Project Set: Lecture 3

Matthew Ellison
SPISE

July 12, 2019

1 Rain

You can use the library rain.py, which you can use to create a pattern of falling rain which will display on the screen.

The library is easy to use. Here's a sample usage.

```
from rain import Sky
sky = Sky() # Initializes the display window
sky.set_rain_schedule([[0, 10, 20, 100], [], [], [50, 70]])
sky.start_rain(loop = False)
```

Give it a run and see what happens.

The `set_rain_schedule` portion takes in a list of lists which describes the rain pattern. In the above example, raindrops begin to fall (at the top of the window) at x-positions 0, 10, 20, and 100 at the first time step, then no raindrops fall for the next two time steps, and raindrops begin to fall from positions 50 and 70 and the next time step. Since `loop` is set to `False`, no more rain falls. If `loop = True` (try it) the pattern will repeat.

The positions must be integers between 0 and 999 (because the window is 1000 pixels wide). In this mini-project, you'll create functions to conveniently create interesting rain schedules.

1. Create a function to generate a random rain schedule. With the `random` library imported, you can use `random.randint(0, 999)` to generate a random integer from 0 to 999. The function should take in parameters `schedule_length` (the length of the rain schedule), `drops_per_step` (the number of rain drops to randomly generate for each time step).

2 Area Under the Curve

2.1 Part 1

Suppose you want to find the area under the parabola $f(x) = x^2$ between 0 and $x = -5$ and 5. When we say area under the curve, we mean the area sandwiched between the curve and the x-axis, where the area counts as positive if it's above the x-axis and negative if it's below. This is a standard problem of 'integration' and can be solved exactly using the methods of calculus—but we can estimate the area very precisely using the computer, as you'll see.

One easy method to approximate the area is to first break the x-axis range of interest (here -5 to 5) into small pieces, say of size $.1$, and then make a rectangle that comes up at each section, and has height equal to the value of the parabola in the middle of the chunk, as shown in class. The area we're looking for is then approximately equal to the total area of all the rectangles (counting rectangles going downward as negative area). Using smaller pieces would give a more accurate result. If we let x_0, x_1, ..., x_n be the corners of the rectangles on the x-axis in order (so $x_0 = -5$, $x_n = 5$), our approximation is to take Area

$$\approx (x_1 - x_0) f\left(\frac{x_0 + x_1}{2}\right) + (x_2 - x_1) f\left(\frac{x_1 + x_2}{2}\right) + ... + (x_n - x_{n-1}) f\left(\frac{x_n + x_{n-1}}{2}\right)$$

Write a program to use this method to approximate the area under the parabola mentioned above.

2.2 Part 2

Now let's make things a bit more general. Write a function which takes in 3 parameters: `lower_bound` and `upper_bound` for x (previously -5 and 5) and `step_size` (previously .1), and estimates the area under the parabola between the given x-values, using given step size.

2.3 Part 3

Let's make things even more general! Extend your function from the previous part to also take in as a parameter any mathematical function (to take the place of the parabola $f(x) = x^2$). A function may be passed as an argument in Python and may be called as shown in the example below:

```
def evaluate_function_at_0(f):
    return f(0)
```

Your function should be exactly the same as before if the parabola is defined as a function like below, and then parabola is passed in for the function argument.

```
def parabola(x):
    return x ** 2
```

2.4 Optional Extension

The method used above, which estimates the area under the curve using rectangles—is improved upon by a technique known as Simpson's Method. Here's how it works:

1. Break the x-values into pieces of some step size (such as .1). Call then x_0, x_1, x_2, ..., x_n in increasing order with x_0 the left endpoint and x_n the right endpoint. With Simpson's method, n must be odd so that there are an even number of pieces in the x-range.
2. Estimate the integral as

$$\frac{step\_size}{3}(f(x_0)+4f(x_1)+2f(x_2)+4f(x_3)+...+2f(x_{n-2})+4f(x_{n-1})+f(x_n)).$$

Write a function which takes as parameters a (mathematical) function, `step_size`, left bound, and right bound and approximates the area under the curve using Simpson's method.