# **Lab 8 – Continuous probability simulations**

In Lab 4, we used the sample() function in order to perform simulations. However, many experiments are modelled by continuous probability distributions such as the ones we have encountered in class – eg - uniform, normal, and exponential. Therefore, in simulating such experiments, it is useful to be able to generate sets of values that follow these kinds of distributions. For instance, if we want to simulate an experiment in which outcomes can be modelled by a normally distributed random variable with mean 10 and standard deviation 2, then if we run the experiment 10000 times, we would want most of the results to be fairly close to 10, and the values to be distributed normally.

As in Lab 4, we are going to simulate experiments and estimate probabilities using the relative frequency approach to probability. That is, we will simulate an experiment *n* times, and count the number of “successes” *k*. This will allow us to estimate the probability of the event as *k.*

**To submit: answers to all numbered questions. When the question asks you to write code, submit the code in the Word document as part of your answer. Also submit a single .R file that contains all of your code.**

# Experiment 1: Waiting for a bus (uniform distribution)

In our first experiment, we imagine a person waiting for a bus that comes very reliably every 20 minutes. (This ideal situation is not very realistic; we will refine the model soon.) However, there’s a problem: the person waiting does not know the bus’s schedule! The person may have been lucky and arrived at the bus stop just before the bus came. Or, they may have just missed the last bus and will have to wait nearly 20 minutes for the next one. The amount of time the person will be waiting before the bus arrives can be modelled by a continuous uniform variable with minimum of 0 and maximum 20.

We can simulate a single person waiting for bus with the **runif()** function.

> runif(n=1, min=0, max=20)

[1] 3.7336

Here, my person was fairly lucky and only had to wait 3.7336 minutes for a bus.

Note that R is rounding the result to 5 digits. We can change the display by using the **options()** command:

> options(digits=20)

> runif(n=1, min=0, max=20)

[1] 14.162643705494702

Technically, the **runif()** function is discrete and not continuous, because R (like all software) can only store finitely many digits. However, it’s pretty close and we can treat it as continuous for our purposes.

We can simulate multiple bus-waiters by changing the first argument **n**. This generates a list of amounts of times that **n** people waited for the bus.

1. Generate appropriately-labelled histograms that give the frequency of waiting times for n=100,1000, and 10000 people who are waiting for a bus that comes every 20 minutes. No need to get fancy with bins. Do the distributions look uniform? That is – when 100 people show up to catch the bus, were there approximately equal numbers of people waiting “short” amounts of time as “medium” and “long” amounts of time? How about when there are 10000 people?
2. Using n=10000, find the proportion of people who wait less than 10 minutes for a bus. Does your answer seem reasonable? Explain.

# Experiment 2: Waiting for a more realistic bus (exponential distribution)

Anyone who has waited for buses knows that no bus comes exactly every 20 minutes (or exactly every 5 minutes, or exactly every hour). More realistically, a bus may come *on average* every 20 minutes. The exponential distribution with mean 20 gives a fairly accurate model of waiting times **between** buses.

We use the **qexp()** function to find exponential waiting times. For an exponential distribution with rate **r,**  the command

qexp(p, rate = r, lower.tail = TRUE, log.p = FALSE)

returns the value such that there is a probability p that the time between occurrences will be **less** than that value.

> qexp(.1, rate = 1, lower.tail = TRUE, log.p = FALSE)

[1] 0.10536

> qexp(.5, rate = 1, lower.tail = TRUE, log.p = FALSE)

[1] 0.69315

> qexp(.9, rate = 1, lower.tail = TRUE, log.p = FALSE)

[1] 2.3026

For instance, at a mean rate of 1 occurrence per minute, there is a 50% chance the waiting time between occurrences will be less than 0.69315 minutes.

1. For the bus example, what is **r**? (Hint: the units of **r** are “buses per minute”.)
2. Find qexp(.5, rate = r, lower.tail = TRUE, log.p = FALSE) for that value of **r**. This should be the median of an exponentially-distributed random variable with mean 20. How does it compare to the mean? Based on the distribution of the exponential random variable, does this makes sense? (Hint: the word “skew” should appear in your answer.)

We use the **pexp()** function to find probabilities associated with exponential distributions. For an exponential distribution with rate **r**, the command

pexp(q, rate = r, lower.tail = TRUE, log.p = FALSE)

returns the probability that the value of an exponentially-distributed random variable with rate **r** is less than **q**.

> pexp(0.3, rate=1)

[1] 0.2591818

For instance, at a mean rate of 1 occurrence per minute, the probability that the waiting time between occurrences is less than 0.3 minutes is 0.2591818.

1. For the bus example, find the following:
   1. The probability you will be waiting less than 10 minutes for a bus. (ans: 0.3934693)

pexp(10,1/20)

* 1. The probability you will be waiting more than 15 minutes for a bus. (ans: 0.473666)

1-pexp(15,1/20)

* 1. The probability you will be waiting between 5 and 10 minutes for a bus. (ans: 0.1722701)

pexp(10,1/20)-pexp(5,1/20)

In order to model **n** waiting times, we need to get waiting times corresponding to **n** probabilities. Those **n** probabilities will be distributed uniformly. That way, approximately 10% of waiting times will be in the bottom 10% of the exponential distribution.

The command

> rexp(n, r)

gives **n** waiting times distributed exponentially with a mean rate of **r** per unit time.

1. Generate appropriately-labelled histograms that give the frequency of waits between buses that have exponentially-distributed waiting times with mean 20 minutes for n=100, 1000, and 10000. Use the same number of classes for each. Do the distributions look exponential?
2. Using n=10000, find the proportion of people who wait less than 10 minutes for a bus. Note that this is the simulation version of question 5a, so your result should be very similar. Now use a formula from lecture to find the probability that a person will wait less than 10 minutes for a bus when the waiting times for buses follow an exponential distribution with mean 20 minutes. Your answers should be very similar – if they aren’t, you’ve done something wrong.

# Experiment 3: Quality control (normal distribution)

In mass-production, companies aim to produce large quantities of identical goods. In practice, the goods are not completely identical, and the variation is typically modelled by a normal distribution.

For example, a battery manufacturer produces thousands of 9V batteries. Ideally, each of the batteries should have a measured voltage of exactly 9.0000000 V. In practice, however, there is some variation in the measured voltages. The true measured voltages of 9V batteries manufactured by this company follow a normal distribution with mean 9.01 V and standard deviation 0.05 V.

We use the **pnorm()** function to find probabilities associated with normal distributions. For a normal distribution with mean **mu** and standard deviation **sigma,** the command

pnorm(q, mean = mu, sd = sigma, lower.tail = TRUE, log.p = FALSE)

returns the probability that the value of a normally-distributed random variable with mean **mu** and standard deviation **sigma** is less than **q**.

> pnorm(9.07, mean=9.01, sd=0.05)

[1] 0.8849303

For instance, when a random variable follows a normal distribution with mean 9.01 and standard deviation 0.05, the probability that a randomly-chosen value will be less than 9.07 is 0.8849303.

1. The true measured voltages of 9V batteries manufactured by a battery company follow a normal distribution with mean 9.01 V and standard deviation 0.05 V. Find the following. (You may want to check at least one of your answers using the tables and formulas from class.)
   1. The probability that a battery’s voltage is less than 9.03V (ans: 0.6554217)

pnorm(9.03,9.01,0.05)

* 1. The probability that a battery’s voltage exceeds 9.02V (ans: 0.4207403)

1-pnorm(9.02,9.01,0.05)

* 1. The probability that a battery’s voltage is between 8.9V and 9.1V (ans: 0.9501662)

pnorm(9.1,9.01,0.05)-pnorm(8.9,9.01,0.05)

We use the **qnorm()** function to find values that follow a normal distribution. For a normal distribution with mean **mu** and standard deviation **sigma,** the command

qnorm(p, mean = mu, sd = sigma, lower.tail = TRUE, log.p = FALSE)

returns the value such that there is a probability **p** that the value of a random variable will be less than that value.

> qnorm(0.75, mean=9.01, sd=0.05)

[1] 9.043724

For instance, when a random variable follows a normal distribution with mean 9.01 and standard deviation 0.05, there is a 75% chance that a randomly-chosen value will be less than 9.043724.

1. The true measured voltages of 9V batteries manufactured by a battery company follow a normal distribution with mean 9.01 V and standard deviation 0.05 V. Find the following. (You may want to check at least one of your answers using the tables and formulas from class.)
   1. The voltage that is larger than 95% of measured voltages (ans: 9.092243)

qnorm(0.95,9.01,0.05)

* 1. The voltage that is lower than 95% of measured voltages (ans: 8.927757)

qnorm(0.05,9.01,0.05)

* 1. The 25th percentile voltage (ans: 8.976276)

qnorm(0.25,9.01,0.05)

We can use the **rnorm()** command to generate **n** values to that follow a normal distribution with given mean and standard deviation. For instance,

> rnorm(n, mean=mu, sd=sigma)

gives **n** normally-distributed values with mean **mu** and standard deviation **sigma**.

1. Suppose the battery manufacturer will ship batteries whose measured voltages are between 8.9V and 9.1V. Give a command that returns the proportion of batteries out of **n** that can be shipped, and give your results for n=100, 1000, and 10000. Note that this a simulation of the question in 9c, so your answers should be very similar. Now use the formulas and tables from lecture to find the proportion of batteries that can be shipped – again your numbers should be very similar.