Week 4 Exercises

This week we will implement a simple observable stream that approximates PI using a Monte Carlo method. We will build upon the exercises from previous weeks by using a more complex state object, additional types, and a pseudo-random number generator.

We have provided a skeleton implementation that includes many of the "busy work" functions that you may choose to use. Your task will be to use what we have learned about RxJS so far to write your own pipeline.

Provided code

We already have functions that assist you in adding dots to the canvas, and helper functions to produce pseudo-random numbers (via a hash).

The random x and y coordinates are in the range -1 to 1. There is a helper function `scaleToRange` which will help you with this. This function takes in a hashed value using `hash` and scales it to the range [-1, 1].

For the calculation of Pi, there is an `approximatePi` function that takes in the inside and outside dot counts and produces the approximate Pi value.

PI Approximation introduction and scenario

> This is **non-essential** information that provides additional context to the exercise and additional explanations. Read this if you are interested.

You're the newest hire on a data visualisation team that's working on visualising the process of approximating Pi using the Monte Carlo method. Congratulations!

Unfortunately most things haven't been finished and you've volunteered to improve the demo! But first, what the heck is a Monte Carlo method!?

The Monte Carlo method is used when a phenomena is easier to simulate than measure. We are going to randomly place dots on a unit circle and tally up which dots hit the circle and which miss. This will approximate Pi because we know the ratio between the circle and square.

Maths

Mathematically the area for a circle is πr^2. Thus the area for a unit circle, or circle with a radius $r = 1$, is π. Let's draw a square around the circle.

The side of the square is 2*radius and therefore the area of the square is equal to:

$$length \times width = 2r \times 2r = 4r^2$$

Therefore the ratio between the circle area and square area =

$$\frac{circle\ area}{square\ area} = \frac{\pi r^2}{4r^2} = \frac{\pi}{4}$$

Thus the probability of a random point falling within the circle is $\frac{\pi}{4}$. Using this probability we can calculate π!

$$\frac{points\ that\ fell\ within\ circle}{total\ data\ point} = \frac{\pi}{4}, \pi = 4 \times \frac{points\ that\ fell\ within\ circle}{total\ data\ point}$$

Below is a concrete example with 17 points.

13 green points inside circle.
4 red points outside circle.
17 total points.

$$\frac{points\ that\ fell\ within\ circle}{total\ data\ point} = \frac{13}{17} \times 4 \approx \pi$$

$$3.05882 \approx \pi$$