有限差分法解一维波动问题

钮博恒*1)

*（兰州大学物理科学与技术学院，兰州 730000）

摘要 波动是物理学中重要的模型，如波动光学，量子力学等，都归结于波动问题。波动这一简单的模型，展示出丰富的物理学内涵。为了描述波动，我们需要建立波动方程，解波动方程，最终得到场量u关于空间位置r和时间t的函数，空间维度可以是一维或更高维。波动方程的建立，需要借助力学手段，如最基本的一维弦的波动方程的导出，需要用到牛顿第二定律；电磁学中电磁波的导出，需要借助 Maxwell 方程组等。解波动方程，则可以得到场量u的表达式，即得到了场量的时空分布。本文利用有限差分法，解最简单的一维波动问题，并探究其他的一些性质。

关键词 波动方程，有限差分法，偏微分方程数值解法

FINITE DIFFERENCE METHOD FOR ONE-DIMENTIONAL WAVE EQUATION

Niu Boheng*1)

*(*School of Physical Science and Technology Lanzhou University*，*Lanzhou* 730000，*China*)

Abstract Waves are important models in physics, such as wave optics, quantum mechanics, etc., which are all due to wave problems. The simple model of wave shows rich physics connotation. In order to describe the wave, we need to set up the wave equation, solve the wave equation, and finally get the function of the field quantity u with respect to the space position and time. The space dimension can be one-dimensional or higher. The establishment of wave equation requires the use of mechanical means, such as the derivation of the most basic one-dimensional chord wave equation, which requires the use of Newton's second law; the derivation of electromagnetic waves in electromagnetics requires the use of Maxwell equations. By solving the wave equation, the expression of the field quantity u can be obtained, that is, the space-time distribution of the field quantity. This paper uses the finite difference method to solve the simplest one-dimensional wave problem and explores some other properties.

Key words Wave equation，Finite defference method，PDE

引 言

以下分别介绍一维波动方程数值解法的理论分析，其解法稳定性 Courant number 和 Magic time step 的讨论，边界条件，初始条件，叠加原理和干涉。

1 波动方程介绍和有限差分法理论公式

1.1 波动方程概述

沿着x轴传播的一维标量波方程可以表示为：

$$\frac{\partial^2 u(x,t)}{\partial t^2} = v^2 \frac{\partial^2 u(x,t)}{\partial x^2}$$

$u(x,t)$表示波函数，v是波速，对于电磁波，场量u是电场E或者磁场B，对于声波场量可能是气体的压强分布，对于一维弦，场量是弦在与传播方向垂直的方向的位移。

1.2 有限差分法推导

要做数值计算，需要先对变量进行离散化。对于时间有限的时间T，将其离散化为N_t个数值，步长为h_t。对于空间量x，将其离散化为N_x个数值，步长为h_x。那么，第n_t个时间单位的“真实”时间为：

$$t[n_t] = (n_t - 1)h_t \quad n_t = 1,2,3,\dots,N_t$$

第n_x个空间单位的的“真实”空间量为：

$$x[n_x] = (n_x - 1)h_x \quad n_x = 1,2,3,\dots,N_x$$

这样一来，场量$u(x,t)$就变为以索引n_x和n_t为格点的二维矩阵$u[n_x,n_t]$！那么原来的对时间和空间的二阶微分，皆可以用中心差商公式来代替：

$$\frac{\partial^2 u[n_x,n_t]}{\partial x^2} = \frac{u[n_x+1,n_t] - 2u[n_x,n_t] + u[n_x-1,n_t]}{h_x^2}$$

$$\frac{\partial^2 u[n_x,n_t]}{\partial t^2} = \frac{u[n_x,n_t+1] - 2u[n_x,n_t] + u[n_x,n_t-1]}{h_t^2}$$

因为是中心差商公式，第一个和最后一个点都不能取到，所以要求：

$$2 \le n_x \le N_x - 1$$

$$2 \le n_t \le N_t - 1$$

将原方程中的二阶偏导数都用差商公式代替，可以得到最终的**递推格式**：

$$u[n_x,n_t+1] = 2u[n_x,n_t] - u[n_x,n_t-1] + \left(\frac{v^2 h_t^2}{h_x^2}\right)(u[n_x+1,n_t] - 2u[n_x,n_t] + u[n_x-1,n_t])$$

可以看到，如果想要得到n_{t+1}时刻，位置n_x处的场量值，需要n_t时刻，n_x，n_{x+1}，n_{x-1}处的数值，以及n_{t-1}时刻，n_x处的数值。这是一个完全**显式**的表达式，因为等式右边的所有值，都依靠之前时刻的值取到。为了能够启动算法，至少需要最初的两个时刻的各点处的场量值，也就是初始条件。

$$n_x = 1,2,3,\dots,N_x$$

$$n_t = 1 \quad t = t[1] = 0 \quad u[n_x,1]$$

$$n_t = 2 \quad t = t[2] = h_t \quad u[n_x,2]$$

对于第一、二类边界条件，可以分别得到：

$$u[1,n_t] = u[2,n_t] \quad u[N_x,n_t] = u[N_x-1,n_t]$$

2 方波和*Guassian*波包的例子

```
clear all;
clc;

%初始化设置
% 空间和时间格点数
  N_x = 100;
% 时间步数
  N_t = 200;
% 区域大小
   L = 100;
%波速
   v = 10;
% 稳定性
   S = 1;
% 两种形状的波包，1为Guassian波包，2为方波
   flag = 1;
% 设置Guassian波包
   A = 0.8;          % 波峰
   s = 2.5;          % 波宽
   n_start = 50;       % 起始位置
% 初始化波形和边界条件
   u = zeros(N_x,N_t);
% 设置步长
   hx = L / N_x;
```

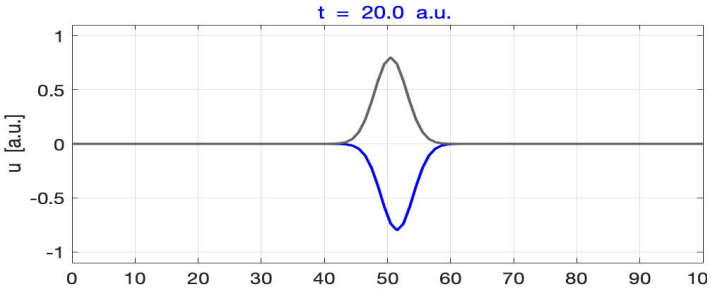
```
   ht = S * hx / v;
   S2 = S^2;
   x = linspace(0, L , N_x);
   t = linspace(0,ht*N_t,N_t);

if flag == 1    % 1表示Guassian波包

   for n_x = 2 : N_x-1  % 第一个时刻
          u(n_x,1) = A .* exp(-0.5 .*
((n_x-n_start)./s).^2);
   end

   for n_x = 2 : N_x - 1   % 第二个时刻
     u(n_x,2) = A .* exp(-0.5 .* ((n_x-
n_start-1)./s).^2);
   end
end

if flag == 2% 2表示方波
   u(10:30,1) = 0.5; % 第一个时刻
   u(11:31,2) = 0.5; % 第二个时刻
end
% 递推格式！
for n_t = 2 : N_t-1
    for n_x = 2 : N_x-1
       u(n_x,n_t+1) = 2*u(n_x,n_t)-
u(n_x,n_t-1)+ S2*(u(n_x+1,n_t) -
2*u(n_x,n_t) + u(n_x-1,n_t));
    end
    % 边界条件
      u(1, n_t+1)  = 0;              % 首端固定
      u(N_x,n_t+1) = u(N_x-1,n_t+1);   % 末端
自由
    end
```

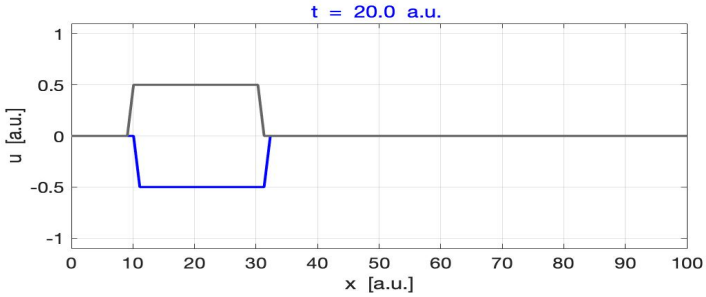
以上就是利用有限差分法模拟计算一维波动的主要代码，无论波形如何，只要改变初始条件的设置即可，对于递推格式，都是不变的。所以我们用第一个标记$flag = 1$表示$Guassian$波包，用第二个标记$falg = 2$表示方波，在最初的时候选择某一个标记，之后程序就根据两个判断语句来决定使用哪一种波形。接下来的代码只是绘制不同时刻波形的子图，以及最终时刻的状态，和动图绘制。绘制子图的代码冗长重复，这里只给出第一个子图的代码。

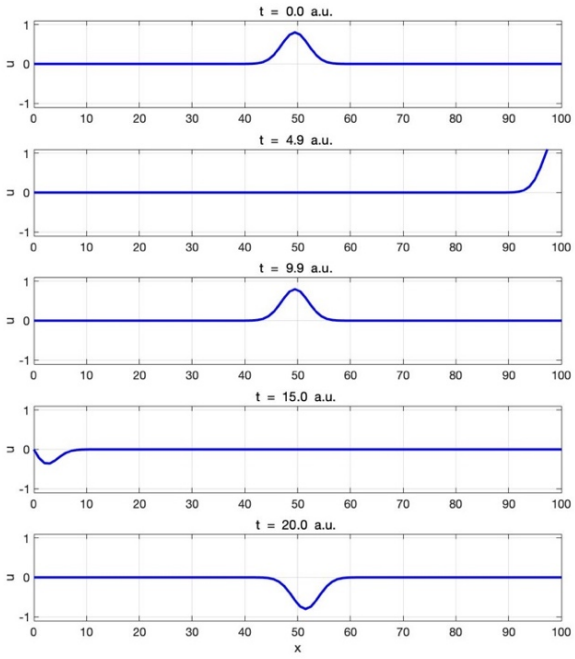
```
   figure(1) % 绘制五个子图
      col = 'b'; LW = 2;
      tm1 = 't  =  ';
      tm3 = '  a.u. ';
      subplot(5,1,1)
        xP = x; yP = u(:,1)';
        plot(xP, yP,col,'linewidth',LW);
        axis([0 L -1.1 1.1])
        tm2 = num2str(t(1));
        tm = [tm1 tm2 tm3];
        title(tm);
        grid on; box on; % 显示网格
       ylabel('u');
   figure(2) % 制作最终时刻的图
      grid on %显示网格
      col = 'b'; LW = 2;
      xP = x; yP = u(:,N_t);
      plot(xP,
yP,'color',col,'linewidth',LW);
      hold on
      col = 'r'; LW = 2;
      xP = x; yP = u(:,2);
      plot(xP,
yP,'color',col,'linewidth',LW);

      axis([0 L -1.1 1.1])%限制坐标轴
      tm1 = 't  =  ';
      tm2 = num2str(N_t);
      tm3 = '  a.u. ';
      tm = [tm1 tm2 tm3];
      title(tm);
      xlabel('x  [a.u.]'); ylabel('u
[a.u.]');
      grid on
      figure(3) % 制作动画
      for n_t = 1: N_t
      xP = x; yP = u(:,N_t);

plot(xP,yP,'color','b','linewidth',2);
      axis([0 L -1.1 1.1])
      grid on
      xlabel('x  [a.u]'); ylabel('u
[a.u.]');
      tm2 = num2str(t(n_t));
      tm = [tm1 tm2 tm3];
      title(tm);
      pause(0.001);
      end
```

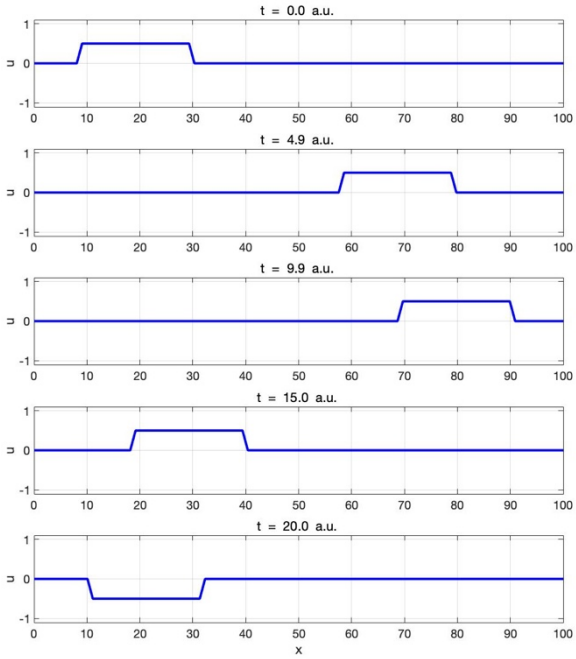
以上是依据有限差分法递推格式解一维波动方程的所有代码，之后的讨论**只对其中的一小部分进行修改**，作图和显示波动效果的代码均不变。

代码中给出了两个不同形状的波包，分别是方波和$Guassian$波包，运行结果如下：



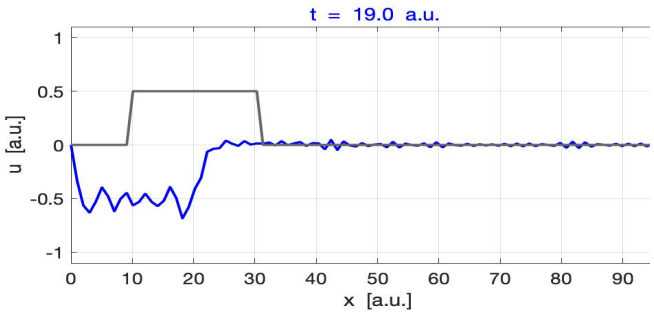


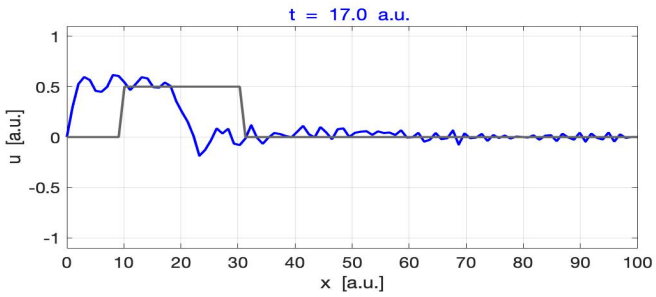




3 稳定性讨论

实际上，以上示例默认设置了$S=1$，$S=\frac{v\Delta t}{\Delta x}$，课程上讲过，也它是表征算法稳定性的重要参数。一般认为时间间隔较小时算法更稳定，也就是说$S\leq 1$时算法可以稳定，而当$S=1$时，容易得 到 $v\Delta t=\Delta x$，也就是说，数值计算得到的是就是实际结果，而不是计算近似，这和上面的模拟得到的结果相符。将此时的Δt解出，也就是$\Delta t=\frac{\Delta x}{v}$那么当$S$不为 1 时，结果会如何呢？





以上的四张图分别显示了当$S=0.95$和$S=0.85$时的波形图。可以看出，当S一旦小于 1，算法的稳定性

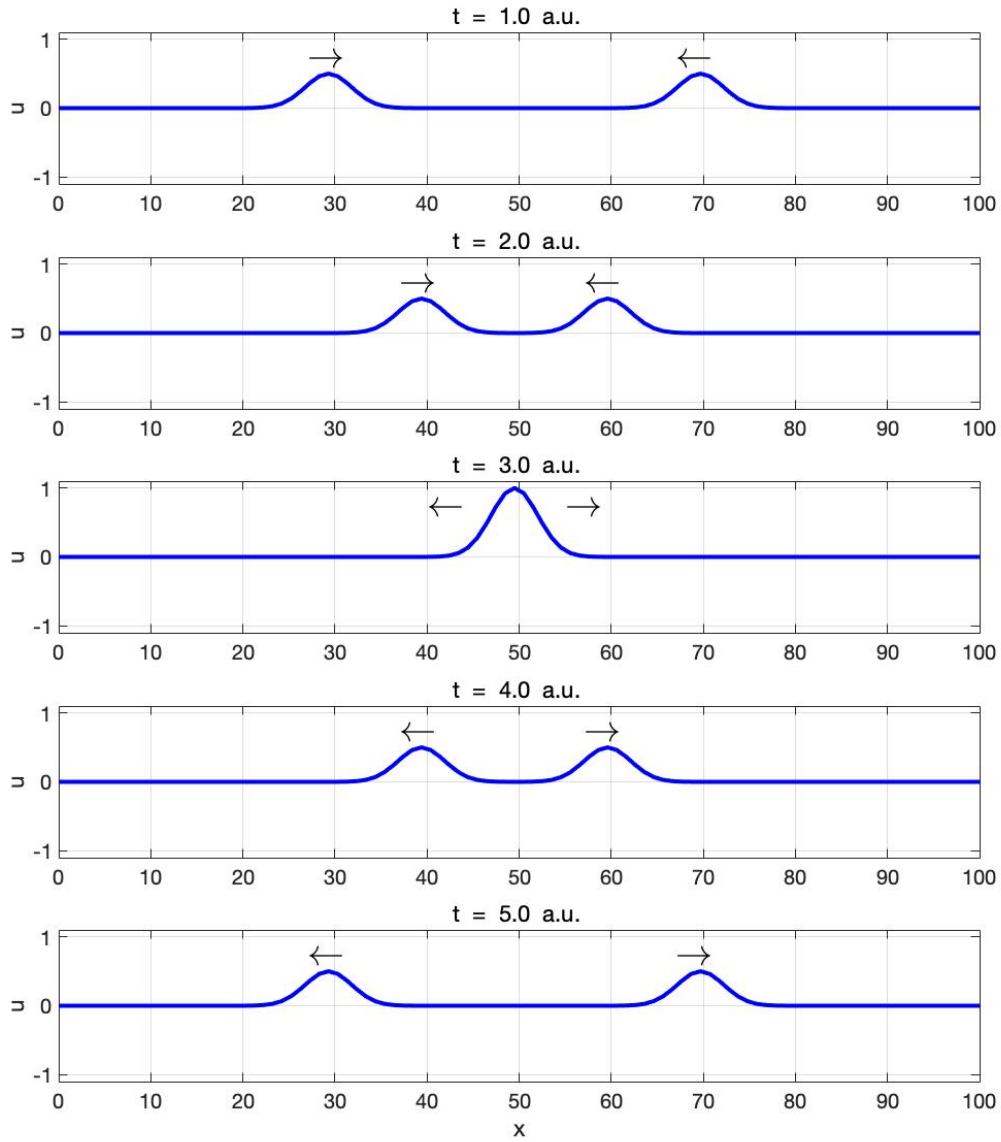
就被破坏。

3 叠加原理和干涉现象

希望实现两个波包从两头相向而行，并观察它们相遇时的情形，方法非常简单，只需要在初始化波包时，在波包的表达式后再加一项即可。

```
u(n_x,1) = A .* exp(-0.5 .* ((n_x-20)./s).^2) ...
+ A .* exp(-0.5 .* ((n_x-80)./s).^2);
```

其他代码不做任何改变，结果如下图，可以很清楚地证实波的叠加原理，看到干涉现象。



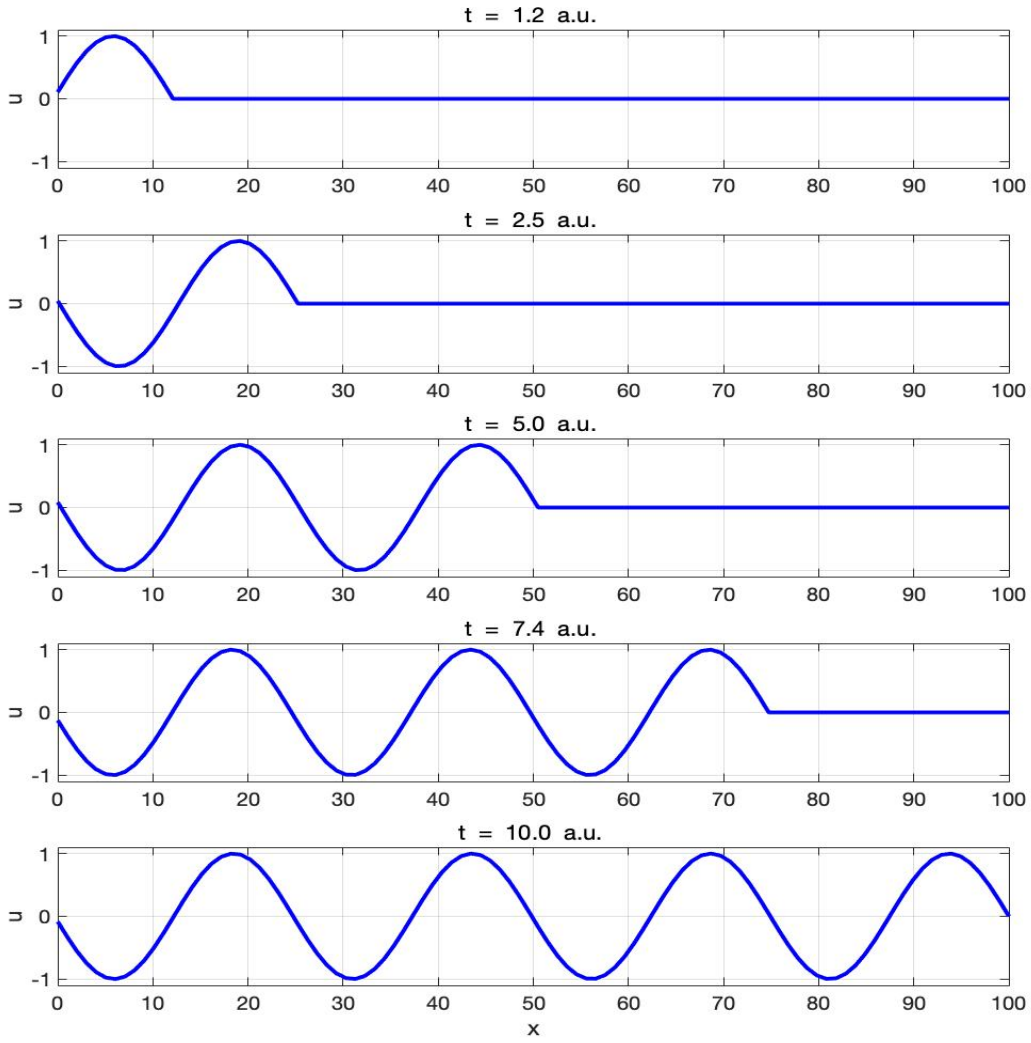
4 行波

要实现行波，同样需要修改初始设置的代码，但是为了没有波形达到边界后的干扰（反射），既不能使用两端固定的边界条件，也不能使用一段自由的边界条件，我们使用：

$$u(Nx,nt+1) = u(Nx-1,nt)$$

这样的话就保证了在有限的区域内，波形不反射，对之前波形的干扰尽可能的小[3]。修改的初始条件和边界条件分别如下；

```
    u(1,:) = A .* sin((2*pi/T).* t);
    for n_t = 2 : N_t-1% 递推格式不做修改
    for n_x = 2 : N_x-1
       u(n_x,n_t+1) = 2*u(n_x,n_t)- u(n_x,n_t-1)+ S2*(u(n_x+1,n_t) - 2*u(n_x,n_t) + u(n_x-
1,n_t));
    end
    % 修改边界条件
    u(Nx,nt+1) = u(Nx-1,nt); end
```



参考文献

1 彭芳麟，计算物理基础，北京，高等教育出版社 2010

2 Finite-Difference numerical method of partial differential equations in finance with MATLAB, Aitor Brgara

3 Doing Physics with MATLAB, School of Physics, University of Sydney