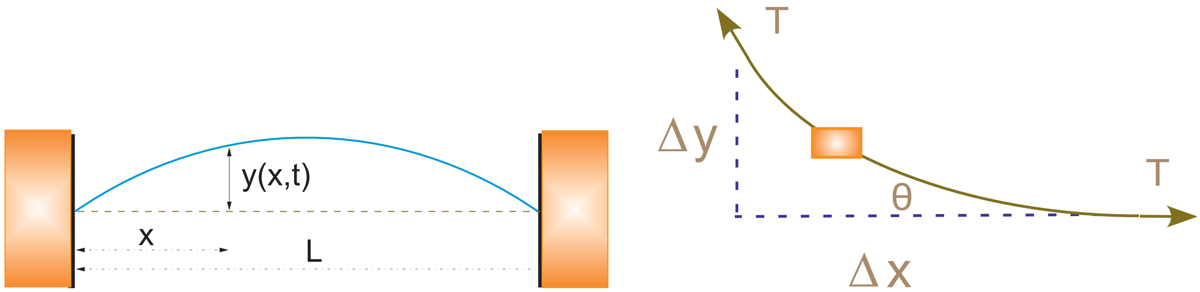
Figure 18.1 *Left:* A stretched string of length L tied down at both ends and under high enough tension to ignore gravity. The vertical disturbance of the string from its equilibrium position is $y(x,t)$. *Right:* A differential element of the string showing how the string's displacement leads to the restoring force.



18.2 THE HYPERBOLIC WAVE EQUATION (THEORY)

Consider a string of length L tied down at both ends (Figure 18.1 left). The string has a constant density ρ per unit length, a constant tension T, no frictional forces acting on it, and a tension that is so high that we may ignore sagging due to gravity. We assume that displacement of the string $y(x,t)$ from its rest position is in the vertical direction only and that it is a function of the horizontal location along the string x and the time t.

To obtain a simple linear equation of motion (nonlinear wave equations are discussed in Chapter 19, "Solitons & Computational Fluid Dynamics"), we assume that the string's relative displacement $y(x,t)/L$ and slope $\partial y/\partial x$ are small. We isolate an infinitesimal section Δx of the string (Figure 18.1 right) and see that the difference in the vertical components of the tension at either end of the string produces the restoring force that accelerates this section of the string in the vertical direction. By applying Newton's laws to this section, we obtain the familiar wave equation:

$$\sum F_y = \rho\, \Delta x\, \frac{\partial^2 y}{\partial t^2}, \tag{18.1}$$

$$\sum F_y = T\sin\theta(x+\Delta x) - T\sin\theta(x) = T\left.\frac{\partial y}{\partial x}\right|_{x+\Delta x} - T\left.\frac{\partial y}{\partial x}\right|_{x} \simeq T\frac{\partial^2 y}{\partial x^2},$$

$$\Rightarrow \quad \frac{\partial^2 y(x,t)}{\partial x^2} = \frac{1}{c^2}\frac{\partial^2 y(x,t)}{\partial t^2}, \qquad c = \sqrt{\frac{T}{\rho}}, \tag{18.2}$$

where we have assumed that θ is small enough for $\sin\theta \simeq \tan\theta = \partial y/\partial x$. The existence of two independent variables x and t makes this a PDE. The constant c is the velocity with which a disturbance travels along the wave and is seen to decrease for a heavier string and increase for a tighter one. Note that this signal velocity c is *not* the same as the velocity of a string element $\partial y/\partial t$.

The initial condition for our problem is that the string is plucked gently and released. We assume that the "pluck" places the string in a triangular shape with the center of triangle $\frac{8}{10}$ of the way down the string and with a height of 1:

$$y(x,t=0) = \begin{cases} 1.25x/L, & x \le 0.8L, \\ (5-5x/L), & x > 0.8L, \end{cases} \quad \text{(initial condition 1)}. \tag{18.3}$$

Because (18.2) is second-order in time, a second initial condition (beyond initial displacement) is needed to determine the solution. We interpret the "gentleness" of the pluck to mean the string is released from rest:

$$\frac{\partial y}{\partial t}(x, t=0) = 0, \quad \text{(initial condition 2).} \tag{18.4}$$

The boundary conditions have both ends of the string tied down for all times:

$$y(0,t) \equiv 0, \quad y(L,t) \equiv 0, \quad \text{(boundary conditions).} \tag{18.5}$$

18.2.1 Solution via Normal-Mode Expansion

The analytic solution to (18.2) is obtained via the familiar separation-of-variables technique. We assume that the solution is the product of a function of space and a function of time:

$$y(x,t) = X(x)T(t). \tag{18.6}$$

We substitute (18.6) into (18.2), divide by $y(x,t)$, and are left with an equation that has a solution only if there are solutions to the two ODEs:

$$\frac{d^2T(t)}{dt^2} + \omega^2 T(t) = 0, \quad \frac{d^2X(x)}{dt^2} + k^2 X(x) = 0, \quad k \stackrel{\text{def}}{=} \frac{\omega}{c}. \tag{18.7}$$

The angular frequency ω and the wave vector k are determined by demanding that the solutions satisfy the boundary conditions. Specifically, the string being attached at both ends demands

$$X(x=0,t) = X(x=l,t) = 0 \tag{18.8}$$

$$\Rightarrow \quad X_n(x) = A_n \sin k_n x, \quad k_n = \frac{\pi(n+1)}{L}, \quad n = 0, 1, \ldots. \tag{18.9}$$

The time solution is

$$T_n(t) = C_n \sin \omega_n t + D_n \cos \omega_n t, \quad \omega_n = nck_0 = n\frac{2\pi c}{L}, \tag{18.10}$$

where the frequency of this nth *normal mode* is also fixed. In fact, it is the single frequency of oscillation that defines a normal mode. The *initial condition* (18.3) of zero velocity, $\partial y/\partial t(t = 0) = 0$, requires the C_n values in (18.10) to be zero. Putting the pieces together, the normal-mode solutions are

$$y_n(x,t) = \sin k_n x \cos \omega_n t, \quad n = 0, 1, \ldots. \tag{18.11}$$

Since the wave equation (18.2) is linear in y, the principle of linear superposition holds and the most general solution for waves on a string with fixed ends can be written as the sum of normal modes:

$$y(x,t) = \sum_{n=0}^{\infty} B_n \sin k_n x \cos \omega_n t. \tag{18.12}$$

(Yet we will lose linear superposition once we include nonlinear terms in the wave equation.) The Fourier coefficient B_n is determined by the second initial condition (18.3), which describes how the wave is plucked:

$$y(x, t=0) = \sum_{n}^{\infty} B_n \sin nk_0 x. \tag{18.13}$$

Multiply both sides by $\sin mk_0 x$, substitute the value of $y(x,0)$ from (18.3), and integrate from 0 to l to obtain

$$B_m = 6.25 \, \frac{\sin(0.8m\pi)}{m^2\pi^2}. \tag{18.14}$$

You will be asked to compare the Fourier series (18.12) to our numerical solution. While it is in the nature of the approximation that the precision of the numerical solution depends on the choice of step sizes, it is also revealing to realize that the precision of the analytic solution depends on summing an infinite number of terms, which can be done only approximately.

18.2.2 Algorithm: Time-Stepping

As with Laplace's equation and the heat equation, we look for a solution $y(x,t)$ only for discrete values of the independent variables x and t on a grid (Figure 18.2):

$$x = i\Delta x, \quad i = 1,\ldots,N_x, \quad t = j\Delta t, \quad j = 1,\ldots,N_t, \tag{18.15}$$

$$y(x,t) = y(i\Delta x,\ i\Delta t) \stackrel{\text{def}}{=} y_{i,j}. \tag{18.16}$$

In contrast to Laplace's equation where the grid was in two space dimensions, the grid in Figure 18.2 is in both space and time. That being the case, moving across a row corresponds to increasing x values along the string for a fixed time, while moving down a column corresponds to increasing time steps for a fixed position. Even though the grid in Figure 18.2 may be square, we cannot use a relaxation technique for the solution because we do not know the solution on all four sides. The boundary conditions determine the solution along the right and left sides, while the initial time condition determines the solution along the top.

As with the Laplace equation, we use the central-difference approximation to *discretize* the wave equation into a difference equation. First we express the second derivatives in terms of finite differences:

$$\frac{\partial^2 y}{\partial t^2} \simeq \frac{y_{i,j+1} + y_{i,j-1} - 2y_{i,j}}{(\Delta t)^2}, \quad \frac{\partial^2 y}{\partial x^2} \simeq \frac{y_{i+1,j} + y_{i-1,j} - 2y_{i,j}}{(\Delta x)^2}. \tag{18.17}$$

Substituting (18.17) in the wave equation (18.2) yields the difference equation

$$\frac{y_{i,j+1} + y_{i,j-1} - 2y_{i,j}}{c^2(\Delta t)^2} = \frac{y_{i+1,j} + y_{i-1,j} - 2y_{i,j}}{(\Delta x)^2}. \tag{18.18}$$

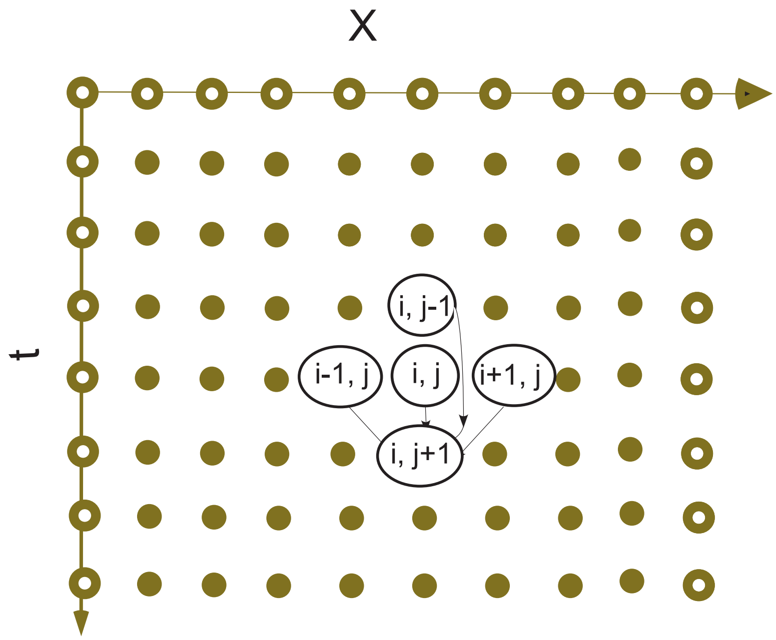
Notice that this equation contains three time values: `j+1` = the future, `j` = the present, and `j-1` = the past. Consequently, we rearrange it into a form that permits us to predict the future solution from the present and past solutions:

$$\boxed{y_{i,j+1} = 2y_{i,j} - y_{i,j-1} + \frac{c^2}{c'^2}\left[y_{i+1,j} + y_{i-1,j} - 2y_{i,j}\right],} \quad c' \stackrel{\text{def}}{=} \frac{\Delta x}{\Delta t}. \tag{18.19}$$

Here c' is a combination of numerical parameters with the dimension of velocity whose size relative to c determines the stability of the algorithm. The algorithm (18.19) propagates the wave from the two earlier times, j and $j-1$, and from three nearby positions, $i-1$, i, and $i+1$, to a later time $j+1$ and a single space position i (Figure 18.2).

As you have seen in our discussion of the heat equation, a leapfrog method is quite different from a relaxation technique. We start with the solution along the topmost row and then move down one step at a time. If we write the solution for present times to a file, then we need to store only three time values on the computer, which saves memory. In fact, because the time steps must be quite small to obtain high precision, you may want to store the solution only for every fifth or tenth time.

Figure 18.2 The solutions of the wave equation for four earlier space-time points are used to obtain the solution at the present time. The boundary and initial conditions are indicated by the white-centered dots.



Initializing the recurrence relation is a bit tricky because it requires displacements from two earlier times, whereas the initial conditions are for only one time. Nonetheless, the rest condition (18.3) when combined with the *central-difference* approximation lets us extrapolate to negative time:

$$\frac{\partial y}{\partial t}(x,0) \simeq \frac{y(x,\Delta t) - y(x,-\Delta t)}{2\Delta t} = 0, \quad \Rightarrow \quad y_{i,0} = y_{i,2}. \tag{18.20}$$

Here we take the initial time as $j = 1$, and so $j = 0$ corresponds to $t = -\Delta t$. Substituting this relation into (18.19) yields for the initial step

$$y_{i,2} = y_{i,1} + \frac{c^2}{2c'^2}\left[y_{i+1,1} + y_{i-1,1} - 2y_{i,1}\right] \quad (\text{for } j = 2 \text{ only}). \tag{18.21}$$

Equation (18.21) uses the solution throughout all space at the initial time $t = 0$ to propagate (leapfrog) it forward to a time Δt. Subsequent time steps use (18.19) and are continued for as long as you like.

As is also true with the heat equation, the success of the numerical method depends on the relative sizes of the time and space steps. If we apply a von Neumann stability analysis to this problem by substituting $y_{m,j} = \xi^j \exp(ikm\,\Delta x)$, as we did in § 17.17.3, a complicated equation results. Nonetheless, [Pres 94] shows that the difference-equation solution will be stable for the general class of transport equations if

$$c \le c' = \Delta x/\Delta t \quad (\text{Courant condition}). \tag{18.22}$$

Equation (18.22) means that the solution gets better with smaller *time* steps but gets worse for smaller space steps (unless you simultaneously make the time step smaller). Having different sensitivities to the time and space steps may appear surprising because the wave equation (18.2) is symmetric in x and t, yet the symmetry is broken by the nonsymmetric initial and boundary conditions.

Exercise: Figure out a procedure for solving for the wave equation for all times in just one step. Estimate how much memory would be required. ▮

Exercise: Try to figure out a procedure for solving for the wave motion with a relaxation technique. What would you take as your initial guess, and how would you know when the procedure has converged? ▮

18.2.3 Wave Equation Implementation

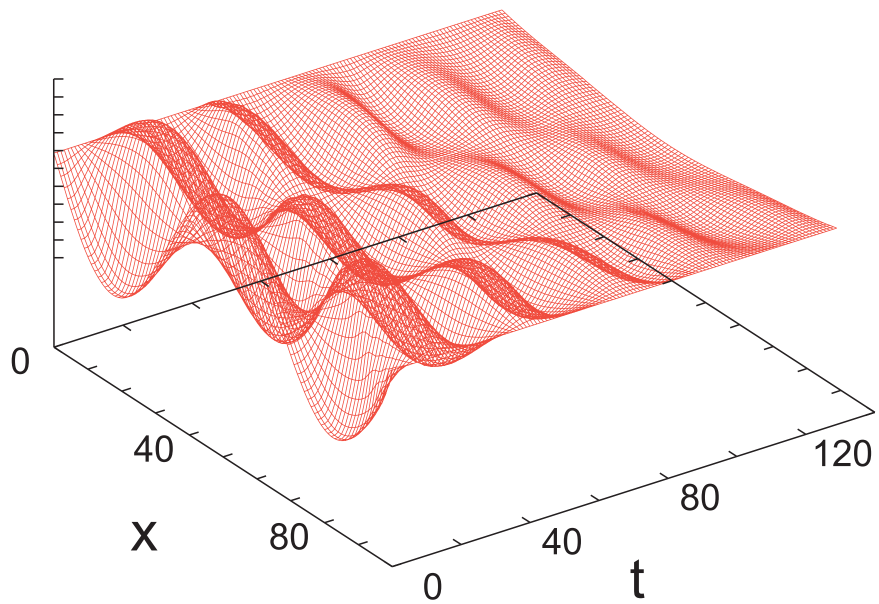
The program `EqString.py` in Listing 18.1 solves the wave equation for a string of length $L = 1$ m with its ends fixed and with the gently plucked initial conditions. Note that our use of $L = 1$ violates our assumption that $y/L \ll 1$ but makes it easy

Listing 18.1 **EqString.py** solves the wave equation via time stepping for a string of length $L = 1$ m with its ends fixed and with the gently plucked initial conditions. You will need to modify this code to include new physics.

```
# EqString.py:          Animated leapfrog solution of wave equation (Sec 18.2.2)


from visual import *

# Set up curve
g = display(width = 600, height = 300, title = 'Vibrating string')
vibst = curve(x = range(0, 100), color = color.yellow)
ball1 = sphere(pos = (100, 0), color = color.red, radius = 2)
ball2 = sphere(pos = ( - 100, 0), color = color.red, radius = 2)
ball1.pos
ball2.pos
vibst.radius = 1.0

# Parameters
rho = 0.01                                                    # string density
ten = 40.                                                     # string tension
c = math.sqrt(ten/rho)                                      # Propagation speed
c1 = c                                                          # CFL criterium
ratio =  c*c/(c1*c1)

# Initialization
xi = zeros( (101, 3), Float)                       # 101 x's & 3 t's (maybe float)
for i in range(0, 81):
    xi[i, 0] = 0.00125*i;                                   # Initial conditions

for i in range (81, 101):                               # first part of string
    xi[i, 0] = 0.1 - 0.005*(i - 80)                         # second part of string

for  i in range(0, 100):                                          # First t step
    vibst.x[i] = 2.0*i - 100.0          # assign & scale x:  0<i<100  - >  - 100<i<100
    vibst.y[i] = 300.*xi[i, 0]                   # assing & scale y: xi to 300*xi
vibst.pos                                                         # draw string

# Later time steps
for i in range(1, 100):                                         # use  algorithm
   xi[i, 1] = xi[i, 0] + 0.5*ratio*(xi[i + 1, 0] + xi[i - 1, 0] - 2*xi[i, 0])
while 1:                                    # continue plotting until user quits
    rate(50)                                # delays plotting, (bigger = slower)
    for i in range(1, 100):
        xi[i, 2] = 2.*xi[i, 1] - xi[i, 0]  +  ratio*(xi[i + 1, 1]  +  xi[i - 1, 1]  -  2*xi[i, 1])
    for i in range(1, 100):
         vibst.x[i] = 2.*i - 100.0                           # scaledx - positions
         vibst.y[i] = 300.*xi[i, 2]                        # scaled y - positions
    vibst.pos                                                   # plot string
    for i in range(0, 101):
        xi[i, 0] = xi[i, 1]                                     # recycle array
        xi[i, 1] = xi[i, 2]
print "finished"
```

to display the results; you should try $L = 1000$ to be realistic. The values of density and tension are entered as constants, $\rho = 0.01$ kg/m and $T = 40$ N, with the space grid set at 101

Figure 18.3 The vertical displacement as a function of position x and time t. *Left:* A string initially plucked near its right end forms a pulse that divides into waves traveling to the right and to the left. In both cases y/L should be small, but that would be harder to show in a small space. *Right:* A string initially placed in a standing wave on a string with friction. Notice how the standing wave moves up and down with time. (Courtesy of J. Wiren.)



points, corresponding to $\Delta = 0.01\,\text{cm}$.

18.2.4 Assessment, Exploration

1. Solve the wave equation and make a surface plot of displacement *versus* time and position.
2. Explore a number of space and time step combinations. In particular, try steps that satisfy and that do not satisfy the Courant condition (18.22). Does your exploration conform with the stability condition?
3. Compare the analytic and numeric solutions, summing at least 200 terms in the analytic solution.
4. Use the plotted time dependence to estimate the peak's propagation velocity c. Compare the deduced c to (18.2).
5. Our solution of the wave equation for a plucked string leads to the formation of a wave packet that corresponds to the sum of multiple normal modes of the string. On the right in Figure 18.3 we show the motion resulting from the string initially placed in a single normal mode (standing wave),

Applet

$$y(x, t=0) = 0.001\,\sin 2\pi x, \qquad \frac{\partial y}{\partial t}(x, t=0) = 0.$$

   Modify the program to incorporate this initial condition and see if a normal mode results.
6. Observe the motion of the wave for initial conditions corresponding to the sum of two adjacent normal modes. Does beating occur?
7. When a string is plucked near its end, a pulse reflects off the ends and bounces back and forth. Change the initial conditions of the model program to one corresponding to a