MATH11712 Statistics I
Semester 2, 2022 / 2023
Example Class 7 - Simulation (R session)

Introduction

In this session we will be looking at using simulation techniques to investigate the sampling distributions of various sample statistics. For each set of exercises, I have written a particular R function for the purpose. Additional computations can be done using R functions that you have practiced in the previous three sessions.

User-written R functions

The R language allows users to create their own functions which may be installed into an R session and the used in exactly the same way as any of the regular, intrinsic R functions. A new R function is defined by an assignment of the following form:

```
> function.name <- function(arg_1, arg_2, ...) {
   # type the body of the function here
 }
```

The body of the function is surrounded by brackets { and }, and defines the commands that are run in order to generate the output of the function. The values of the arguments `arg_1`, `arg_2`, `...` may be used as part of the computations. The function body should contain commands that store the desired output in an object, say `x`, and then finally instruct R to return this object via `return(x)`. An R function can output a single numerical value (e.g. the `mean` function), or it can gather together several different types of objects (e.g. scalars, matrices, and so on) in a a data frame or `list` object.

The emphasis in this session is on using the functions that I have provided and drawing conclusions from the results. However, I would strongly encourage you to thoroughly read the code supplied line by line to check you understand it. Do this while referring back to the Example Sheets for the previous R sessions, and/or other references, to aid your understanding. For example, you may find Chapter 9 of 'An Introduction to R' by Venables and Smith useful for information on grouping, loops and conditional execution (`if` statements) in R. This document is available through the R help.

In this session we will be using four specially written functions called `newf.1`, `newf.2`, `newf.3` and `newf.4`. These are all fully given in the file `https://minerva.it.manchester.ac.uk/~saralees/Rfunctions.txt` The first steps you should take are as follows:

(i) Create a folder for this session in which you can store the results from your simulations.

(ii) Change directory to this new folder from within R, using the menu.

(iii) Copy and paste the entire contents of the txt file `https://minerva.it.manchester.ac.uk/~saralees/Rfunctions.txt` into the command window of R.

(iv) Type `ls()` after the prompt in R. In the list of objects stored in your workspace you should see the functions `newf.1`, `newf.2`, `newf.3` and `newf.4`. They are now ready to use like any other function in R.

Simulating the sampling distribution of the sample variance

In this section we will look at simulating the sampling distribution of the sample variance for a normal random sample. We will use the function `newf.1` which is given by:

```
newf.1 <- function(n, mu=0, stdev=1, nsamps=500) {
   ns <- nsamps
   simdat <- numeric(n)
   nsvar <- numeric(ns)
   ss1 <- numeric(ns)
   for (i in 1:ns) {
      simdat <- rnorm(n, mean=mu, sd=stdev)
      nsvar[i] <- var(simdat)
   }
   ss1 <- (n-1)*nsvar/(stdev^2)
   return(ss1)
}
```

This function generates `nsamps` samples, each of size `n` from a normal distribution with mean and standard deviation equal to `mean` and `stdev` respectively. At the very least, the user must specify the sample size `n`. By default, the other arguments `mean`, `stdev` and `nsamps` are set to 0, 1, and 500 respectively. If desired, other values for the latter three arguments may be specified when using the function.

The function returns a vector `ss1` which contains the value of

$$\frac{(n-1)s^2}{\sigma^2}$$

for each sample.

Exercise:

(i) Construct a histogram of the simulated values of $\frac{(n-1)s^2}{\sigma^2}$, and superimpose the appropriate χ^2 p.d.f. to check the goodness-of-fit. Do this for $n = 50, n = 20$ and $n = 100$.

An example of using the function in this way is given in the following code:

```
res1 <- newf.1(n=50)
hist(res1, freq=F)
xv <- seq(from=15, to=95, by=0.5)
c49 <- dchisq(xv, df=49)
lines(xv, c49)
```

If you would like to look at a histogram of just s^2 values, then transform the values in `res1` by multiplying by the value of σ^2 and dividing by $(n-1)$. E.g. for $\sigma = 1$ and $n - 1 = 49$ we have:

```
sv <- 1^2 * res1/49
hist(sv, freq=F)
```

You can add your own title and axis labels to the histogram. Simulations such as this are a good way of directly assessing the variability that can occur in the value of s^2 over different random samples of size n from a specified normal distribution.

Simulating the sampling distribution of the sample mean and median

To simulate the sampling distributions of the sample mean and median for normally distributed data we will use the function `newf.2` defined as follows:

```
newf.2 <- function(n, mu=0, stdev=1, nsamps=500) {
   ns <- nsamps
   simdat <- numeric(n)
   xbar <- numeric(ns)
   med <- numeric(ns)
   for (i in 1:ns) {
      simdat <- rnorm(n, mean=mu, sd=stdev)
      xbar[i] <- mean(simdat)
      med[i] <- median(simdat)
   }
   ss2 <- data.frame(xbar, med)
   return(ss2)
}
```

The function `newf.2` can be used to simulate `nsamps` random samples each of size `n` from a normal distribution with specified mean and standard deviation. By default 500 samples are generated from a standard normal distribution. For each sample the mean and median are recorded and the full sets of simulated sample statistics are then returned to the R session in a data frame. The output may be examined numerically and graphically, e.g. as follows:

```
res2<-newf.2(n=30)

summary(res2)
sd(res2$xbar)
sd(res2$med)
boxplot(res2)
```

Exercise:

(i) Run the above code with $n = 30$. How do you think that the distributions of the sample mean and median differ, if at all?

(ii) Repeat for sample sizes $n = 100$ and $n = 200$ and compare your results. Examine histograms of the simulated sets of means and medians.

(iv) For the means we know that $\bar{X} \sim N(\mu, \sigma^2/n)$. Superimpose this p.d.f. on to your histogram.

(v) On the basis of your simulations, which statistic would you choose to estimate an unknown normal mean in practice?

Simulating the sampling distributions of the sample mean, median and variance for Poisson data

This section is a repeat of the previous one, except now we will sample from a Poisson distribution. The relevant function is `nsamp3` defined by:

```
newf.3 <- function(n, lambda=10, nsamps=500) {
   ns <- nsamps
   simdat <- numeric(n)
   xbar <- numeric(ns)
   med <- numeric(ns)
```

```
   xvar <- numeric(ns)
   for (i in 1:ns) {
      simdat <- rpois(n, lambda=lambda)
      xbar[i] <- mean(simdat)
      med[i] <- median(simdat)
      xvar[i] <- var(simdat)
   }
   ss3 <- data.frame(xbar, med, xvar)
   return(ss3)
}
```

The default value of the Poisson parameter is $\lambda = 10$. By default, the number of samples generated is 500.

(i) Obtain results for sample sizes $n = 30$, 100 and 200. Make comparisons between the distributions of the three estimators and between the results for different sample sizes.

(ii) Which estimator of the Poisson parameter λ (the sample mean, median or variance) would you prefer to use in practice and why?

Simulating the sampling distribution of the sample proportion for Binomially distributed data

To simulate the distribution of the sample proportion based on a random sample of n independent $\text{Bi}(1, p)$ observations we use the following function:

```
newf.4 <- function(n, p, nsamps=500) {
   ns <- nsamps
   simdat <- numeric(n)
   ss4 <- numeric(ns)
   for (i in 1:ns) {
      simdat <- rbinom(n=n, size=1, prob=p)
      ss4[i] <- sum(simdat)/n
   }
   return(ss4)
}
```

To use this function, you must specify the values of the sample size `n` and the Binomial probability `p`. By default the number of simulated samples is set to be `nsamps=500`. The output is a vector containing the proportion of successes found in each of the `nsamps` random samples.

Exercise:

(i) Perform simulations using all combinations of $n = 100, 200$ and $p = 0.1, 0.6$.

(ii) Obtain summary statistics for the results. In each case plot a histogram with a $N(p, p(1-p)/n)$ density curve superimposed.

For example:

```
res4 <- newf.4(n=100, p=0.6)
summary(res4)
var(res4)
```

```
sd(res4)
hist(res4, freq=F, xlim=c(0.4, 0.8))
xv <- seq(from=0.4, to=0.8, length=100)
dv <- dnorm(xv, mean=0.6, sd=sqrt(0.6*(1-0.6)/100))
lines(xv, dv)
```