Lesson 22: Partial Differential Equations

1 Introduction

Partial differential equations (PDE's) describe systems with more than one independent variable. Many examples in physics involve wave motion, such as water waves, electromagnetic waves, sound waves, *etc*. In each of these cases, the dependent variable is the amplitude of the wave and the independent variables are time t and the space coordinates. Other examples of PDE's include the Schrödinger equation, which describes the quantum mechanical wave function, the Poisson equation, which describes the electrostatic and gravitational potentials, and the heat equation, which describes the flow of thermal energy through a body.

2 Waves on a string

Consider a string, streched along the x–axis, under tension T. The mass per unit length of the string is μ. We will ignore gravity and consider only small amplitude, transverse waves. That is, we only allow small displacements of the string away from the x axis in, say, the y direction. Our goal is to describe the displacement amplitude $y(t, x)$ as a function of the two independent variables t and x.

Figure (1) shows a short segment of the string. The forces on the right and left ends of the segment depend on the tension T and the angles θ_R, θ_L that the ends make with the x direction. The net force in the y direction is

$$F_y = T \sin\theta_R - T \sin\theta_L \ . \tag{1}$$

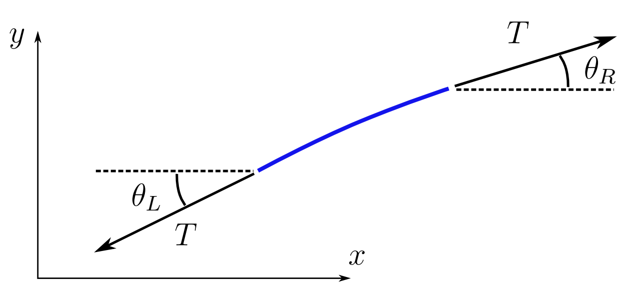

Figure 1: Segment of string under tension T.

By applying Newton's second law to the string segment, we find

$$\mu \Delta x \frac{d^2 y}{dt^2} = T(\sin\theta_R - \sin\theta_L) \ , \tag{2}$$

where Δx is the length of the segment (assuming displacements are small) and $\mu\Delta x$ is the segment's mass. For small displacements of the string the angles θ_R and θ_L will be small. To a good approximation the sin's can be replaced with tan's. But the tangent of θ_R is the slope dy/dx at the right end of the segment. Likewise, the tangent of θ_L is the slope at the left end of the segment. Thus we have

$$\mu \frac{d^2 y}{dt^2} = \frac{T}{\Delta x}\left(\left.\frac{dy}{dx}\right|_R - \left.\frac{dy}{dx}\right|_L \right) \ . \tag{3}$$

In the limit as the segment size shrinks to zero, $\Delta x \to 0$, the right–hand side is proportional to the second derivative d^2y/dx^2. Therefore the PDE that governs small–amplitude wave motion on a string is

$$\ddot{y} = c^2 y'' \ , \tag{4}$$

where we have defined $c^2 \equiv T/\mu$. (Dots denote derivatives with respect to t and primes denote derivatives with respect to x.) This equation is often called "the wave equation".

It is easy to verify that the wave equation has solutions of the form

$$y(t,x) = F(x+ct) + G(x-ct) \ , \tag{5}$$

where F and G are arbitrary functions of their arguments. In fact, Eq. (5) is the general solution. Any solution can be described as the superposition of a left–moving contribution $F(x+ct)$ and a right–moving contribution $G(x-ct)$. A purely left–moving wave $y(t,x) = F(x+ct)$ satifies $y(t+T, x-cT) = y(t,x)$; this wave shifts to the left by an amount cT during time T. Likewise, a purely right–moving wave $y(t,x) = G(x-ct)$ satisfies $y(t+T, x+cT) = y(t,x)$; this wave shifts to the right by an amount cT during time T. The constant c is the *wave speed.*

3 Finite difference schemes

A PDE such as (4) can be solved numerically through finite differencing. Let us assume that the ends of the string are held fixed at locations x_a, x_b on the x–axis. Divide the spatial interval $x_a \le x \le x_b$ into N segments, each of length $\Delta x = (x_b - x_a)/N$. This defines $N+1$ *nodes* at locations $x_0, x_1, \ldots x_N$, where $x_i = x_a + i\Delta x$. We can also divide the time interval $0 \le t \le \infty$ into "timesteps" $t^0, t^1, \ldots$ of length Δt, where $t^n = n\Delta t$. Now let $y_i^n = y(t^n, x_i)$ denote the amplitude of the string at spatial location x_i and time t^n. The finite difference approximations to the second derivatives of y are

$$\ddot{y}(t^n, x_i) \approx \frac{y_i^{n+1} - 2y_i^n + y_i^{n-1}}{\Delta t^2} \ , \tag{6a}$$

$$y''(t^n, x_i) \approx \frac{y_{i+1}^n - 2y_i^n + y_{i-1}^n}{\Delta x^2} \ . \tag{6b}$$

Inserting these approximations into the PDE (4) and solving for y_i^{n+1}, we find

$$y_i^{n+1} = 2y_i^n - y_i^{n-1} + \left(\frac{c\Delta t}{\Delta x}\right)^2 (y_{i+1}^n - 2y_i^n + y_{i-1}^n) \ . \tag{7}$$

If we know the amplitudes at each node at times t^0 and t^1, we can use Eq. (7) to predict the amplitudes at nodes $i = 1, \ldots, N-1$ at time t^2. Since the endpoints of the string are fixed, the amplitudes at the endpoints are given by $y_0^2 = y_0^1$ and $y_N^2 = y_N^1$. Having found all of the amplitudes at time t^2, we can iterate this proceedure to obtain the amplitudes at t^3, t^4, *etc.*

Homework: Write a code to solve the wave equation using the finite difference scheme (7). Choose a mass density of $\mu = 0.01\,\mathrm{kg/m}$ and a tension of $T = 25.0\,\mathrm{N}$; this yields the wave speed $c = 50.0\,\mathrm{m/s}$. Let $x_a = -5.0\,\mathrm{m}$ and $x_b = 5.0\,\mathrm{m}$ be the endpoints locations. Use $N = 100$ for your initial tests; this determines Δx. For the timestep Δt you need to choose a value that is small relative to $\Delta x/c$. We will discuss this later. For now, take $\Delta t = 0.25\Delta x/c$. For the amplitude at the initial time $t = 0$, use a Gaussian pulse

$$y_i^0 = Ae^{-x_i^2/\sigma^2} \ . \tag{8}$$

You will need to choose reasonable values for the constants A and σ. Make sure σ is sufficiently small so that y_0^0 and y_N^0 are approximately zero.
The finite difference scheme (7) requires us to specify the amplitude at the first timestep, y_i^1, before we can compute the amplitudes at future timesteps. For now take $y_i^1 = y_i^0$. This is *approximately* equivalent to setting the initial velocity of the string to zero, since $\dot{y}(0, x_i) \approx (y_i^1 - y_i^0)/\Delta t$ is the initial velocity at node i.
Graph y versus x at various times, say, $t = 0.01\,\mathrm{s}$, $0.02\,\mathrm{s}$, *etc.* Use your numerical data to verify that the wave speed is $c = 50.0\,\mathrm{m/s}$.

4 Numerical stability

Exercise 1: Change the timestep interval to $\Delta t = 1.25\Delta x/c$. You should find that your code works well for $\Delta t = 0.25\Delta x/c$, but fails for $\Delta t = 1.25\Delta x/c$. Determine the number η such that the code fails whenever $\Delta t \geq \eta\Delta x/c$.

As a general rule, a finite difference scheme will become *numerically unstable* if the timestep Δt is too large. That is, the code will yield physically meaningful results only if the timestep satisfies $\Delta t \leq \eta\Delta x/c$, where the number η is typically of order unity. This condition can be written as $\eta\Delta x/\Delta t \geq c$. Roughly speaking, the ratio $\eta\Delta x/\Delta t$ represents the speed at which the numerical algorithm can transmit information across the grid. If this speed is smaller than the physical speed c, the code becomes unstable and the numerical results are unphysical.

The restriction $\Delta t \leq \eta\Delta x/c$ is called the *Courant* (or *Courant–Friedrichs–Lewy*) condition. The value of η depends on the details of the PDE and the details of the numerical algorithm. If η is infinite, the numerical scheme is *unconditionally stable*. In these cases the scheme is stable (but not necessarily accurate) for arbitrarily large Δt. If η is zero, the numerical scheme is *unconditionally unstable*. In these cases the scheme is unstable for all Δt.

We can modify Eq. (7) as follows:

$$y_i^{n+1} = 2y_i^n - y_i^{n-1} + \left(\frac{c\Delta t}{\Delta x}\right)^2 (y_{i+1}^{n-1} - 2y_i^{n-1} + y_{i-1}^{n-1}) \ . \tag{9}$$

This scheme, like (7), is *consistent* in the sense that is yields the wave equation (4) in the limit as Δt and Δx go to zero.

Exercise 2: Modify your code for the wave equation to use the finite difference scheme (9). Investigate the stability of this scheme by experimenting with different timesteps Δt. You might need to run your code up to, say, $t = 0.5$ to see the long–term behavior.

5 First order system

The wave equation (4) contains second order time derivatives. It can be rewritten as a system of equations with first order time derivatives by introducing a new variable $v = \dot{y}$ to represent the transverse velocity of the string. Thus, the system

$$\dot{y} = v \ , \tag{10a}$$

$$\dot{v} = c^2 y'' \tag{10b}$$

is equivalent to the single equation (4). Now, a finite difference approximation to the first time derivative of y is

$$\dot{y}(t^n, x_i) \approx \frac{y_i^{n+1} - y_i^n}{\Delta t} \ . \tag{11}$$

A similar approximation can be made for $\dot{v}$. Using these results and the approximation (6b) for y'', we find

$$y_i^{n+1} = y_i^n + \Delta t\, v_i^n \ , \tag{12a}$$

$$v_i^{n+1} = v_i^n + \Delta t\, c^2 \left(\frac{y_{i+1}^n - 2y_i^n + y_{i-1}^n}{\Delta x^2}\right) \ . \tag{12b}$$

This is called the *forward time–centered space* (FTCS) discretization of the system (10). Can you guess where this name comes from?

The finite difference equations (12) are easy to implement. Begin by choosing initial amplitudes y_i^0 and velocities v_i^0. Next, apply Eqs. (12) to determine y_i^1 and v_i^1 at the interior nodes $i = 1, \ldots, N-1$. At the endpoints, the amplitudes stay fixed ($y_0^{n+1} = y_0^n$ and $y_N^{n+1} = y_N^n$) and the velocities are zero ($v_0^{n+1} = 0$ and $v_N^{n+1} = 0$). Iterate to obtain the amplitudes and velocities at t^2, t^3, *etc.* One of the advantages of using the first order algorithm (12) rather than the second order algorithm (7), is that the initial conditions now have a direct physical interpretation as the initial string amplitude and velocity.

Exercise 3: Write a code for the wave equation using the FTCS scheme (12). Investigate stability by experimenting with different timesteps Δt. For initial conditions, use a Gaussian pulse (8) and $v_i^0 = 0$.

As you should now realize, the FTCS algorithm is not useful. There are other numerical algorithms that are much better. Consider the following two–step scheme:

$$\begin{aligned}
\tilde{y}_i &= y_i^n + \frac{\Delta t}{2} v_i^n \ , && (13a)\\
\tilde{v}_i &= v_i^n + \frac{\Delta t}{2} c^2 \left(\frac{y_{i+1}^n - 2y_i^n + y_{i-1}^n}{\Delta x^2} \right) \ , && (13b)\\
y_i^{n+1} &= y_i^n + \Delta t\, \tilde{v}_i \ , && (13c)\\
v_i^{n+1} &= v_i^n + \Delta t\, c^2 \left(\frac{\tilde{y}_{i+1} - 2\tilde{y}_i + \tilde{y}_{i-1}}{\Delta x^2} \right) \ . && (13d)
\end{aligned}$$

The calculation in the first step, Eqs. (13a), (13b), yields approximations $\tilde{y}_i$, $\tilde{v}_i$ to the amplitudes and velocities at time $t^n + \Delta t/2$ half way between t^n and t^{n+1}. The second step, Eqs. (13c), (13d), uses the amplitudes and velocities at the half–timestep to approximate the right–hand sides of the differential equations (10). This algorithm is closely related to the second–order Runge–Kutta method for ordinary differential equations.

Homework: Write a code to solve the wave equation based on the algorithm in Eqs. (13). Use the same initial data as in the previous exercises. Determine the Courant condition using numerical experiments. Plot a graph of y versus x at various times.

Recall that the general solution (5) of the wave equation is a linear combination of left and right–moving waves. Consider, in particular, a right–moving wave formed from a Gaussian function:

$$y(t, x) = Ae^{-(x-ct)^2/\sigma^2} \ . \qquad (14)$$

The initial amplitude and velocity for this solution is

$$\begin{aligned}
y(0, x) &= Ae^{-x^2/\sigma^2} \ , && (15a)\\
v(0, x) &= (2Acx/\sigma^2)e^{-x^2/\sigma^2} \ , && (15b)
\end{aligned}$$

where $v(t, x) \equiv \dot{y}(t, x)$.

Homework: Modify your code to use the initial velocities $v_i^0 \equiv v(0, x_i)$ from Eq. (15b). (The initial amplitudes (15a) coincide with the initial amplitudes from the previous exercises.) Plot a graph of y versus x at various times. Do the data evolve in the way you expect? What happens when the pulse hits the end of the string?