MATH1019 Linear Algebra and Statistics for Engineers

Laboratory 3

1 Statistical Distributions in R

We will look at some of the basic operations associated with probability distributions. There are a large number of probability distributions available, but we only look at a few. If you would like to know what distributions are available you can do a search using the command **help**(*Distributions*).

For every distribution there are four commands. The commands for each distribution are prepended with a letter to indicate the functionality:

"d"	returns the height of the probability density function
"p"	returns the cumulative density function
"q"	returns the inverse cumulative density function (quantiles)
"r"	returns randomly generated numbers

1.1 Binomial Distribution

The binomial distribution model deals with finding the probability of success of an event which has only two possible outcomes in a series of experiments.

R has four in-built functions to generate binomial distribution. They are described below.

dbinom(x, size, prob)
pbinom(x, size, prob)
qbinom(p, size, prob)
rbinom(n, size, prob)

Following is the description of the parameters used −

- **x** is a vector of numbers.
- **p** is a vector of probabilities.
- **n** is number of observations.
- **size** is the number of trials.
- **prob** is the probability of success of each trial.

Try the following:

```
> x <- seq(0,50,by = 1)
> y <- dbinom(x,50,0.5)
> plot(x,y)

> x <- qbinom(0.25,51,1/2)
> print(x)

> x <- pbinom(26,51,0.5)
> print(x)

> x <- rbinom(8,150,.4)
> print(x)
```

In the above example, we have used the **seq**() function which generates a sequence of numbers between two given numbers (in this case 0 and 50) in increments given by the third argument (in this case 1). Of course, for this example, we could have used **c(0:50)** to get the same result!

For discrete distributions, where variables can take on only distinct values, it is preferable to draw a pin diagram, here for the binomial distribution with n = 50 and p = 0.33, we have:

```
> x <- 0:50
> plot(x,dbinom(x,size=50,prob=.33),type="h")
```

The last argument (`type="h", /histogram-like`) causes the pins to be drawn.

Exercises:

1. Simulate tossing a coin 1000 times. Are the results what you would expect?
2. Suppose that n_1 items are to be inspected from one production line and n_2 items are to be inspected from another production line. Let p_1 = the probability of a defective from line 1 and p_2 = the probability of a defective from line 2. Let X be a binomial random variable with parameters n_1 and p_1. Let Y be a binomial random variable with parameters n_2 and p_2. A variable of interest is W, which is the total number of defective items observed in both production lines. Let $W = X + Y$. Use simulation to see how the distribution of W will behave. Useful information could be obtained by looking at the histogram of W_i's generated and also considering the sample mean and the sample variance. In your simulation use the following random variables X and Y: X is binomial with n_1=7, p_1=0.2; and Y is binomial with n_2=8, p_2=0.6.

1.2 Poisson Distribution

Again, there are four functions here, as above. Type **?dpois** to see the usage of these functions.

Try the following:

```
> px<-dpois(x,1)
> plot(x,px,type="h",xlab="x", ylab="P(X = x)", main="PF of X~Pois(1)")
```

1.3 Normal Distribution

In a random collection of data from independent sources, it is generally observed that the distribution of data is normal. Which means, on plotting a graph with the value of the variable in

the horizontal axis and the count of the values in the vertical axis we get a bell shape curve. The center of the curve represents the mean of the data set. In the graph, fifty percent of values lie to the left of the mean and the other fifty percent lie to the right of the graph. This is referred as normal distribution in statistics.

R has four in-built functions to generate normal distribution. They are described below.

dnorm(x, mean, sd)
pnorm(x, mean, sd)
qnorm(p, mean, sd)
rnorm(n, mean, sd)

Following is the description of the parameters used in above functions −

- **x** is a vector of numbers.
- **p** is a vector of probabilities.
- **n** is number of observations(sample size).
- **mean** is the mean value of the sample data. It's default value is zero.
- **sd** is the standard deviation. It's default value is 1.

Try the following:

```
> x <- seq(-10, 10, by = .1)
> y <- dnorm(x, mean = 2.5, sd = 0.5)
> plot(x,y)

> x <- seq(1.5,3.5,by=.01)
> y <- dnorm(x,mean=2.5,sd=0.1)
> plot(x,y,type="l",col="red")


> x <- seq(-10,10,by = .2)
> y <- pnorm(x, mean = 2.5, sd = 2)
> plot(x,y)

> x <- seq(0, 1, by = 0.02)
> y <- qnorm(x, mean = 2, sd = 1)
> plot(x,y)

> y <- rnorm(50)
> hist(y, main = "Normal Distribution")
```

Plot the standard normal curve:

```
> x <- seq(-4,4,0.1)
> plot(x,dnorm(x),type="l")
```

R also makes it easy to combine plots. Let’s overlay a histogram with a normal density plot:

```
> x <- rnorm(100)
> hist(x,freq=F)
> curve(dnorm(x),add=T)
```

The **freq=F** argument to **hist** ensures that the histogram is in terms of densities rather than absolute counts. The curve function graphs an expression (in terms of x) and its **add=T** allows it to over-plot an existing plot.

Exercises:

If X is a Normally distributed random variable with $\mu = 20$ and $\sigma = 5$. Calculate the following using *R*:

(i) $P(X < 15)$
(ii) $P(14 < X < 23)$
(iii) Find k such that $P(X < k) = 0.9345$.

2 Sampling Distribution of the Mean

Let $\bar{X}$ be the mean of a random sample $X_1, X_2, \dots, X_n$ taken from a distribution with mean μ and finite variance $\sigma^2 > 0$. Then the Central Limit Theorem tells us that for a sufficiently large sample size n, the distribution of $\bar{X}$ can be approximated by a distribution that is $N(\mu, \frac{\sigma^2}{n})$.

To view some essential features of sample means, we will generate samples from a population whose structure is known to us. That population is the set of random digits (0 through 9). The population mean can be worked out quite easily using

```
> mean(0:9)
[1] 4.5
```

Now let's generate 100 means of samples of size 2, 3, 4, 5, and 10 taken from the digits 0,…,9. To generate a random sample in *R*, we can use the **sample()** function. So, to generate a sample of size 2 from 0…9, we would type

```
> sample(0:9,2,replace =TRUE)
```

The third argument tells us that sampling should be done with replacement, that is, once an item is selected it is put back in to be available for the next selection. Now we want to repeat, or **r**eplicate, the process 100 times; and each time we do so, we also want to calculate the mean of each sample. We can use the **replicate()** function to achieve this:

```
> replicate(100,mean(sample(0:9,2, replace=TRUE)))
```

You may want to assign the above to a variable to store the numbers in vector form. Do the same for sample means of size 3, 4, 5 and 10. What do you notice about the mean of each of the 100 means? Draw histograms or box plots of each to see what the distribution looks like. Now try a sample size of 50, replicated 1000 times. Does the shape look familiar?