MAS115 R programming 2016-17
Homework Solutions 1

Homework Format

Please make sure that your homework is readable (in the way that a standard set of notes is). It is really hard to understand and provide useful feedback if you only hand in a script file with just the lines of the code that you plugged in and a few, very sparse, comments.

It is certainly not enough to just give R code and no output: as mentioned previously, you should aim to include enough output in your homework to ensure that the marker can be confident that your code is working, but not an excessive amount. You may find it useful to use `head()` which shows the first few (6, by default) elements in an object.

I don't mind the format of what you hand in, within reason. For example it's fine for your work to be a mix of some handwritten text and some code. You don't have to produce LaTeX (though it is good practice if you want to, of course) but I do want a proper document that can be read from start to finish by anyone and still make sense. The easiest way to do this could be in the form of a .txt file which fully answers all the questions asked (including the mathematical ones which you can write by hand and add at the end). Please don't use a 'screen dump', i.e. an image of what is showing on your screen; it is not really appropriate for exercises of this kind.

Question 1 - gamma random variables

1. The command

   ```
   rgamma(10, 1 , 5)
   ```

   will produce 10 random variables from the gamma distribution with shape $a = 1$ and **rate** $s = 5$. If you looked at the help file the order that the arguments to `rgamma` are passed is

   ```
   rgamma(n, shape, rate = 1, scale = 1/rate)
   ```

   If you do not name the arguments to the function then R will fill them in using the order given in the help file. **It is generally therefore best to use the argument names given when specifying variables to pass to a function.**

   Having looked at the help file you should have seen that to specify the scale parameter, there is an argument called `scale` that you can give, instead of the rate. To generate 10000 numbers from the gamma distribution with shape parameter 2 and scale 4 the easiest way is to write

   ```
   # Produce 10000 numbers from a gamma(2,4)
   x <- rgamma(10000, shape = 2, scale = 4)
   ```

2. In order to find the mean and the variance you use the commands

```
> mean(x)
[1] 8.04621
> sd(x)
[1] 5.648225
```

3. To find the mean of all the entries in `x` which are strictly greater than 0.5 we use the command

```
> mean(x[x > 0.5])
[1] 8.095217
```

This can look intimidating so to explain we'll split it into its several steps. You might want to try these out with a smaller number (maybe 10 or so) of gamma variables and observe what each step gives

```
# Can create a logical condition to check if each value of x is > 0.5
#(vector of TRUE and FALSE values)
x > 0.5
# Use this logical condition to select the values
# i.e just select the x's which have a TRUE in the logical above
x[x > 0.5]
# Find out the mean of these values
mean(x[x > 0.5])
```

4. The command `sum(x > 0.5)` counts the number of random variables which are greater than 0.5. Again, the inner part just creates a boolean or logical vector (i.e. a vector with values `TRUE` or `FALSE`). If you apply a 'numerical' operator, e.g. `sum`, to boolean variables, then R will automatically (and silently!) convert `TRUE` to 1 and `FALSE` to 0 before carrying out the calculation. Try it with some other operators!

```
# Count the number of gamma random variables (out of the 10000)
# which are larger than 0.5
sum(x > 0.5)
```

Question 2 - Estimating π

1. The distance from the centre of the toothpick and the nearest notepad line is $U[0, 2]$ since the toothpick has equal likelihood of landing at any one point. Hence the vertical distance up the page of its centre is also uniformly distributed. Its centre can't be more than 2cm away from the nearest line and so `CentDist` $\sim U[0, 2]$. To create 10000 such distances use the following commands.

```
# Firstly we make sure we are going to be using the same distances
# by calling the set.seed() command. This initialises our random number
```

```
# generator at the same value so our experiment is repeatable and we get
# the same random numbers
set.seed(13) # set.seed() takes as an argument any integer
nsims <- 10000
CentDist <- runif(nsims, 0, 2)
```

2. The distribution of the angle from the vertical is $U[0, \frac{\pi}{2}]$. We can sample some values using

```
# Angle from the vertical is Uniform[0, pi/2]
Angles <- runif(nsims, 0, pi/2)
```

3. Consider a toothpick of length 2 at an angle θ. If we draw a vertical line going through its centre then trigonometry (see the figure in the question) tells us that the ends of the toothpick will be $\cos\theta$ vertically above or below its centre. If this is greater than the distance to the nearest line then the toothpick will cross it. Hence it will cross if and only if

$$x < \cos(\theta)$$

We can count the number of times our simulated throws cross the nearest line (and hence the proportion) by using a logical filtering. It was given in the question that this will tend to $\frac{1}{\pi}$ so we can see how close we get too. We will first create a vector `d` and then compare it to the sampled distances to the nearest line

```
> # For toothpick to touch we need CentDist < cos(Angles)
> d <- cos(Angles)
> ntouch <- sum(CentDist < d)
> ptouch <- ntouch/nsims
> ptouch
[1] 0.32
>
> # We can then check to see if this is 1/pi
> 1/ptouch
[1] 3.125
```

4. Remember that $X \sim U[0,2]$ and $\Theta \sim U[0, \frac{\pi}{2}]$. We will work out the answer by conditioning on the value of Θ observed:

$$\begin{aligned}
P\{X < \cos(\Theta)\} &= \int_0^{\pi/2} P\{\mathbf{X} < \cos(\Theta) | \Theta = \theta\} p_\Theta(\theta) d\theta \\
&= \int_0^{\pi/2} P\{\mathbf{X} < \cos(\theta)\} p_\Theta(\theta) d\theta \\
&= \int_0^{\pi/2} \left\{ \frac{\cos(\theta)}{2} \right\} \frac{2}{\pi} d\theta \\
&= \frac{1}{\pi} \int_0^{\pi/2} \cos(\theta) d\theta \\
&= \frac{1}{\pi}.
\end{aligned}$$