CS/INFO 3300; INFO 5100
Homework 4
Due 11:59pm Thursday 3/17

Distributions and histograms

There's an important result in statistics called the central limit theorem. It says that, under fairly mild conditions, the mean of a random sample from a distribution behaves like a Gaussian random variable, even if the distribution you're sampling from behaves nothing like a Gaussian. We're going to see if this is true for a number of different probability distributions.

1. Create a function `plotHistogram` that creates an SVG histogram for continuous (ie floating point) data. The function should take two arguments: a string representing an element id (eg "`#normal`") and an array of numbers. The body of the function should select the element with that id, append an SVG element inside it, construct a linear scale for the x-axis that is appropriate for the values in the data array, and then construct a histogram from the provided data array, with x- and y-axes. Use any d3 functions that are useful. (10 pts)

2. **Normal distribution**. Use `d3.random.normal()` to generate a function called `gaussian`, which will return random values drawn from a normal (Gaussian) distribution with mean 0.0 and standard deviation 1.0. Create a function `gaussian1000` that will generate an array of 1000 samples from your `gaussian` function. Sample an array from this function and use your `plotHistogram` function to create a density plot of this data. (10 pts)

3. Now create an array of length 100, where each element is the mean of an array returned by your `gaussian1000` function (that is, generate the means of 100 different arrays). The function `d3.mean()` will be useful. Use your `plotHistogram` function to create a density plot of this array of means. Does it have roughly the same shape as the plot from Problem 3, and does it have the same x-scale? If not, how is it different? (8 pts)

4. **Lognormal distribution**. You can generate a sample from a lognormal distribution by generating a sample from a Gaussian distribution, and then exponentiating that value. Create a function `lognormal1000` that will generate an array of 1000 samples from a lognormal distribution. You can use your `gaussian` function from the previous problems. Sample an array from this function and use your `plotHistogram` function to create a density plot of this data. (10 pts)

5. Create an array of length 100, where each element is the mean of an array returned by your `lognormal1000` function (that is, generate the means of 100 different arrays). The function `d3.mean()` will be useful. Use your `plotHistogram` function to create a density plot of this array of means. Does it have roughly the same shape as the plot from Problem

3, and does it have the same x-scale? If not, how is it different? (8 pts)

6. **Exponential distribution**. You can generate a sample from this distribution with this expression:

```
 -Math.log(Math.random())
```

Create a function `exponential1000` that will generate an array of 1000 numbers drawn from an exponential distribution. Sample an array from this function and use your `plotHistogram` function to create a density plot of this data. (10 pts)

7. (Seeing a pattern?) Create an array of length 100, where each element is the mean of an array returned by your `exponential1000` function. Use your `plotHistogram` function to create a density plot of these means. Does the histogram of the distribution (from the previous question) look like the histogram of the mean of samples from the distribution? If not, how is it different? (8 pts)

8. **Gamma distribution**. The gamma distribution has a parameter α (there's another one too, but we'll ignore it). The sum of *n* exponential random variables is a samples from a gamma distribution with α=*n*. Create a function `gamma1000` that will generate an array of 1000 numbers drawn from a gamma distribution with α=10. Sample an array from this function and use your `plotHistogram` function to create a density plot of this data. (10 pts)

9. Create an array of length 100, where each element is the mean of an array returned by your `gamma1000` function. Use your `plotHistogram` function to create a density plot of these means. Does the histogram of the distribution (from the previous question) look like the histogram of the mean of samples from the distribution? If not, how is it different? (8 pts)

10. **Cauchy distribution**. Remember about those "mild conditions"? This one doesn't meet them. You can generate a sample from this distribution with this expression:

```
gaussian() / gaussian()
```

where `gaussian` is the Gaussian random variable generator function you created in Problem 2. The expression samples two independent Gaussian random variables and returns their *ratio*.

Create a function `cauchy1000` that will generate an array of 1000 numbers drawn from a Cauchy distribution. Sample an array from this function and use your `plotHistogram` function to create a density plot of this data. (10 pts)

11. Create an array of length 100, where each element is the mean of an array returned by

your `cauchy1000` function. Use your `plotHistogram` function to create a density plot of these means. Does the histogram of the means from this distribution look like the histograms of means from Problems 3 and 5? If not, how is it different? Pay particular attention to the x-axis. (8 pts)