Simulation, manipulation & power analysis

Aims of this practical

1. Simulate different data types (normal, poisson, binomial)

2. Manipulation of dataset

    Renaming and recoding variables

    Converting between data types

    Long and wide formats

3. Power analysis

Coding you will learn:

Creating data using *rnorm*, *seq*, rep

Restructuring data

Power analysis using *pwr.t.test, pwr.r.test, pwr.chisq.test, power.f2.test*

Data simulation

Simulating a normal distribution

The following basic commands will be useful for most of you in generating data for your Research Poster. Don't forget you may need a poisson distribution for count data. If you are testing for differences between factors, you can simulate two or more continuous variables for groups separately and set the means and SD to be different.

```
# Makes a vector of values with a normal distribution where n=number of observations you want
# to simulate.

n1<-rnorm(n=100, mean = 5, sd = 1)
```

Simulating a poisson distribution

```
# Makes a vector of values with a Poisson distribution where n=number of observations you
# want to simulate and lambda is the mean.

p1<-rpois(n=100, lambda = 3)
```

Check the distributions of the two data sets you have created with an appropriate plot and a normal distribution test?

Simulating a binomial distribution

```
# Makes a vector of values with a binomial distribution where n=number of observations you
# want to simulate, p=the probability of success in each trial (must be value from 0-1) and size is
# the number of trials (e.g. if you set this to 1 the values will be either 1 or 0; if you set this to 10
# the values will be between 0-10).

b1<-rbinom(n=100, size=2, prob=0.75)
```

Simulating sequence data

```
# Simulating sequence data
# Makes a sequence of data from 0-100, every 5 (e.g. 0, 5, 10)
s1<-seq(0, 100, by=5 )

# A sequence of 10 values (length.out) evenly spaced between 0-10
s2<-seq(0, 10, length.out= 10)

# Repeats 1 2 3 4 5 two times
s3<-rep.int(1:5,2)

# Repeats “cat, dog” fifty times
s4<-rep(c("cat","dog"),50)
```

Now create new data called s5 that contains the length of coat each animal has (long, short, medium, curly), you need 25 of each category.

Simulating relationships

Remember from the testing relationships practical that Y=intercept+(slope*X). Using the linear equation $Y=mx+c$ where m=slope and c=intercept, we can create some relationship dummy data.

```
# Simulating relationship data
n2<-rnorm(n=100, mean= 50, sd=5)  # simulate your first variable

# Then simulate your y data using a y=mx+c linear equation. Below I have set c to be 10 and m to
# be 0.5. You can change the slope and the intercept to be appropriate to your study. Changing
# the + slope to a –slope will make it a negative relationship. You also need to add on some
# variation in the relationship to make it realistic, see below (i.e. your R2 isn’t going to be 1!).

y1 <- 10+0.5*n2 + rnorm(100,0,5)  #Noisy relationship, low R2
y2 <- 10+0.5*n2 + rnorm(100,0,2)  #Neater, moderate R2
y3 <- 10+0.5*n2 + rnorm(100,0,1)  #Neat, high R2
```

Plot your relationships and check the R-squared values for the linear relationships by running a linear model for each relationship (i.e. n2~y1)

Data manipulation

Dataframes: renaming variables

Try adding together, in a data frame, some of the variables you have just created (n1, p1, b1, s3, s4, s5, y1, y2, y3) and call the data frame “myData” (hint: use the data.frame function)

Using R’s built in functions rename a column name: change "n1" to "normal"

```
names(myData)[names(myData)=="n1"] <- "normal"
head(myData) # check using “head” to see if this is what you expected
names(myData) # can also check using “names”
```

You can also rename by position in the dataframe (NOTE: this can be dangerous if your data frame could change in the future, why might this be?). Rename by index in the names vector of your dataframe: change the second item, "p1", to "poisson". Again, check using “head” or “names” to make sure you understand what happened.

```
names(myData)[2] <- "poissøn"
```

You could also use the helpful "plyr" package which makes renaming things very easy, search for the "rename" function in the plyr package and try renaming the b1 variable to "binomial".

```
library(plyr) # load plyr package
myData<-rename(myData, c("b1"="binomial"))
head(myData)
```

Dataframes: converting between data types

Use the “str” function to view the data types in your dataframe, what are they?

For some analysis we may need to change between data types (e.g. numbers to factors). There are several ways to do this. Using the following code we can easily change our binomial data to a factor.

```
myData$binomial <- as.factor(myData$binomial)
str(myData)
```

Try changing the “normal” data column to an integer. What has this done to your numeric data?

Another handy “plyr” function is the “revalue” function for recoding variables. Use the code below, what has this done to your dataframe?

```
# Recoding using the plyr package
myData$new.s4 <- revalue(myData$s4, c("cat"="1", "dog"="2"))
```

Again there are several other ways to do this in R, it’s a matter of finding one you prefer. See the R Cookbook website ‘Recoding data’ page for different methods. Go to the website and find out how you would do this using R’s built in functions?

Now lets recode a continuous variable to a categorical variable.

```
# Recode poisson variable above and below 4 to "small" and "large"
myData$category[myData$poisson< 4] <- "small"
myData$category[myData$poisson>=4] <- "large"
# Convert the column to a factor
myData$category <- factor(myData$category)
head(myData)
str(myData)
```

Finally, remember R can be used as a very overly complicated calculator, lets create a new variable called “calc” and add y1 and y2 multiplied by y3

```
myData$calc <- (myData$y1 + myData$y2) * myData$y3
head(myData)
str(myData)
```

Restructuring data

Often data sets are constructed so that variables are in columns and replicates (often sites, or individuals) are in rows. This standard format, which we have been using for most of the course, is called 'long format'. It allows a large variety of analysis and graphing styles and techniques. However, there are times we would need data in a different format (e.g. paired t-test, repeated measures) that require the data to be in 'wide format'. Wide format usually requires the rows to represent individuals and columns to represent repeated measurements.

As you should know by now, R is very clever, and can therefore easily convert between long and wide format without the need to retype all of your data. Below is an example of a data set in long and wide format, look at the differences between these formats, this is the same data just set out differently.

Long format: repeated measures on 4 individuals with three treatments

```
subject sex condition measurement
       1   M   control         7.9
       1   M     cond1        12.3
       1   M     cond2        10.7
       2   F   control         6.3
       2   F     cond1        10.6
       2   F     cond2        11.1
       3   F   control         9.5
       3   F     cond1        13.1
       3   F     cond2        13.8
       4   M   control        11.5
       4   M     cond1        13.4
       4   M     cond2        12.9
```

Same data in wide format:

```
subject sex control cond1 cond2
       1   M     7.9  12.3  10.7
       2   F     6.3  10.6  11.1
       3   F     9.5  13.1  13.8
       4   M    11.5  13.4  12.9
```

Lets try converting this data between long and wide using the *gather* and *spread* functions in the **tidyr** package. First, read in the csv file called *olddata_long* and inspect the data and change the subject column to be a factor.

Long to wide format

The *spread* function requires three bits of information, the data object you want to reformat, the name of the column whose values will be used as column heading, and the name of the column whose values will populate the new columns.

```
## Long to Wide format
olddata_long <- read.csv("olddata_long.csv")  # read in csv
# convert 'subject' column to factor
olddata_long$subject <- factor(olddata_long$subject)
str(olddata_long)

library(tidyr) # load the package

# The arguments to spread() function;
# data: Data object
# key: Name of column containing the new column names
# value: Name of column containing values
data_wide <- spread(data=olddata_long, key=condition, value=measurement)
data_wide
```

Wide to long format

The *gather* function requires you to enter four things; the data object you want to reformat, the name of the column that will contain factors, the name of the new column that will contain values, and the names of the source columns that contain the values.

Read in the csv file *olddata_wide*, again you will need to change the subject column to a factor. Use the code below and make sure you understand what is happening to the original olddata_wide format.

```
# The arguments in the gather() function;
# data: Data object
# key: Name of new key column (made from names of data columns)
# value: Name of new value column
# Names of source columns that contain values
# factor_key=TRUE: Treat the new key column as a factor (instead of character vector)
data_long <- gather(data=olddata_wide, key=condition, value=measurement,
control,cond1,cond2, factor_key=TRUE)
data_long
str(data_long)
```

Power analysis

Its very important when undertaking any statistical analysis to understand the size of the effect you are reporting and the power you have in a dataset to find significant results. An experiment with good quality data will not be able to detect a difference, even if one exists, if the sample size is small relative to the magnitude of difference.

Type I & Type II Error

Type I (False Positive) finding a difference/relationship when it actually doesn't exist

Type II (False Negative) finding no significant difference/relationship when there is one

Research findings	Situation in Reality: Null hypothesis true	Situation in Reality: Null hypothesis false
Null hypothesis true	**Accurate**	**False positive Type I error**
Null hypothesis false	**False negative Type II error**	**Accurate**

How can Power Analysis help?

Power tells you the probability you will find a difference when it is actually real

What information you need to conduct power analysis

1. The statistical test you will be conducting

2. Your significance level (often we use 0.05)

3. Power to detect an effect (the standard recommended power is 0.80)

4. Effect size – how big is the change of interest (small, medium, large?)

5. Sample size – easier to detect an effect with larger sample size

We should already know the type of test to be conducted (e.g. t-test, ANOVA), and the level of significance we are testing at (usually 0.05) and we should be looking to have a high degree of Power (0.80).

Sample size is normally the thing we want to find out so we can plan our fieldwork or sampling regime. To find sample size we need to find out what type of **effect size** is likely to be suitable for our particular research system. This is the tricky bit and may require,

1. Data from a pilot study (not possible for your posters)
2. Looking at the literature to see how big or small differences are, or how small or weak relationships are
3. Making a conservative but educated guess

Today, we'll use R to calculate experimental power for simple experimental designs. In practice, the math is very simple. BUT the effect size (ES) calculation is unique for most statistical tests. Thus, we'll cover just a few basics in this practical. Cohen (1992) provides useful rules of thumb / conventions regarding levels of effect size.

Cohen's power conventions for small, medium and large effects

These conventions should be used with caution. What is a small or even trivial effect in one context may be a large effect in another context. For example, Rosnow and Rosenthal (1989) discussed a 1988 biomedical research study on the effects of taking a small, daily dose of aspirin. Each participant was instructed to take one pill a day. For about half of the participants the pill was aspirin, for the others it was a placebo. The dependent variable was whether or not the participant had a heart attack during the study. In terms of a correlation coefficient, the size of the observed effect was $r = .034$. In terms of percentage of variance explained, that is 0.12%. In other contexts this might be considered a trivial effect, but it this context it was so large an effect that the researchers decided it was unethical to continue the study and contacted all of the participants who were taking the placebo and told them to start taking aspirin every day.

Calculating effect sizes for different tests – don't worry about the maths that follows here. I've put it in so that you can calculate it if you wish to one day! But the size of the effects listed below are important and useful in the exercises to follow.

T-test, difference between two means

Size of effect	*d*	% variance
small	.2	1
medium	.5	6
large	.8	16

$$d = \frac{|\mu_1 - \mu_2|}{\sigma}$$

where μ_1 = mean of group 1
μ_2 = mean of group 2
σ^2 = common error variance

Pearson Correlation Coefficient

Size of effect	*ρ*	% variance
small	.1	1
medium	.3	9
large	.5	25

We use the population correlation coefficient *p* as the effect size measure.

Chi square, contingency tables

Size of effect	*w* = f	odds ratio*
small	.1	1.49
medium	.3	3.45
large	.5	9

$$w = \sqrt{\sum_{i=1}^{m} \frac{(p0_i - p1_i)^2}{p0_i}}$$

where $p0_i$ = cell probability in ith cell under H_0
$p1_i$ = cell probability in ith cell under H_1

i.e., p0 is the expected proportion of expected, p1 is the proportion observed for m categories.

ANOVA

Size of effect	*f*	% of variance
small	.1	1
medium	.25	6
large	.4	14

$$f = \sqrt{\frac{\sum_{i=1}^{k} p_i * (\mu_i - \mu)^2}{\sigma^2}}$$

where $p_i = n_i / N$,
n_i = number of observations in group i
N = total number of observations
μ_i = mean of group i
μ = grand mean
σ^2 = error variance within groups

Multiple R^2

Size of effect	f^2	% of variance
small	.02	2
medium	.25	13
large	.4	26

$$f^2 = \frac{R^2}{1 - R^2}$$

where R^2 = population squared multiple correlation

This is used when we are evaluating the impact of a set of predictors on an outcome.

$$f^2 = \frac{R^2_{AB} - R^2_A}{1 - R^2_{AB}}$$

where R^2_A = variance accounted for in the population by variable set A
R^2_{AB} = variance accounted for in the population by variable set A and B together

This is used when we are evaluating the impact of one set of predictors above and beyond a second set of predictors (or covariates).

Comparing proportions, h

Size of effect	*h*
small	.2
medium	.5
large	.8

p1=First proportion

p2=Second proportion

h = 2*asin(sqrt(p1))-2*asin(sqrt(p2))

For non-parametric tests, it is an acceptable start to use the parametric equivalent power test. However, because the statistical power of non-parametric tests tends to be weaker than their parametric counterparts, you should plan for a slightly larger sample size to account for it (e.g., +10-15%) .

Power Analysis in R – pwr package

Download and load the library **pwr** in R. This library is used to calculate power for many of the most common statistical test you are likely to encounter.

You will probably find that liberal use of the help() function is useful.

The goal here for every example below will be to calculate the required sample size to achieve power = 0.80 (high statistical power).

To do this, we'll need to guess or calculate an effect size (ES). Doing this is specific to each individual test. There are some functions to help us in pwr.

T-test: 1 and 2 sample

Here we'll compare one sample variable against a test parameter, or, more typically, we'll compare two means separated by a categorical variable with two levels using the pwr.t.test() function.

We need to know the effect size for a t-test. The effect size is the parameter d. The definition for a two sample t-test for d is:

d = |(mean1 - mean2)| / (standard deviation)

note: |stuff| is the symbol for absolute value of stuff - no negatives. You can do this in r with the abs() function.

Again, 0.2 is a small ES, 0.5 is medium ES and 0.8 is a large ES. You need pilot data, estimates from literature or an educated guess here (i.e., how SURE are you that there is a difference?).

Let's do some tests.

```
?pwr.t.test

# First, what is the power for a one sample t-test with n=60 subjects, and effect size of d = 0.2
# (small) and alpha (p-value) = 0.05?

pwr.t.test(power = NULL, d=0.2, n=60,   sig.level=0.05,  type = "one.sample", alternative =
"two.sided")
```

R Output:

```
  One-sample t test power calculation

           n = 60
           d = 0.2
   sig.level = 0.05
       power = 0.3316786
 alternative = two.sided
```

Rather low power! How many subjects needed for 0.80 power?

What is the power of a (type=) "paired" t-test with medium ES and n=40.?

What is the power of an experiment for a two-sample, two-sided t-test with n = 30, alpha = 0.05, absolute mean difference of 2 , and pooled standard deviation of 2.8?

Is this good enough to do the experiment? How might you change the experiment to your liking?

Correlation

Here, we'll be using pwr.r.test(). Look at the help for *pwr.r.test*

Conveniently, the correlation coefficient, r, is itself the ES.

If r = 0.3 and n = 50, what is the power for a 2-sided (i.e., positive or negative) correlation?

```
?pwr.r.test

pwr.r.test(r=0.3, n=50, sig.level=0.05 ,alternative="two.sided")
```

What about an experiment where your alternative hypothesis is (for justifiable biological reasons) that the correlation coefficient is “greater” than 0?

How many subjects required to detect a (two.sided) correlation with .80 power if the ES is 0.10? 0.30? 0.50?

Surprised?

Chi square

?pwr.chisq.test

Here Cohen recommends the ES parameter, w, to be 0.1, 0.3 and 0.5, when power is small, medium and large respectively.

Let's say you have an experiment looking at the proportion of offspring expected from 3 phenotypes. You expect a large effect size. If your sample size is 200 offspring, what is your power expected to be?

```
pwr.chisq.test(w=0.5 ,df=(3-1), N=200, pøwer = NULL)
```

What sample size do you need for power = 0.8?

In an experiment looking at the frequency of animals visiting 5 types of animal enrichment objects you expect a small effect size. What sample size of visitation records do you need for high power in this experiment?

General linear model

Calculating power for a "regular GLM model is fairly straightforward, but there are a few extra bits of information required. Check the help file *?pwr.f2.test*

pwr.f2.test(u =, v = , f2 = , sig.level = , power =)

where u and v are the numerator and denominator degrees of freedom. We use f2 as the effect size measure. Cohen suggests f2 values of 0.02, 0.15, and 0.35 to represent small, medium, and large effect sizes.

For a simple, one factor model (analogous to one-way ANOVA): The **numerator** degrees of freedom is usually the number of levels of a factor minus one (may be more complicated in more complicated models). The denominator is the sample size minus the numerator degrees of freedom.

E.g., in an experiment with one factor that has 4 levels (e.g., population north, south, east, west) measuring body size in 100 individuals (25 from each population), u = 4 - 1 and v = 100 - u.

Let's calculate power for the experiment above if the expected ES is medium

```
pwr.f2.test(u = 3, v = 97, f2 = 0.15, sig.level = 0.5, power = NULL)
```

R Output

```
Multiple regression power calculation

         u = 3
         v = 97
        f2 = 0.15
 sig.level = 0.05
     power = 0.9081116
```

How many subjects do we need for 80% power if the ES is small?

What about large effect size?

Additional information:

A piece of software called G-power is also good – open source, free, academically precise, relatively easy to use.

http://www.gpower.hhu.de/en.html