UQ Winter School - The Basics, session 2

Kathryn Kemper

14/06/2022

Objective

This practical session will review some of the key concepts in the lecture today. Namely, we will review a z-score and introduce the standard distributions inbuilt in R. We will also review 3 common hypothesis tests; a chi-square goodness-of-fit test, t-tests, and an Analysis of Variance. We investigate the power of statistical tests and in Q5 examine the pdf (probability density function) of the normal distribution.

Important skills in R include writing functions and loops, as well as simulating data from a normal distribution.

Q1: calculating a Z-score & probability of an observation

The normal distribution is the most important distribution in genetics, linear models and statistics. It is defined by a mean (μ, a measure of the centre) and variance (σ^2, a measure of spread around the mean). We can use inbuilt functions in R to plot or sample from a normal distribution. These functions include:

(1) `dnorm()` - density, returns the height of the normal pdf for the given z-score

(2) `qnorm()` - quartile, returns the z-score for the given quartile

(3) `pnorm()` - probability, returns area under pdf to the left of the given z-score

(4) `rnorm()` - generate random variables from the supplied normal distribution.

Note that the default is a standard normal, N(0,1), but any normal distribution can be generated using the 'mean' and 'sd' arguments. The 'sd' is standard deviation (σ), i.e. the square root of the variance. At the console use `?dnorm` to bring up more information about the functions.

From the lecture, recall that the z-score can be calculated as:

$$z = \frac{(x - \mu)}{\sigma}$$

When applied to all observations, the process of subtracting the mean and dividing by the standard deviation is sometimes referred to as 'standardising' a variable as the resulting variable is has a mean of zero and standard deviation of 1, i.e. a standard normal distribution of N(0,1).

Q1. Calculate the z-score of x = 18 from X ~ N(10,5). Use `pnorm()` to calculate the probability of making this observation, plot the distribution and observation.

```r
# calculate the z-score
mean=10 ; variance=5 ; obs=18
z = (obs-mean)/sqrt(variance)

# what is the probability of the observed value or a value more extreme?
1-pnorm(z) #using the z-score
```

```
## [1] 0.0001733097
```

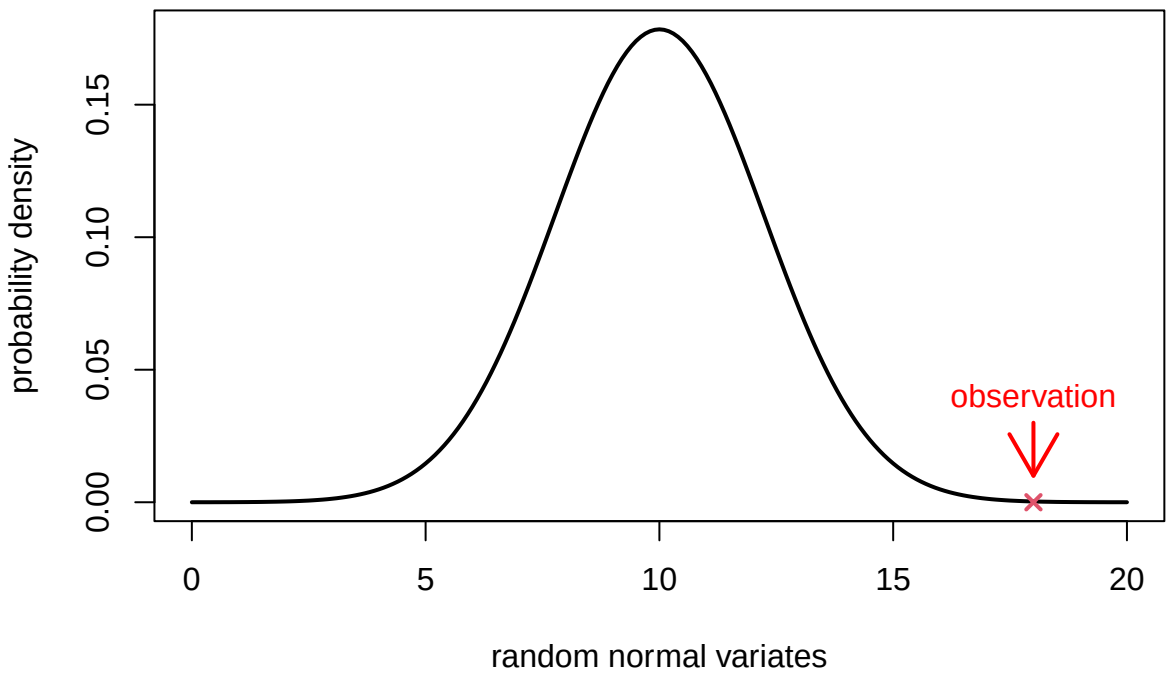
```r
1-pnorm(obs,mean=mean,sd=sqrt(variance)) #alternative, input actual normal curve
```

```
## [1] 0.0001733097
```

```r
pnorm(z,lower.tail = FALSE) # alternative, Pr[obs>x] rather than default, see ?pnorm
```

```
## [1] 0.0001733097
```

```r
# plot the distribution and observation
x = seq(0,20,0.1)
y = dnorm(x, mean=mean, sd=sqrt(variance))
plot(y~x,type="l",lwd=2, ylab="probability density", xlab="random normal variates")
points(obs,0,pch=4,lwd=2,col=2)
text(obs,0.04,"observation",col="red",lwd=2)
arrows(obs,0.03,obs,0.01,col="red",lwd=2)
```



Note that functions for the density function, cumulative distribution function, quantile function and random variate generation are named in the form dxxx, pxxx, qxxx and rxxx respectively in R. They are available for many standard distributions, including those discussed in class such as binomial (`dbinom`), χ^2 (`dchisq`), F- (`df`), and t- (`dt`). It helps to play with these functions and plot to get a feel for the parameters required, and the expected values. In Q5 (below) we examine the inter-relationship between the functions using the normal distribution.

Q2: Chi-squared test

A Chi-squared test tests for independence in a frequency table. It is commonly seen in genetics when testing if a genetic marker deviates from Hardy-Weinberg equilibrium (HWE). Under HWE, if the frequency of a binomial allele A is p, then the frequency of the alternate allele B is $q = 1 - p$. The expected frequency of AA is p^2, AB is $2pq$ and BB is q^2. From probability theory, can you see why this is so?

Recall from the lecture that the chi-square test statistic is given by:

$$\chi^2 = \sum \frac{(O - E)^2}{E}$$

where O is the number observed, and E is the number expected. The test is χ^2 distributed with degrees of freedom (df) equal to the number of categories minus 1. Intuitively, if the number observed is similar to the number expected then the test statistic is relatively small and we have a high chance (p-value) of observing such a deviation.

Q2: The number of individuals with 0, 1 or 2 copies of a SNP allele are 80, 120 and 60. Does this SNP show significant deviation from HWE? Write your own function for a chi-squared test, compare your answer to `chisq.test()`.

Note the notation to write a function in R, `object <-function(parameters) {actions}`

```
# input the observations
obs<-c(80,120,60)

# write a chisq test function
chi_test<-function(obs) {
   n<-sum(obs)
   p<-(2*obs[1]+obs[2])/(2*n)
   q<-1-p
   exp<-c(p^2*n,2*p*q*n,q^2*n)
   test<-sum((obs-exp)^2/exp)
   return(c(test,1-pchisq(test,df=(length(obs)-1))))
}

# test out the function using the observations
chi_test(obs)
```

```
## [1] 1.3265306 0.5151664
```

```
# now use the chisq.test() function in R, note ?chisq.test() to see the input
n=sum(obs)
p<-(2*obs[1]+obs[2])/(2*n)
q<-1-p
```

```r
exp<-c(p^2,2*p*q,q^2)
chisq.test(obs,p=exp)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  obs
## X-squared = 1.3265, df = 2, p-value = 0.5152
```

Q3: t-tests and experimental power

There are many different forms of t-tests but they all construct the test statistic as:

$$t = estimate/s.e.$$

The statistic is assumed to come from a t-distribution with the appropiate degrees of freedom (here df = n - 2, where n is the total number of observations and we estimate the difference between two means). Almost always a 2-tailed t-test is required, i.e. we want to test if an effect or estimate is significantly different (larger or smaller) from zero, and so we must multiply the probability of obtaining the test statistic by two to account for both sides of the distribution.

It might be useful to know that one can **informally** apply a t-test/calculate a z-score to quoted estimates with standard errors very easily. If the ratio (est/s.e.) is > 2 then the estimate is likely to be significantly different to zero. Can you see why this is so?

Q3. Simulate a sample of 8 treatment and control observaions (16 in total) from N(100,100) with a treatment effect of +10 units. Conduct (by-hand) a t-test and evaluate if there is a significant difference between the means. Compare your results to `t.test()` function in R. Finally, use simulation to determine the power of the experiment.

```r
set.seed(1234)  #for reproducibility
n=8
ctl=rnorm(n,mean=100,sd=10)
trt=rnorm(n,mean=110,sd=10)

#t-test ~ estimate/s.e.
est=mean(trt)-mean(ctl) ; est
```

```
## [1] 9.480536
```

```r
se=sqrt(1/n*(var(ctl)+var(trt))) ; se
```

```
## [1] 4.526757
```

```r
t=abs(est/se) ; t
```

```
## [1] 2.094333
```

```r
df=2*n-2  #total sample size - 2
2*pt(t,df=df,lower.tail=FALSE)
```

```
## [1] 0.05490556
```

```r
# using function in R
t.test(trt,ctl,var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  trt and ctl
## t = 2.0943, df = 14, p-value = 0.05491
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2283916 19.1894635
## sample estimates:
## mean of x mean of y
## 106.50918  97.02864
```

Now we are going to determine the power of our experiment using simulation. We know that there is a population difference between our control and treatment groups (+10 units), but sometimes due to sampling we might fail to reject the null-hypothesis. The power of the experiment is the ability to reject the null hypothesis when it is false.

For hypothesis testing there are four possible results:
(1) True positive, reject the null hypothesis when it is false (yeah! right answer)
(2) True negative, accept the null hypothesis when it is true (yeah! right answer)
(3) False positive, reject the null hypothesis when it is true (boo, wrong conclusion - Type I error)
(4) False negative, accept the null hypothesis when it is false (boo, wrong conclusion - Type II error)

We 'control' for the false-positive (Type I) error rate by setting the p-value threshold that we are prepared to accept. Generally this is 0.05, meaning that we are prepared to accept making the wrong conclusion 5% of the time.

Well powered experiments avoid making Type II errors, i.e. accepting the null hypothesis when it is false. Power is a measure of sensitivity of your test to reject the null hypothesis given the size of effect you are aiming to detect. An experimental power of 80% is a rule-of-thumb benchmark for a well-powered study.

We going to use simulation to examine experimental power determine how often we reject the null-hypothesis when we know it is false. From this simulation, hopefully you can see the effect of sampling when estimating population parameters.

```r
reps<-NULL
for (i in 1:10000) {
  ctl<-rnorm(n,100,sd=10)
  trt<-rnorm(n,110,sd=10)

  #difference between the means and s.e.
  est=mean(trt)-mean(ctl)
  se=sqrt(var(ctl)/n+var(trt)/n)
  t=abs(est/se)

  #save the test-statistic
```

```
  reps=c(reps,t)
}
threshold=qt(0.975,df=df) # this is p=0.05 t-value, note abs() t-value is tested
sum(reps>threshold)/length(reps)
```

```
## [1] 0.4606
```

This means we have about 45% power to reject the null-hypothesis when it is false. This is not fantastic, try running the simulation again to determine the sample size needed to have >90% power to reject the null hypothesis when it is false.

Q4: ANOVA

[adapted from Kaps & Lamberson (2004) 'Biostatistics for Animal Science')]

An analysis of variance (ANOVA) generalizes the t-test when there are > 2 groups in the data. It 'partitions' (or divides) the total variance (variance think SSQ/'n') into between and within group components. The technique can be applied to estimate genetic variance in balanced designs by equating the variance components to there expected genetic variance.

To construct a ANOVA table, you require the overall sample mean, and the mean of each treatment group. Then calculate:
(1) the total sums-of-squares (SSQ) as $\sum(x - \mu_{overall})^2$,
(2) the residual SSQ as $\sum(x - \mu_{grp})^2$, and
(3) the treatment SSQ (i.e. $\sum(\mu_{grp} - \mu_{overall})^2$) as $SSQ_{tot} - SSQ_{grp}$.
Determining the df is easiest starting from the total df ($n - 1$, 1 for overall mean), the treatment df as the number of groups/treatments - 1, and the residual as total df - treatment df.

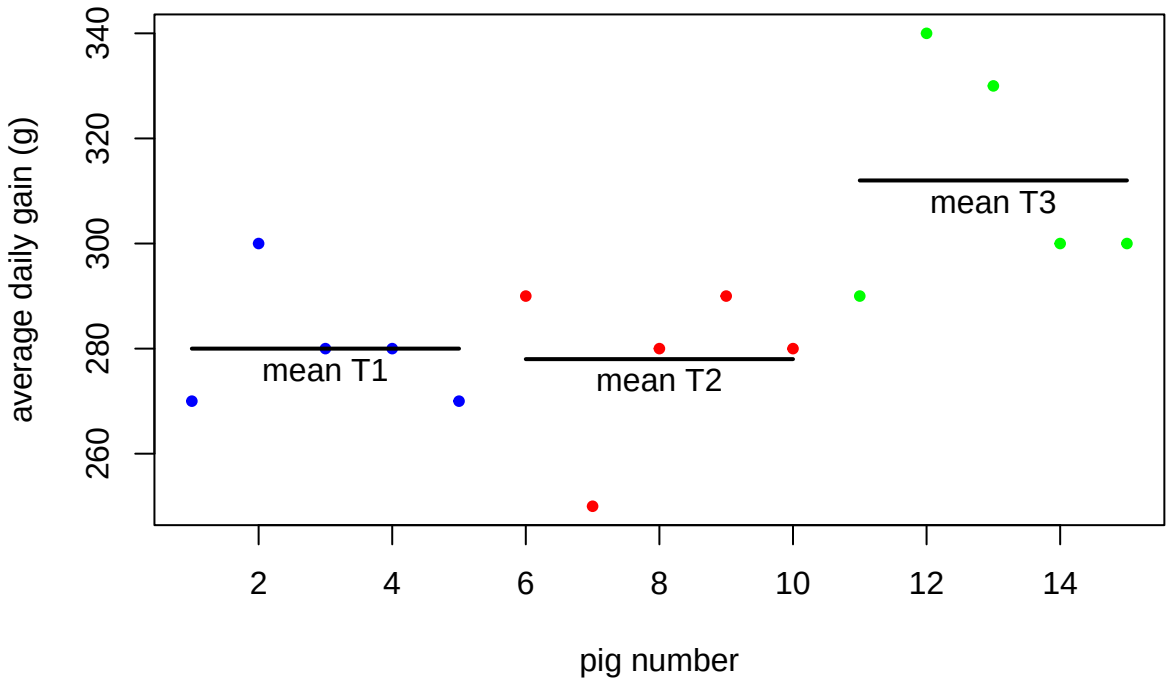
From here we calculate the mean-square (SSQ/df, i.e. the variance) of the residual and treatment groups, and the F-statistic as the ratio of var(treatment)/var(residual). Note that the one-way ANOVA will not tell you which treatment groups were significantly different from each other, only that between group (treatment) effect explained more variation in the sample than within group (residual) variation.

Q4. An experiment was conducted to determine the average daily gain (g) in pigs when fed three different diets; labeled T1, T2 and T3. There were 5 pigs per diet. Conduct an ANOVA by-hand, and then using the `aov()` function in R to determine if diet had a significant effect on average daily gain.

The null hypothesis is that there is no treatment effect of diet on the average daily gain in pigs. The data is shown below, and the mean per group plotted.

```
gain<-c(270,300,280,280,270,290,250,280,290,280,290,340,330,300,300)
treat<-factor(rep(c("T1","T2","T3"),each=5))

plot(1:15,gain,pch=20,col=c(rep("blue",5),rep("red",5),rep("green",5)),
 xlab="pig number",ylab="average daily gain (g)")
for (i in 1:3) {
    segments((i*5-4),mean(gain[(i*5-4):(i*5)]),(i*5),mean(gain[(i*5-4):(i*5)]),lwd=2)
    text((i*5-2),mean(gain[(i*5-4):(i*5)])-4,paste("mean",treat[(i*5)]))
    }
```



Now lets do the ANOVA.

```
  #ANOVA table by hand
  sums<-tapply(gain,treat,sum)
  n<-tapply(gain,treat,length)
  means<-tapply(gain,treat,mean)
  rbind(sums,n,means)
```

```
##          T1   T2   T3
## sums   1400 1390 1560
## n         5    5    5
## means   280  278  312
```

```
  overallMu=mean(gain)
  trtMu=c(rep(means[1],5),rep(means[2],5),rep(means[3],5))

  SST<-sum((gain-overallMu)^2)
  SSE<-sum((gain-trtMu)^2)
  SSR<-SST-SSE

  label<-c("treat","resid","total")
  SS<-c(SSR,SSE,SST)
  df<-c(2,12,14)
  MS<-c(SS[1:2]/df[1:2],NA)
  F<-c(MS[1]/MS[2],NA,NA)
  AOV<-cbind(label,SS,df,MS,F) ; AOV
```

```
##      label   SS     df   MS                 F
## [1,] "treat" "3640" "2"  "1820"             "6.13483146067416"
## [2,] "resid" "3560" "12" "296.666666666667" NA
## [3,] "total" "7200" "14" NA                 NA
```

```
  pVal<-1-pf(F[1],df[1],df[2]) ; pVal
```

```
## [1] 0.01461184
```

```
  #easier using aov()
  summary(aov(gain~treat))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## treat        2   3640  1820.0   6.135 0.0146 *
## Residuals   12   3560   296.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Q5: probability density function of a normal curve

The `dnorm()` function returns the probability density (or height) of a normal curve for a given observation. It is a 'probability density' because the total area under the curve is equal to 1. It is used in more advanced topics in statistics such as in maximum likelihood. Look up `?dnorm` to see the equation for the normal probability density curve (warning, it looks nasty!).

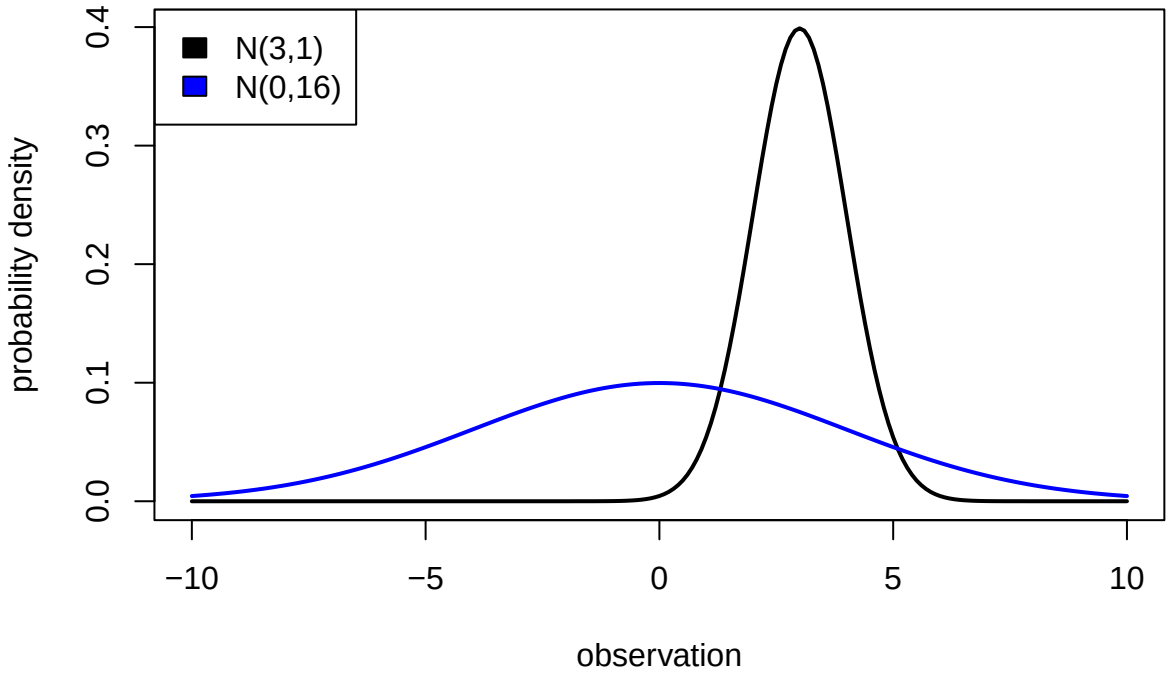
Q5. Write a simple function for the normal density curve. Use this function to approximate the probability of observations between 0 and 1 for N(3,1) and N(0,16). Compare your results to using the `pnorm()` function.

```
  density <- function(x,mu=0,sigma=1) {
    1/sqrt(2*pi*sigma^2) * exp((-1*(x - mu)^2)/(2*sigma^2))
  }
  # height of the standard normal probability density curve at x = 0
  density(0)
```

```
## [1] 0.3989423
```

We are going to examine the behaviour of this function for the two different normal distributions.

```
  # generated some potential observations between -10 and +10
  obs = seq(-10,10,0.1)
  norm1 = density(obs, mu=3)
  norm2 = density(obs, sigma=4)
  plot(norm1~obs, lwd=2, type="l", xlab="observation", ylab="probability density")
  points(norm2~obs, lwd=2, type="l", col="blue")
  legend("topleft",legend=c("N(3,1)","N(0,16)"), fill=c("black","blue"))
```



We can see, for example, that it is much more likely to obtain a value near the mean for the 1st normal curve [N(3,1)] compared to the second curve [N(0,16)] as the 'spread' (or variance) of the distribution is smaller for the 1st curve.

Lets consider the probability of obtaining a value between 0 and 1 for both curves. First, we'll approximate the area under the curve using a trapezoid, i.e. the area of a square with the average height of the curve. We can also use `rnorm()` to generate some random normal deviates from the curves and calculate the proportion of values between 0 and 1. More precise answers can be obtained with `pnorm()`.

```
  # normal 1
avgHeight1 = mean(density(c(0,1),mu=3))
avgHeight1 * 1
```

```
## [1] 0.02921141
```

```
# normal 2
avgHeight2 = mean(density(c(0,1),sigma=4))
avgHeight2 * 1
```

```
## [1] 0.0982013
```

```
# we could also simulate some data to verify
rv1 = rnorm(1000, mean=3)
rv2 = rnorm(1000, sd=4)

sum(rv1>0&rv1<1)/1000
```

```
## [1] 0.026
```

```
sum(rv2>0&rv2<1)/1000
```

```
## [1] 0.1
```

```
# alternatively we could get a more precise answer using pnorm(), use ?pnorm() to examine why?
pnorm(1,mean=3)-pnorm(0,mean=3)
```

```
## [1] 0.02140023
```

```
pnorm(1,sd=4)-pnorm(0,sd=4)
```

```
## [1] 0.09870633
```

We used three approaches [approximate area under the curve, simulating some data and `pnorm()`] to see that the probability of observing a value between 0 and 1 for the first normal curve is about 2%, and about 10% for the second curve.