



Newcastle
University

MAS2602

Computing for Statistics

Dr. Lee Fawcett

Semester 1, 2019/20

Lecturer information

The lecturer is Lee Fawcett. If you have any questions about the module he can be contacted via email at lee.fawcett@newcastle.ac.uk. Lee is based in room 2.07 on the second floor of the Herschel Building.

Class schedule

Week 5	Mon 28 Oct	11–12	Lecture (Herschel LT1)
	Fri 1 Nov	10–12	Practical (Herschel PC)
Week 6	Mon 4 Nov	11–12	Lecture (Herschel LT1)
	Wed 6 Nov	11–1	Practical (Herschel PC)
Week 7	Mon 11 Nov	11–12	Lecture (Herschel LT1)
	Thu 14 Nov	11–1	Practical (Herschel PC)
Week 8	Mon 18 Nov	11–12	Lecture (Herschel LT1)
	Thu 21 Nov	11–1	Practical (Herschel PC)
	Fri 22 Nov	1–2	Revision (Herschel LT2)
Week 9	Tue 26 Nov	9–10	Test (Herschel PC)

Office hours are 3-4 on Tuesdays in Herschel room 2.07.

Assessment

The statistics component accounts for 40% of module marks in total. The different assessment components are weighted as follows:

- Assignment – 10% of module marks. Due on **Thursday 21 November, 3pm**.
- Class test – 30% of module marks.

The class test will be held in the Herschel cluster on **Tuesday 26 November, 9am**. The test will last 1 hour, and will be open-book. You will need to complete an R programming task and submit your results via NESS.

Late work policy

In this module, deadline extensions can be requested for the final project (by means of submitting a PEC form), and work submitted within 7 days of the deadline without good reason will be marked for reduced credit. This module also contains tests worth more than 10% for which rescheduling can be requested (by means of submitting a PEC form). For details of the policy (including procedures in the event of illness etc.) please look at the School web site:

<http://www.ncl.ac.uk/maths/students/teaching/homework/>

Course Materials

All course materials will be posted on Blackboard and on a dedicated webpage:

www.mas.ncl.ac.uk/~nlf8/teaching/mas2602

Books

The following books are in the Robinson Library, and are suitable for the material we cover.

- John Braun and Duncan James Murdoch: A First Course in Statistical Programming with R
- Sheldon Ross: A First Course in Probability
- N. A. Weiss, Paul T. Holmes and Hardy Michael: A Course in Probability

Introduction

In this module we will learn about computational tools for performing probability calculations and statistical analysis. We will use the R programming language. R is very widely used in universities, businesses and other organizations due to its many advantages over other similar software packages. It is available free of charge, it is 'open source' and can be used for a very wide range of statistical and scientific applications. Libraries of R functions and data-structures have been created and are readily available on the web. These enable statisticians to perform a very wide range of tasks but with a minimum of additional programming. Learning to use R is important because: (i) it is the foremost tool in modern computational statistics; (ii) it is a great way to learn general concepts in programming which can be used with other languages or in other contexts; and (iii) it can be used to illustrate mathematical ideas in probability and statistics.

The lectures will present a mixture of background theory in probability and statistics, together with elements of the R programming language. Each lecture will therefore contain both some mathematical theory and some R programming concepts. This mixture of material means that notes written only on the board are impractical for this module, and so this booklet of notes has been provided: you need to fill in the blank sections yourself. In addition to the lectures, there are three computer practicals during which the principles discussed in the lectures can be put into practice. The practicals are designed to help students complete the assignment and prepare for the class test. Code written in the practicals can subsequently be used in the class test, which is an open-book exam.

MAS1802 is a prerequisite for this module. You saw how to use R and RStudio in that module, and learned the basics of R programming: how to write functions, and how to use `if` statements and `for` loops. This material will not be covered again during the lectures for this module, but in MAS2602 we will see this material in lots of examples.

1 Simulating random variables

1.1 Discrete distributions: Binomial, geometric and Poisson

The binomial distribution: If $X \sim \text{Bin}(n, p)$ then X has PMF (probability mass function) given by:

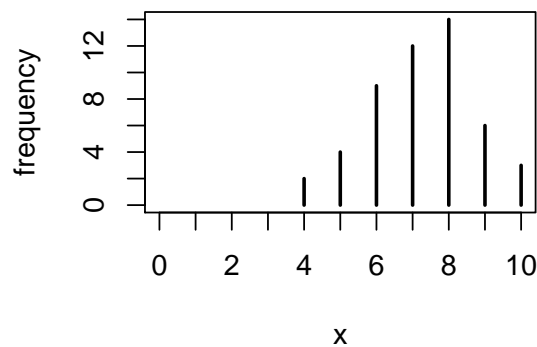
There is no closed formula for the CDF (cumulative distribution function; i.e. $\Pr(X \leq x)$).

The R commands `dbinom`, `pbinom`, and `rbinom` are used respectively to compute the PMF, compute the CDF, and simulate from a binomial distribution. Here are examples:

```
1 dbinom(5, 10, 0.7)      # Pr(X = 5), X ~ Bin(10, 0.7)
2 pbinom(4, 10, 0.7)      # Pr(X ≤ 4), X ~ Bin(10, 0.7)
3 rbinom(50, 10, 0.7)     # Sample a Bin(10, 0.7) distribution 50 times
```

Here's an example of how to create a bar plot of a sample from a binomial distribution:

```
1 x = rbinom(50, 10, 0.7)
2 plot(table(x), xlim = c(0,10), xaxt = 'n', xlab = 'x', ylab = 'frequency')
3 axis(1, at = seq(0, 10))
```



The geometric distribution: If $Y \sim \text{Geom}(p)$ then Y has PMF and CDF given by:

Note that Y takes values in $1, 2, 3, \dots$. R uses a slightly different definition of the geometric distribution. We use the definition that Y is the number of Bernoulli trials with parameter p up to and including the first trial which is a success. The definition in R is the number of trials up to, but **not** including the first success, so in R geometric random variables take values $0, 1, 2, \dots$. Like the binomial distribution, the R commands `dgeom`, `pgeom`, and `rgeom` are used respectively to compute the PDF, compute the CDF, and simulate from the geometric distribution. However, they need to be adjusted to match our definition:

```
1 dgeom(4, 0.2)           # Pr(Y = 5), Y ~ Geom(0.2)
2 pgeom(2, 0.2)           # Pr(Y ≤ 3), Y ~ Geom(0.2)
3 1 + rgeom(100, 0.2)     # Sample a Geom(0.2) distribution 100 times
```

Here's a function to replace `dgeom` with our definition of the geometric distribution:

```
1 mydgeom = function(x, p) {
2   dgeom(x-1, p)}
```


The Poisson distribution: If $Z \sim Po(\lambda)$ then Z has PMF given by:

There is no closed formula for the CDF.

Examples of R commands:

```
1 dpois(5, 3.5)      # Pr(Z = 5), Z ~ Po(3.5)
2 ppois(2, 3.5)      # Pr(Z ≤ 2), Z ~ Po(3.5)
3 rpois(100, 3.5)     # Sample a Po(3.5) distribution 100 times
```

Summary:

Distribution	Binomial	Poisson	Geometric 
PMF	<code>dbinom(...)</code>	<code>dpois(...)</code>	<code>dgeom(...)</code>
CDF	<code>pbinom(...)</code>	<code>ppois(...)</code>	<code>pgeom(...)</code>
sample	<code>rbinom(...)</code>	<code>rpois(...)</code>	<code>rgeom(...)</code>

1.2 Continuous random variables

R has functions for the uniform, exponential and normal distributions:

Distribution	Uniform	Exponential	Normal
PDF	<code>dunif(...)</code>	<code>dexp(...)</code>	<code>dnorm(...)</code>
CDF	<code>punif(...)</code>	<code>pexp(...)</code>	<code>pnorm(...)</code>
Quantile	<code>qunif(...)</code>	<code>qexp(...)</code>	<code>qnorm(...)</code>
sample	<code>runif(...)</code>	<code>rexp(...)</code>	<code>rnorm(...)</code>

Examples of R commands:

```
1 dunif(3, 2, 5)      # f_X(3), X ~ U(2, 5)
2 pexp(1.5, 5)         # F_Y(1.5), Y ~ Exp(5)
3 rnorm(10, 0, 2)      # Sample a N(0, 2^2) distribution 10 times
```

For the standard uniform ($U(0, 1)$) and standard normal ($N(0, 1)$) distributions you don't need to provide the parameters $a = 0$, $b = 1$ and $\mu = 0$, $\sigma = 1$ respectively. For example:

```
1 runif(20)           # Samples a U(0, 1) distribution 20 times
2 pnorm(1.96)          # Pr(Z < 1.96), Z ~ N(0, 1)
3 [1] 0.9750021
```

Quantile functions such as `qexp`, `qnorm` solve equations of the form $F_X(\alpha) = p$, for α , given a probability p . For example:

```
1 qnorm(0.9750021)
2 [1] 1.96
```

Example 1.1: Suppose annual maximum wave heights observed off the coast at a flood-prone town are assumed Normally distributed, with mean 2 metres and standard deviation 0.5 metres.

- (a) Write down the R command to find the probability that the largest wave height next year will exceed 3.25 metres.
- (b) Write down the R command to estimate the height of a new sea wall such that we might expect the town to be flooded, on average, once per century. Why might our modelling assumption be invalid?

1.3 Some more advanced probability models

The main reason to use computers for mathematics and statistics is to solve problems which cannot be done by hand or which require very lengthy calculations. This section contains a simple motivating example to illustrate this.

The number of arrivals per day at a university IT help-desk can be modelled using a Poisson distribution. However, it is thought that the mean of the Poisson distribution might vary from day-to-day. For example, the desk is often busier nearer the end of term due to more students having deadlines. The following model is proposed. Suppose the number of arrivals $X|\Lambda \sim Po(\Lambda)$ where the rate Λ is itself a random variable, with $\Lambda \sim Exp(c)$ for a constant $c = 0.05$. What can we say about the distribution of X ?

The random variable X does not have a distribution which we can immediately recognize. In fact, much can be done analytically (as we will see in MAS2903 in semester 2), but we will use simulation to answer the following questions:

1. What are the expectation and variance of X ?
2. What is $\Pr(X > 30)$?

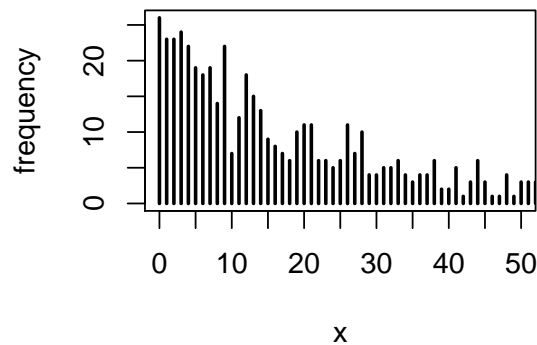
Here is code to simulate a number n of samples from the distribution of X :

```

1 arrivals = function(n, c = 0.05) {
2   x = vector(mode = 'numeric', length = n)
3   for (i in 1:n) {
4     lambda = rexp(1, rate = c)
5     x[i] = rpois(1, lambda)
6   }
7   return(x)
8 }
```


Here's how to create a bar plot of the sample and compute the **sample** mean and variance:

```
1 x = arrivals(500)
2 plot(table(x), xlim = c(0, 50), xaxt = 'n', xlab = 'x', ylab = 'frequency')
3 axis(1, at = seq(0, 50, 5))
```



```
1 mean(x)
2 [1] 19.618
3 var(x)
4 [1] 382.1764
```

The mean and variance of the **distribution** will be close to the sample mean and variance (see section 6). To calculate $\Pr(X > 30)$ we count the proportion of times this occurs in the sample:

```
1 sum(x>30)/500
2 [1] 0.202
```

1.4 Mixtures of normals

Suppose $\mu_1, \mu_2, \sigma_1 > 0, \sigma_2 > 0$ are fixed constants, and w_1, w_2 are positive constants with $w_1 + w_2 = 1$. Consider the following function:

$$f(x) = w_1 f_1(x) + w_2 f_2(x)$$

where f_1 and f_2 are the density functions for $Z_1 \sim N(\mu_1, \sigma_1^2)$ and $Z_2 \sim N(\mu_2, \sigma_2^2)$ respectively.

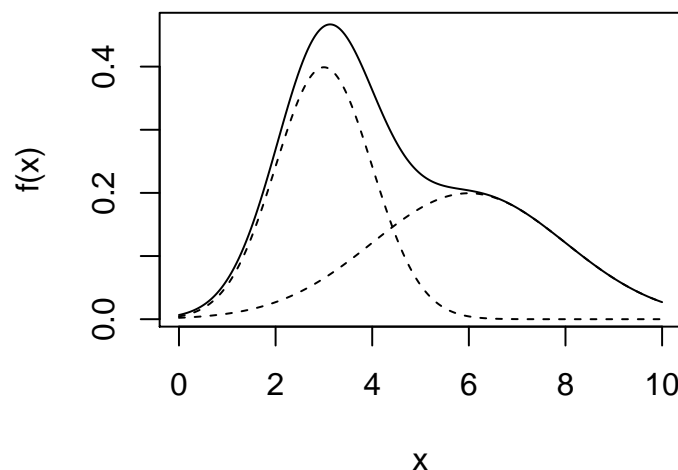
Check that $f(x)$ represents a valid probability density function:

If a random variable X has PDF corresponding to $f(x)$ we say it is a **mixture** of normal distributions $N(\mu_1, \sigma_1^2)$ and $N(\mu_2, \sigma_2^2)$ with weights w_1, w_2 . Note that X is **not** the sum of two normal random variables, i.e.

$$X \neq w_1 Z_1 + w_2 Z_2 \quad \text{where } Z_i \sim N(\mu_i, \sigma_i^2), \quad i = 1, 2.$$

The right hand side is actually another normal distribution (as seen in MAS1604), but X **does not have** the PDF of a normal random variable.

For example: suppose $\mu_1 = 3, \sigma_1 = 1$ and $\mu_2 = 6, \sigma_2 = 2$ with $w_1 = w_2 = 1/2$. Here is a plot of $f(x)$:



Consider how to sample X from this distribution:

1. Sample a random variable $J \in \{1, 2\}$ such that $\Pr(J = 1) = w_1$ and $\Pr(J = 2) = w_2$. The random variable J tells you which **component** of the mixture to sample X from.
2. If $J = 1$ then sample X from $N(\mu_1, \sigma_1^2)$, but if $J = 2$ then sample X from $N(\mu_2, \sigma_2^2)$.

Here is an R function to perform this sampling n times:

```

1  normal.mixture = function(n, mu1, sig1, w1, mu2, sig2, w2) {
2    p = c(w1, w2)
3    x = vector(mode = 'numeric', length = n)
4    for (i in 1:n) {
5      j = sample(c(1, 2), 1, prob = p)
6      if (j==1) {
7        x[i] = rnorm(1, mu1, sig1)
8      }
9      else {
10       x[i] = rnorm(1, mu2, sig2)
11     }
12   }
13   return(x)
14 }
```

Write down any notes about this function.

Normal mixture distributions can contain more than two components – the generalization is very straightforward. They offer a very flexible way to model data with different shaped distributions.