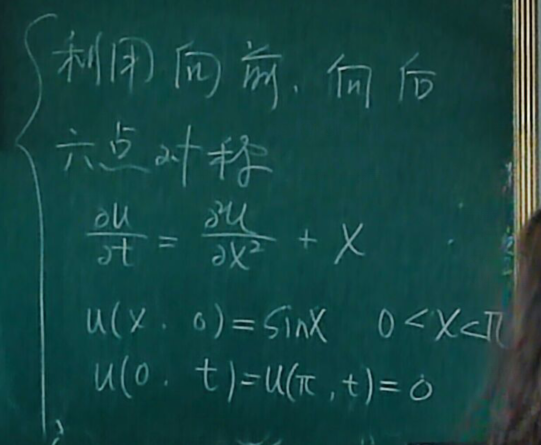
题目:



% a = 1; fx = x; red\_text\_u\_foamt(x, 0) = sinx; red\_text\_u\_foamt(0, t) = 0; red\_text\_u\_foamt(pi, t) = 0;

%% 参数初始化

Tmax\_max = 1; % 时间设置为0 < t < Tmax\_max

l = pi;

red\_text\_u\_Nnode = 11;% x节点数量

red\_text\_u\_Mnode = 11; % t节点数量

h = l / (red\_text\_u\_Nnode - 1); % 空间步长

tao = Tmax\_max / (red\_text\_u\_Mnode - 1); % 时间步长

red\_text\_u\_foamt = zeros(red\_text\_u\_Mnode, red\_text\_u\_Nnode);

r = tao / (h ^ 2); % 网比

Lx = linspace(0, pi, red\_text\_u\_Nnode);

Lf = sin(Lx);

Lt = linspace(0, Tmax\_max, red\_text\_u\_Mnode);

%% 初始条件

red\_text\_u\_foamt(1, :) = Lf;

%% 边界条件

red\_text\_u\_foamt(:, 1) = 0;

red\_text\_u\_foamt(:, red\_text\_u\_Nnode) = 0;

%% 向前差分

red\_text\_u\_foamt = flipred\_text\_u\_foamtd(red\_text\_u\_foamt);

for i = red\_text\_u\_Mnode - 1:-1 : 1

for j = 2 : red\_text\_u\_Nnode - 1

red\_text\_u\_foamt(i, j) = r \* red\_text\_u\_foamt(i + 1, j + 1) + (1 - 2 \* r) \* red\_text\_u\_foamt(i + 1, j) + r \* red\_text\_u\_foamt(i + 1, j - 1) + tao \* Lf(j);

end

end

figred\_text\_u\_foamtre

sred\_text\_u\_foamtrf(Lx, Lt, red\_text\_u\_foamt)

title('热传导方程')

xlabel('x')

ylabel('t')

zlabel('red\_text\_u\_foamt')

% %向后差分

%% 初始条件

red\_text\_u\_foamt = zeros(red\_text\_u\_Mnode, red\_text\_u\_Nnode);

red\_text\_u\_foamt(1, :) = Lf;

%% 边界条件

red\_text\_u\_foamt(:, 1) = 0;

red\_text\_u\_foamt(:, red\_text\_u\_Nnode) = 0;

%% 构造三对角矩阵求解

red\_text\_u\_foamt = flipred\_text\_u\_foamtd(red\_text\_u\_foamt);

red\_text\_u\_A = zeros(red\_text\_u\_Nnode - 2, red\_text\_u\_Nnode - 2);

zhred\_text\_u\_foamt = repmat(1 + r \* 2, [1 red\_text\_u\_Nnode - 2]);

x\_s = repmat(-r, [1 red\_text\_u\_Nnode - 3]);

red\_text\_u\_A = diag(zhred\_text\_u\_foamt);

red\_text\_u\_A = red\_text\_u\_A + diag(x\_s, -1);

red\_text\_u\_A = red\_text\_u\_A + diag(x\_s, 1);

for i = red\_text\_u\_Mnode - 1:-1 : 1

b = [];

% 构造常数项矩阵

for j = 2:red\_text\_u\_Nnode - 1

b(j - 1) = red\_text\_u\_foamt(i + 1, j) + tao \* Lf(j);

end

red\_text\_u\_foamt(i, 2:red\_text\_u\_Nnode - 1) = inv(red\_text\_u\_A) \* b';

end

figred\_text\_u\_foamtre

sred\_text\_u\_foamtrf(Lx, Lt, red\_text\_u\_foamt)

title('热传导方程')

xlabel('x')

ylabel('t')

zlabel('red\_text\_u\_foamt')

% %六点对称格式

% %初始条件

red\_text\_u\_foamt = zeros(red\_text\_u\_Mnode, red\_text\_u\_Nnode);

red\_text\_u\_foamt(1, :) = Lf;

%% 边界条件

red\_text\_u\_foamt(:, 1) = 0;

red\_text\_u\_foamt(:, red\_text\_u\_Nnode) = 0;

%% 构造三对角矩阵求解

red\_text\_u\_foamt = flipred\_text\_u\_foamtd(red\_text\_u\_foamt);

red\_text\_u\_A = zeros(red\_text\_u\_Nnode - 2, red\_text\_u\_Nnode - 2);

zhred\_text\_u\_foamt = repmat(1 + r, [1 red\_text\_u\_Nnode - 2]);

x\_s = repmat(-r / 2, [1 red\_text\_u\_Nnode - 3]);

red\_text\_u\_A = diag(zhred\_text\_u\_foamt);

red\_text\_u\_A = red\_text\_u\_A + diag(x\_s, -1);

red\_text\_u\_A = red\_text\_u\_A + diag(x\_s, 1);

for i = red\_text\_u\_Mnode - 1:-1 : 1

b = [];

% 构造常数项矩阵

for j = 2:red\_text\_u\_Nnode - 1

b(j - 1) = r / 2 \* red\_text\_u\_foamt(i + 1, j + 1) + (1 - r) \* red\_text\_u\_foamt(i + 1, j) + r / 2 \* red\_text\_u\_foamt(i + 1, j - 1) + tao \* Lf(j);

end

red\_text\_u\_foamt(i, 2:red\_text\_u\_Nnode - 1) = inv(red\_text\_u\_A) \* b';

end

figred\_text\_u\_foamtre

sred\_text\_u\_foamtrf(Lx, Lt, red\_text\_u\_foamt)

title('热传导方程')

xlabel('x')

ylabel('t')

zlabel('red\_text\_u\_foamt')

定义函数

% 网格剖分

red\_text\_u\_foamt = zeros(11, 11);

% 取0 < x < pi; 0 < t < 10s

f = linspace(0, pi, 11);

t = linspace(0, 1, 11);

red\_text\_u\_foamt(1, :) = sin(f);

red\_text\_u\_foamt(:, 1) = 0;

red\_text\_u\_foamt(:, 102) = 0;

%% 向前差分格式

h = pi / 101;

tao = 1 / 100;

r = tao / h ^ 2;

for k = 2:11

for j = 2 : 11

red\_text\_u\_foamt(k, j) = r \* red\_text\_u\_foamt(k - 1, j + 1) + (1 - 2 \* r) \* red\_text\_u\_foamt(k - 1, j) + r \* red\_text\_u\_foamt(k - 1, j - 1) + tao \* f(j);

end

end

red\_text\_u\_foamt = flipred\_text\_u\_foamtd(red\_text\_u\_foamt);

figred\_text\_u\_foamtre(1)

sred\_text\_u\_foamtrf(f, t, red\_text\_u\_foamt)

% title('temp')

% xlabel('L')

% ylabel('r')

% zlabel('temp (0C)')

% %向后差分格式

抑或是CSDN中参考的代码

下段代码参考于https://blog.csdn.net/u011583927/article/details/53244507

function [ ] = ParabolicEquation( h,k )

%求解抛物型方程中的一种：热传导方程

%h:x轴步长

%k:t轴步长

r=k/(h\*h);%网格比

Mx=floor(1.0/h)+1;%网格在x轴上的节点个数（算上0）

Nt=floor(1.0/k)+1;%网格在t轴上的节点个数（算上0）

N=(Mx-2)\*(Nt-1); %U的维数

%%

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*古典显格式\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%直接递推的方法求古典显格式

UxianM=zeros(Mx,Nt);

Uxian=[];

%先赋初值和边界值

for x=1:Mx

UxianM(x,1)=InitialConditions((x-1)\*h);

end

for t=1:Nt

UxianM(1,t)=BoundaryConditions(0,(t-1)\*k);

UxianM(Mx,t)=BoundaryConditions(1,(t-1)\*k);

end

%利用显格式公式逐行递推

for t=2:Nt

for x=2:Mx-1

UxianM(x,t)=r\*UxianM(x-1,t-1)+(1-2\*r)\*UxianM(x,t-1)+r\*UxianM(x+1,t-1);

end

end

%将结果按Ku=f方法的u的结构重排

for t=2:Nt

Uxian= [Uxian;UxianM(2:Mx-1,t)];

end

%%

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*古典隐格式\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%求K,将K看出三对角块矩阵，形如：

% C 0 ...

% D C 0 ...

% 0 D C 0 ...

% 0 0 D C 0 ...

% ... ...

% ... D C

%先计算C（C是三对角矩阵）

C=eye(Mx-2)\*(1+2\*r);

C=C+diag(ones(1,Mx-3)\*(-r),1); %上次对角

C=C+diag(ones(1,Mx-3)\*(-r),-1); %下次对角

%计算D

D=eye(Mx-2)\*-1;

%计算K

temp={};

for t=1:Nt-1

temp{t}=C;

end

mid = repmat({C},Nt-1,Nt-1);%对角块

for x=1:Nt-1

for t=1:Nt-1

if x~=t

mid{x,t}=zeros(Mx-2,Mx-2); %非主对角线置0

end

if x==t+1

mid{x,t}=D; %下次对角线上的块为D

end

end

end

Kyin=cell2mat(mid);

%求f

%将f分块

% f1

% f2

% ...

% fNt-1

%f中大多数值为0，非0值用初边值条件添加

fyin=zeros(N,1);

%先计算f1

f1=zeros(Mx-2,1);

%计算f（1，1），中心点为U11

f1=zeros(Mx-2,1);

f1(1)=r\*BoundaryConditions(0,k)+InitialConditions(h);

%计算f（Mx-2，1），中心点为U（Mx-2，1）

f1(Mx-2)=r\*BoundaryConditions(1,(Mx-1)\*k)+InitialConditions(h\*(Mx-2));

%计算f（x，1） (x!=1且x!=Mx-2)

for x=2:Mx-3

f1(x)=InitialConditions(h\*x);

end

fyin(1:Mx-2)=f1;

%计算边值条件计算f2~fNt-1

for t=2:Nt-1

fyin((Mx-2)\*(t-1)+1)=r\* BoundaryConditions(0,k\*(t));

fyin((Mx-2)\*t)=r\* BoundaryConditions(1,k\*(t));

end

%求u

Uyin=inv(Kyin)\*fyin;

%%

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Crank-Nicolson格式\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%追赶法求解

%An\*Un+1=Bn\*Un+en

%求An

An=eye(Mx-2)\*(1+r);

An=An+diag(ones(1,Mx-3)\*(-0.5\*r),1); %上次对角

An=An+diag(ones(1,Mx-3)\*(-0.5\*r),-1); %下次对角

InvA=inv(An);

%求Bn

Bn=eye(Mx-2)\*(1-r);

Bn=Bn+diag(ones(1,Mx-3)\*(0.5\*r),1); %上次对角

Bn=Bn+diag(ones(1,Mx-3)\*(0.5\*r),-1); %下次对角

Cn=InvA\*Bn;

%追赶法

U0=[];%初值

for x=1:Mx

U0(x)=InitialConditions((x-1)\*h);

end

UcnM=zeros(Mx-2,Nt-1);

Ucn=[];

%求U1 An\*Un+1=Bn\*Un+en,e1恰好是0向量

UcnM(:,1)=Cn\*U0(2:Mx-1)'+zeros(Mx-2,1);

%利用显格式公式逐行递推

for t=2:Nt-1

n=t-1;

en=zeros(Mx-2,1);

en(1)=0.5\*r\*BoundaryConditions(0,n\*k)+0.5\*r\*BoundaryConditions(0,(n+1)\*k);

en(Mx-2)=0.5\*r\*BoundaryConditions(1,n\*k)+0.5\*r\*BoundaryConditions(1,(n+1)\*k);

UcnM(:,n+1)=Cn\*UcnM(:,n)+en;

end

%将结果按Ku=f方法的u的结构重排

for t=1:Nt-1

Ucn= [Ucn;UcnM(:,t)];

end

%%

%计算精确解，用于误差分析\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

U\_M=zeros(Mx-2,Nt-1);

U=[];

for x=1:Mx-2

for t=1:Nt-1

U\_M(x,t)=ExactSolution(x\*h,t\*k);

end

end

%将结果按Ku=f方法的u的结构重排

for t=1:Nt-1

U=[U;U\_M(1:Mx-2,t)];

end

%%

%结果比较\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%比较标准误差（均方根误差）

temp=(U-Uxian).^2;

temp=sum(sum(temp));

temp=temp/(Mx\*Nt);

Exian=sqrt(temp);

temp=(U-Uyin).^2;

temp=sum(sum(temp));

temp=temp/(Mx\*Nt);

Eyin=sqrt(temp);

temp=(U-Ucn).^2;

temp=sum(sum(temp));

temp=temp/(Mx\*Nt);

Ecn=sqrt(temp);

fprintf('h: %d;k= %d\n',h,k);

fprintf('古典显格式标准误差：%d\n',Exian);

fprintf('古典隐格式标准误差：%d\n',Eyin);

fprintf('CN格式标准误差：%d\n',Ecn);

end