Lesson 21: Boundary Value Problems

1 Introduction

What is the shape of a hanging chain? We can make this question more precise. Suspend a chain (or rope or cord) of length L from its ends, which are fixed to the points (x_a, y_a) and (x_b, y_b) in the x-y plane. What is the curve $y(x)$ taken by the chain? To answer this question, we first derive an ordinary differential equation satisfied by the function $y(x)$. The equation can be solved numerically as a *boundary value problem*. With a boundary value problem, the freely chosen data are specified at the boundaries (or endpoints) of the system. In contrast, the freely chosen data for an *initial value problem* consist of initial data–typically the initial positions and velocities. For the hanging chain, the boundary data (or boundary conditions) consist of the relations $y_a = y(x_a)$ and $y_b = y(x_b)$.

2 Equation for the hanging chain

Divide the chain into equal length segments of length $\Delta\ell$. Figure (1) shows one of these segments. The forces $\vec{F}_L$ and $\vec{F}_R$ act on the left and right ends of the segment, respectively, by the rest of the chain. The force mg is due to gravity, where m is the mass of the segment. Split the end forces into components:

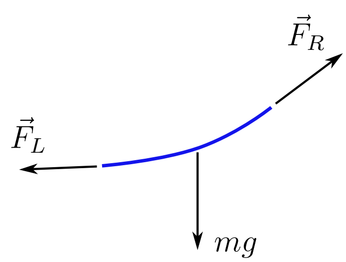

Figure 1: The free body diagram for a segment of a hanging chain.

$$\vec{F}_L = F_{Lx}\hat{x} + F_{Ly}\hat{y} \tag{1a}$$
$$\vec{F}_R = F_{Rx}\hat{x} + F_{Ry}\hat{y} \tag{1b}$$

Since the segment of chain is in static equilibrium, the forces must balance. This yields

$$F_{Rx} + F_{Lx} = 0 \tag{2a}$$
$$F_{Ry} + F_{Ly} = mg \tag{2b}$$

The force vectors at the ends of the segment are tangent to the chain. Thus, the ratio F_{Ry}/F_{Rx} equals the slope at the right end of the segment. Likewise, the ratio F_{Ly}/F_{Lx} equals the slope at the left end of the segment. Since the slope is given by the derivative of the function $y(x)$, we have

$$\frac{F_{Ry}}{F_{Rx}} = \left.\frac{dy}{dx}\right|_R \tag{3a}$$

$$\frac{F_{Ly}}{F_{Lx}} = \left.\frac{dy}{dx}\right|_L \tag{3b}$$

Equations (2a) and (3) can be solved for F_{Ry} and F_{Ly} in terms of F_{Rx}. We then find

$$\left(\left.\frac{dy}{dx}\right|_R - \left.\frac{dy}{dx}\right|_L \right) F_{Rx} = mg \tag{4}$$

from Eq. (2b).

The mass of the segment can be written in terms of the mass per unit length, μ, as $m = \mu\Delta\ell$. From the Pythagorean theorem, the length of the (approximately straight) segment is $\Delta\ell = \sqrt{\Delta x^2 + \Delta y^2}$, where Δx and Δy are the differences in the x and y coordinates of the ends. Now Eq. (4) becomes

$$\frac{1}{\Delta x}\left(\left.\frac{dy}{dx}\right|_R - \left.\frac{dy}{dx}\right|_L \right) = \frac{\mu g}{F_{Rx}}\sqrt{1 + (\Delta y/\Delta x)^2} \tag{5}$$

Note that F_{Rx} is a constant across the length of the chain.[1] Taking the limit as the size of the segment shrinks to zero, the lefthand side becomes the second derivative d^2y/dx^2 . Similarly, the ratio $\Delta y/\Delta x$ becomes the first derivative dy/dx. Thus, we have

$$\frac{d^2y}{dx^2} = k\sqrt{1 + (dy/dx)^2} \tag{6}$$

where $k \equiv \mu g/F_{Rx}$ is a positive constant.

The ordinary differential equation (6) describes the shape of a hanging chain. It is a boundary value problem—the freely specifiable data are given at the boundaries (the ends) of the chain. This equation can be integrated analytically, although we will not use the explicit solution here. The solution curve is called a *catenary*.

3 Numerical solution

There are many numerical techniques that can be used to solve a boundary value problem such as Eq. (6). The most simple (and inefficient) technique is called *relaxation*. Divide the x–axis into N equal intervals of width h. The x–coordinate values of the nodes are x_i where

[1]This follows from Newton's third law. Let A and B denote adjacent segments, with A to the left of B. The value of F_{Rx} for A is equal in magnitude, but opposite in sign, to the value of F_{Lx} for B. In turn, Eq. (2a) shows that F_{Lx} for B is equal in magnitude, but opposite in sign, to F_{Rx} for B. Therefore, F_{Rx} for A equals F_{Rx} for B.

$i = 0, \dots N$. The endpoints are $x_0 = x_a$ and $x_N = x_b$. The separation between adjacent nodes is $x_{i+1} - x_i = h$.

Let y_i denote the y coordinate value of the chain at node x_i. The first and second derivatives of y with respect to x, at node i, can be approximated by

$$\left(\frac{dy}{dx}\right)_i \approx \frac{y_{i+1} - y_{i-1}}{2h} , \tag{7a}$$

$$\left(\frac{d^2y}{dx^2}\right)_i \approx \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} . \tag{7b}$$

If we insert these approximations into the differential equation (6) and solve for y_i, we find

$$y_i = \frac{1}{2}(y_{i+1} + y_{i-1}) - \frac{kh^2}{2}\sqrt{1 + (y_{i+1} - y_{i-1})^2/(2h)^2} . \tag{8}$$

This relationship might appear useless. After all, we can't use it to find y_i since we don't know y_{i+1} or y_{i-1}. However, if we have a reasonable guess for each of the y_i's, we can use this relationship to improve that guess.

To implement this idea, start with a trial solution. The trial solution is a guess for each value of y_i that satisfies the boundary conditions $y_0 = y_a$ and $y_N = y_b$. We then step through the grid, applying Eq. (8) to each of the interior nodes $i = 1, \dots, N-1$. This process is called a *relaxation sweep*. The end result of a single relaxation sweep is an improved set of values y_i. This process can be repeated, as many times as needed, until the values *relax* to the solution of Eq. (8). The (approximate) solution is reached when the y_i values stop changing by a significant amount.

Homework: Write a code that uses relaxation to solve the differential equation (6). Use $k = 5.0$ for the constant, and choose boundary conditions $y(0) = 0$ and $y(1) = 2$. For your trial solution, use the straight line $y_i = 2.0\, x_i$. Experiment with different numbers of relaxation sweeps using a fixed number of nodes, say, $N = 100$. Plot a graph of y versus x for various numbers of sweeps. About how many sweeps are required for curve to relax to the solution?

You probably wrote your code using a `for` loop, or similar control structure, to implement the relaxation sweeps. For example, you could write

```
for i in range(1,N):
     y[i] = 0.5*(y[i+1] + y[i-1]) + ...
```

With numpy's intrinsic indexing, this loop can be replaced by a single statement

```
y[1:-1] = 0.5*(y[:-2] + y[2:]) + ...
```

Here, `y` is a numpy array with elements `y[0]`, `y[1]`,..., `y[N]`. The syntax `y[1:-1]` denotes the range of elements beginning with `y[1]`, and ending with `y[N-1]`. The syntax `y[:-2]` is equivalent to `y[0:-2]`; it denotes the range of elements beginning with `y[0]` and ending with `y[N-2]`. The syntax `y[2:]` is equivalent to `y[2:N+1]`; it denotes the range of elements beginning with `y[2]` and ending with `y[N]`. Your code will run *much* faster if you use the single statement above, rather than the explicit loop.

4 Controlling the error

How many relaxation sweeps are required before the solution stops changing significantly? How should you decide what constitutes a significant change? One way to quantify the change is to monitor the difference between successive "solutions". Let y_i^{old} denote the solution obtained after $s-1$ sweeps, and y_i denote the solution obtained after s sweeps. The L_1 norm is the average of the absolute value of the difference:

$$L_1 = \frac{1}{N}\sum_{i=0}^{N} |y_i - y_i^{old}| \ . \tag{9}$$

This norm is sometimes called the taxicab, or Manhattan norm.

Homework: Modify your code to monitor the L_1 norm of the difference between successive solutions. Have your code continue to carry out relaxation sweeps until the L_1 norm drops below some prescribed value L_1^{max}. You will need to experiment with your code to determine a reasonable value for L_1^{max}. Your goal is to have some assurance that at each node, your numerical solution is within, say, a fraction of a percent of the "ideal" $s \to \infty$ solution.

5 Length of the chain

The numerical solution describes the shape of a chain hanging between the chosen endpoints. The length L of the chain is given by the length of the curve,

$$L = \int_{x_a}^{x_b} \sqrt{1 + (dy/dx)^2}dx \tag{10}$$

as described in elementary calculus.

Homework: Modify your code to include the numerical integration for the length of the chain. Run your code with various values of the constant k, say, $1, 2, \ldots, 8$. Use the resulting data to create a graph of L versus k.

You can approximate the derivative dy/dx at the interior nodes using Eq. (7a). How should you approximate the derivative dy/dx at the endpoints? You could use a one sided stencil. Alternatively, you can avoid this issue by using an "open" integration rule. Let F denote the integrand with values F_i at the nodes $i = 0, \ldots N$. A simple open formula for the integral is given by

$$\int_{x_0}^{x_N} F(x)\,dx \approx h\left[\frac{3}{2}F_1 + F_2 + \cdots + F_{N-2} + \frac{3}{2}F_{N-1}\right] \ . \tag{11}$$