Module 4

Making A Volcano Plot with R

Paul Brennan

Cardiff University

[BrennanP@cardiff.ac.uk](mailto:BrennanP@cardiff.ac.uk)

Oct 2017

### Welcome to R for Biochemists 101 Module 4

Each module is designed to take approximately 60 minutes to complete.

The material provided includes text to read, video demonstrations, an example R script and exercises. We hope you enjoy the modules. We welcome feedback and comments.

### How to use this module

1. To learn from this module, cut and paste the R code into R-Studio and then run the code line by line.
2. See if you can make the script work
3. Watch the demonstration videos to see how the code works
4. Look at how the 'Global Environment' changes.
5. Change the code and test it
6. Try the exercises
7. Extend your learning using the resources

### Learning objectives

1. Make a volcano plot
2. Improving our understanding of factors
3. Force different data set into an existing plot
4. Explore labelling points and colours

### Generating multiple volcano plots

This chapter shows you how to a volcano plot for the visualisation of some genomics data.

### The experiment

This is a gene expression experiment. A cell line was grown in culture. This cell was exposed to three chemicals and their effect on gene expression was investigated by the extraction of mRNA from the cells and their analysis by Affymetrix gene expression chip. The data can be extracted, normalised and comparisons made between the treatments. The statistical analysis was done in R but is not shown here.

The starting point for the analysis today is a file with a list of gene IDs, a list of fold changes and a p-value based on the expression data.

An interesting way to visualise sets of proteomic and genomic data is a volcano plot. In these plots, we show fold change on the x-axis and the p-value on the y axis.

In the data set we are going to import there are three different sets of data. Each data set represents cells treated with different drugs compared to untreated samples. For our example, the drugs are called Drug A, Drug C and Drug D. Drug D causes the biggest changes in gene expression so we are going to start off with that set of samples and then re-use our graph with two other drugs.

Get the packages that we will need

library(ggplot2) # graphing package  
library(ggthemes) # expand the themes available  
library(dplyr) # package for subsetting & manipulating dataframe

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

Download and import the data. We are downloading a csv file from the internet.

data <- read.csv("https://raw.githubusercontent.com/brennanpincardiff/RforBiochemists/master/R\_for\_Biochemists\_101/data/drug\_data\_4\_volc.csv")  
  
# View(data)  
str(data)

## 'data.frame': 5000 obs. of 9 variables:  
## $ X : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Comment.gene\_symbols.: Factor w/ 4862 levels "A1BG","A2LD1",..: 1076 2121 3049 111 3652 1623 4317 273 621 3354 ...  
## $ Comment.descriptions.: Factor w/ 4861 levels "1-acylglycerol-3-phosphate O-acyltransferase 1 (lysophosphatidic acid acyltransferase, alpha)",..: 1061 2024 2974 140 3843 1675 4391 297 448 3327 ...  
## $ Drug\_A.diff.logFC : num 0.03323 0.01242 0.0271 0.05228 0.00646 ...  
## $ a\_p\_val : num 1 1 1 1 1 ...  
## $ Drug\_C.diff.logFC : num -0.164 -0.16 -0.017 -0.185 -0.185 ...  
## $ c\_p\_val : num 0.117 0.175 0.875 0.137 0.107 ...  
## $ Drug\_D.diff.logFC : num -0.176 -0.199 -0.244 -0.34 -0.233 ...  
## $ d\_p\_val : num 0.03358 0.0404 0.01139 0.00388 0.01408 ...

Look at the argument "stringsAsFactors = FALSE". This ensure that the names are recognised as characters.

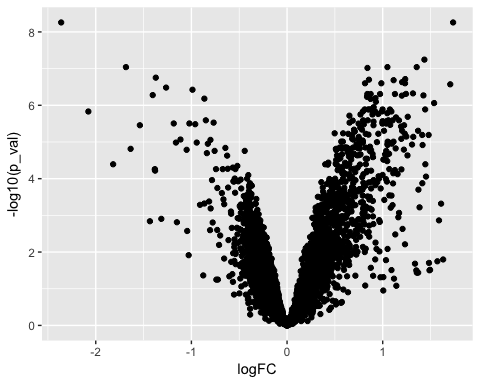
Drug D causes the most changes in gene expression so we're going to start by looking at that. We're going to use the dplyr package to select the columns from our data. Then the colnames() function change the names to the ones we want and understand.

# the dplyr select() function pulls out columns from our object data  
drug\_d <- select(data, Comment.gene\_symbols., Drug\_D.diff.logFC, d\_p\_val)  
colnames(drug\_d) <- c("names", "logFC", "p\_val")

### Draw our first volcano plot

Using ggplot2, we can draw our first volcano plot. Using the treatment of cells with Drug D, we visualise the changes by expressing a negative and positive fold change on the x-axis and a p-value change on the y-axis.

## Draw our first plot  
ggplot(data=drug\_d,   
 aes(x=logFC,   
 y =-log10(p\_val))) +  
 geom\_point()



Note, we are transforming our y values within the ggplot() function. This is sometimes referred to transforming our data 'on the fly'.

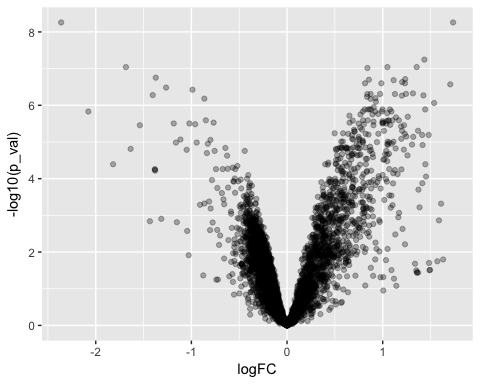
Because of the defaults in ggplot, we can shorten the code a little. The ggplot function assumes that the first argument will be the data and that the first two aesthetics will be the x and y values. Here is the shorter code - it does the same as the previous code.

## Draw our first plot  
ggplot(drug\_d,   
 aes(logFC,   
 -log10(p\_val))) +  
 geom\_point()

### Over plotting...

Because our dataset has 5,000 data points, there is some over plotting. One of the ways to illustrate this is to make the points more transparent by using the alpha argument in the geom\_point() function.

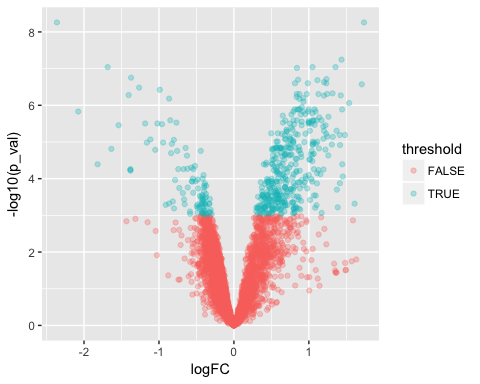
##Construct the plot object  
ggplot(drug\_d,   
 aes(logFC,   
 -log10(p\_val))) +  
 geom\_point(alpha=0.3)



### Colour by threshold

The p-value is an adjusted p-value so colouring the graph with a treshold of the p-value makes the graph quite nice.

# it's useful to add a colour at a threshold to p-value to indicate the number of transcripts that have changed significantly.   
  
drug\_d$threshold = as.factor(drug\_d$p\_val < 0.001)  
  
# plot again...  
ggplot(drug\_d,   
 aes(logFC, -log10(p\_val),   
 colour=threshold)) +  
 geom\_point(alpha=0.3)



### Exercise 1

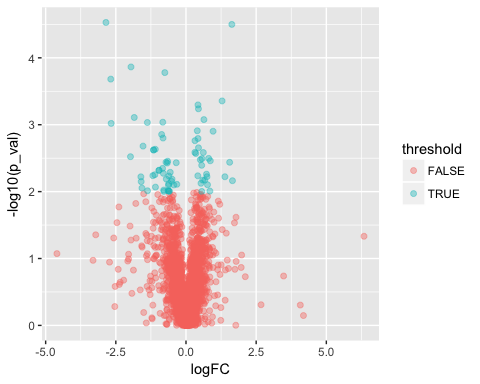
You can test your coding skills a different data set. Here is the code to download the data.

# this is the link to the data  
link <- "https://github.com/brennanpincardiff/RforBiochemists/raw/master/R\_for\_Biochemists\_101/data/mcp.M114.044479-2.xls"  
  
# the download.file() function downloads and saves the file with the name given  
download.file(url=link,destfile="file.xls", mode="wb")  
  
library(readxl)  
# then we can open the file and extract the data using the read\_excel() function.   
data2 <- read\_excel("file.xls", col\_names=TRUE)

Now select the columns you need, check (and change) column names. Add your threshold. Draw your plot.

### Answers for Exercise 1

library(dplyr)  
# select the columns you want...  
data\_s <- select(data2, "Protein Name", "SwissProt Acc. No.",  
 "Log2 Fold Change", "P Value")  
  
colnames(data\_s) <- c("names", "swissprot\_acc", "logFC", "p\_val")  
  
  
##Identify the genes that have a p-value < 0.01  
data\_s$threshold = as.factor(data\_s$p\_val < 0.01)  
  
library(ggplot2)  
## Just plot the data  
ggplot(data=data\_s,   
 aes(x=logFC, y =-log10(p\_val),   
 colour=threshold)) +  
 geom\_point(alpha=0.4, size=1.75)



## N.B. there is a Warning message.

### Make a plot object to reuse

If we create a plot object we can replace the data in it at a later point. Also we can take this opportunity to change the theme, add labels and get rid of the legend which is not useful.

##Construct the plot object  
volc\_d <- ggplot(drug\_d,   
 aes(x=logFC, y =-log10(p\_val),   
 colour=threshold)) +  
 geom\_point(alpha=0.3)

The object appears in the Global Environment as a List of 9.

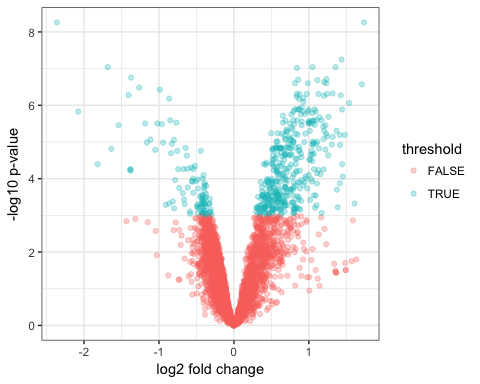
We would like to add a few more thing to this plot...

volc\_d <- volc\_d +   
 xlab("log2 fold change") +   
 ylab("-log10 p-value") +  
 theme(legend.position="none")  
volc\_d # shows us the object - the graph

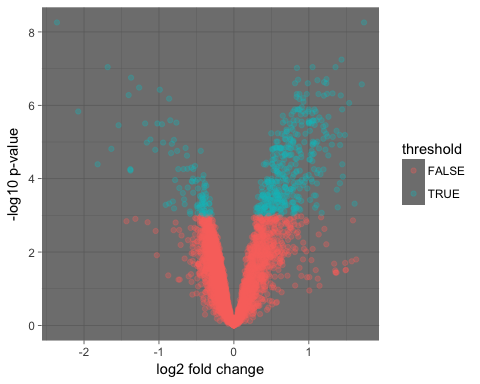


Change the theme - various ones are available...

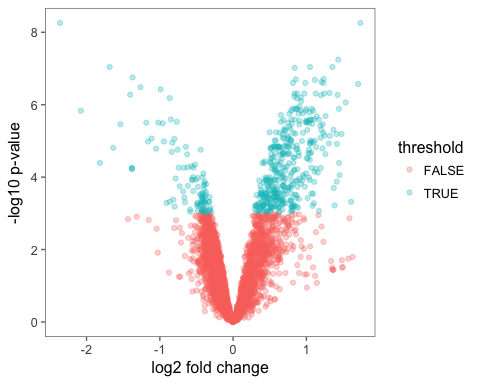
volc\_d + theme\_bw()



volc\_d + theme\_dark() # from ggthemes



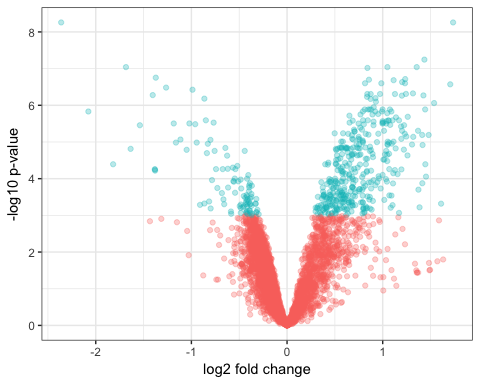
volc\_d + theme\_few() # from ggthemes



# when you have one you like, then modify the object...  
volc\_d <- volc\_d + theme\_bw()

Get rid of the legend as it doesn't really add any information

volc\_d <- volc\_d + theme(legend.position="none")  
volc\_d # shows us the object - the graph

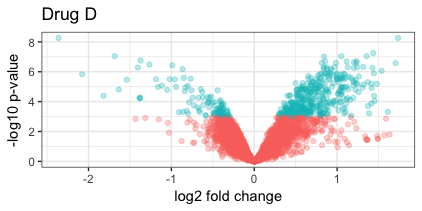
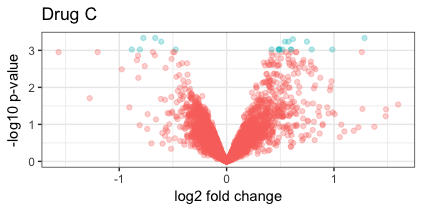
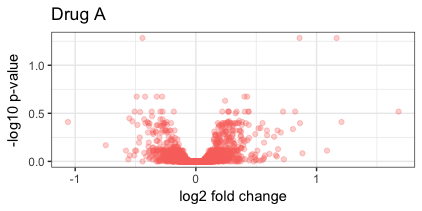


Add a title...

volc\_d <- volc\_d + ggtitle("Drug D")  
# add source? more details?

### Now, let's make plots with other data...

# pull out data for drug A and C & change names...  
drug\_a <- select(data, Comment.gene\_symbols., Drug\_A.diff.logFC, a\_p\_val)  
colnames(drug\_a) <- c("names", "logFC", "p\_val")  
drug\_a$threshold = as.factor(drug\_a$p\_val < 0.001)  
  
drug\_c <- select(data, Comment.gene\_symbols., Drug\_C.diff.logFC, c\_p\_val)  
colnames(drug\_c) <- c("names", "logFC", "p\_val")  
drug\_c$threshold = as.factor(drug\_c$p\_val < 0.001)  
  
  
# this %+% allows us to push a different dataset into an existing   
# ggplot object. It overwrites the data  
# it must contain the same column names otherwise it won't work.  
volc\_c <- volc\_d %+% drug\_c  
  
# then we need to give it a new title  
volc\_c <- volc\_c + ggtitle("Drug C")  
  
# do the same for the other Drug A  
volc\_a <- volc\_c %+% drug\_a  
volc\_a <- volc\_a + ggtitle("Drug A")

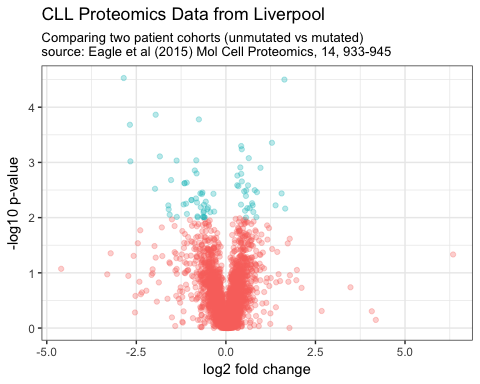
Next page: show our three plots 

### Exercise 2

You should try to reuse the graph from above with the data from exercise 1 or if you prefer customise this plot either manually yourself... Please add a title and maybe the source...

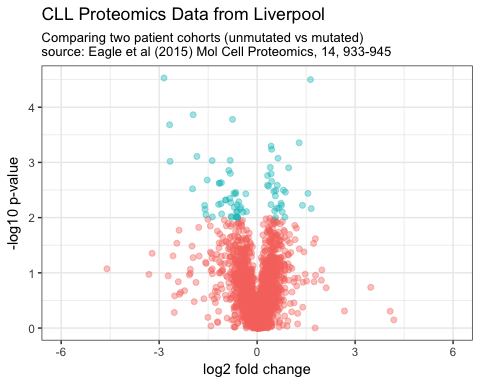
### Answers for Exercise 2

liv\_volc <- volc\_d %+% data\_s  
  
# then we need to give it a new title  
liv\_volc <- liv\_volc +   
 labs(title = "CLL Proteomics Data from Liverpool",  
 subtitle = "Comparing two patient cohorts (unmutated vs mutated)  
source: Eagle et al (2015) Mol Cell Proteomics, 14, 933-945")  
  
liv\_volc



##Construct the plot object  
liv\_volc\_2 <- ggplot(data=data\_s,   
 aes(x=logFC, y =-log10(p\_val),   
 colour=threshold)) +  
 geom\_point(alpha=0.4, size=1.75) +  
 xlim(c(-6, 6)) +  
 labs(x = "log2 fold change", # label x-axis  
 y = "-log10 p-value", # label y-axis  
 title = "CLL Proteomics Data from Liverpool",  
 subtitle = "Comparing two patient cohorts (unmutated vs mutated)  
source: Eagle et al (2015) Mol Cell Proteomics, 14, 933-945") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
liv\_volc\_2

## Warning: Removed 1 rows containing missing values (geom\_point).



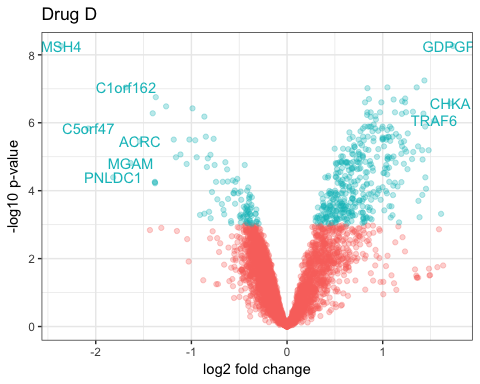
## N.B. there is a Warning message.

# Label some of the points on our plot

It's nice to identify some of the points on the plot to highlight some of the data points of interest.

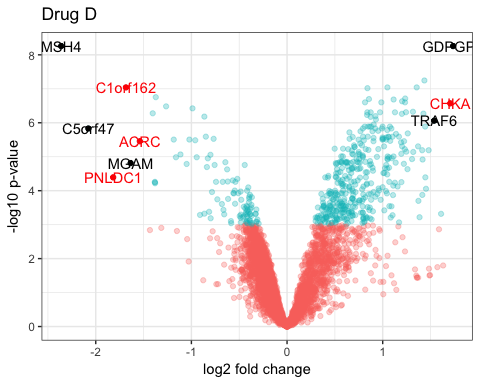
We want to identify points with the biggest changes using the filter() function from the dplyr package.

# find points with the biggest changes  
# want both up and down genes so generate an absolute Fold change  
drug\_d$absFC <- abs(drug\_d$logFC)  
# using the filter() of the dplyr package  
drug\_d\_2label <- filter(drug\_d, absFC>1.5, p\_val <0.0001)  
  
# now add the text using geom\_text  
volc\_d + geom\_text(data = drug\_d\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val),   
 label = names))



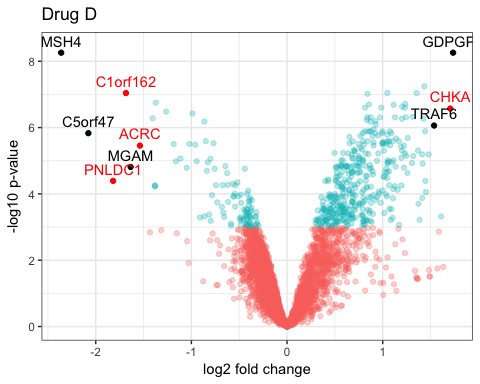
Adding colour to the points to make them more obvious.

# I find it a little difficult to know which points goes with which.   
# so overplot with colour  
# make a simple palette of colours  
# same length as the number of points - 9  
my\_palette <- c("black", "red", "black", "red", "black", "red",  
 "black", "red", "black")  
  
volc\_d + geom\_point (data = drug\_d\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val)),   
 colour = my\_palette) +  
 geom\_text(data = drug\_d\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val),   
 label = names),   
 colour = my\_palette)



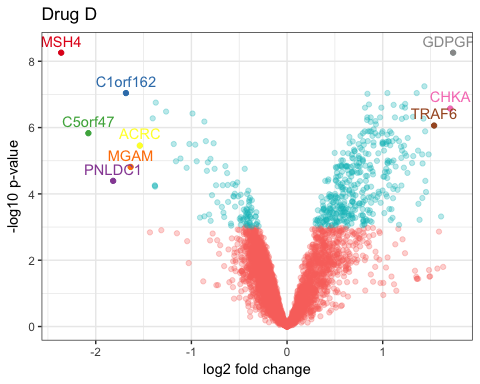
Nudge the labels so that they are just above the points.

# nudge the labels so that just above the points  
volc\_d + geom\_point (data = drug\_d\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val)),   
 colour = my\_palette) +  
 geom\_text(data = drug\_d\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val),   
 label = names),   
 colour = my\_palette,  
 vjust = 0, nudge\_y = 0.2)



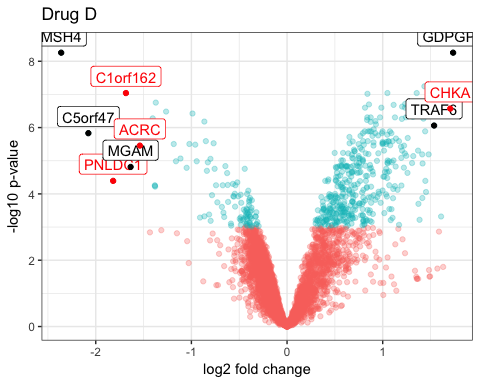
RColorBrewer is a package that allows the automatic selection of palletes. The palettes can be sequential, diverging or qualitative.

# other pallettes can be made of any length  
# the library RColorBrewer has lots of pallettes  
library(RColorBrewer)  
our\_palette <- brewer.pal(nrow(drug\_d\_2label), "Set1")  
  
volc\_d + geom\_point (data = drug\_d\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val)),   
 colour = our\_palette) +  
 geom\_text(data = drug\_d\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val),   
 label = names),   
 colour = our\_palette,  
 vjust = 0, nudge\_y = 0.2)



pagebreak The geom\_label option is also feasible. Here we draw the labels first and then the points.

# geom\_label gives another method   
# here we have drawn the points after the labels...  
volc\_d + geom\_label(data = drug\_d\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val),   
 label = names),  
 colour = my\_palette,  
 vjust = 0, nudge\_y = 0.2) +  
 geom\_point (data = drug\_d\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val)),   
 colour = my\_palette)

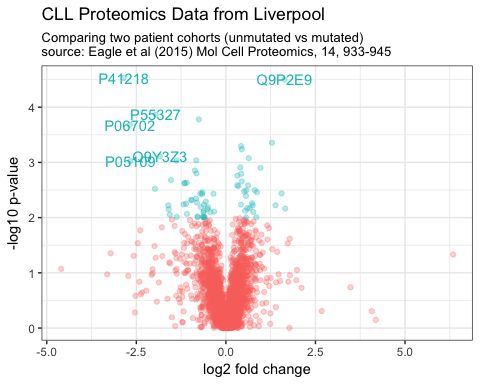


### Exercise 3

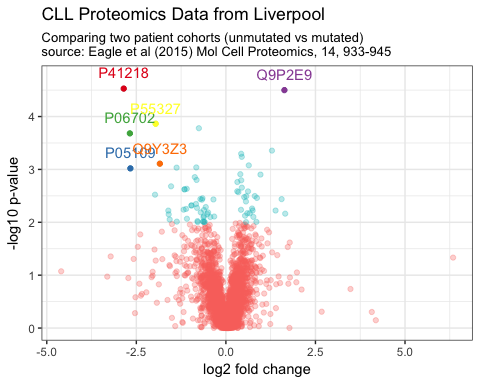
With the data from exercise 1 and the plot from exercise 2, try labelling some of the points. I recommend a fold change cut off of 1.5 and a p value of <0.001. You will need to do absolute Fold Change using the abs() function and then use the filter() function as before.

### Answers for Exercise 3

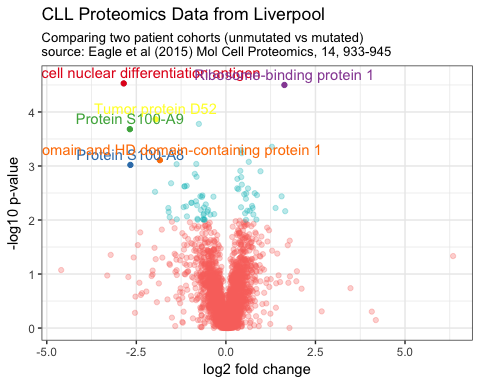
## Exercise 3  
# label some of the points on the data set...  
data\_s$absFC <- abs(data\_s$logFC)  
  
data\_s\_2label <- filter(data\_s, absFC>1.5, p\_val <0.001)  
  
# now add the text using geom\_text  
liv\_volc + geom\_text(data = data\_s\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val),   
 label = swissprot\_acc))



# colour them with Color Brewer pallette  
library(RColorBrewer)  
our\_palette <- brewer.pal(nrow(data\_s\_2label), "Set1")  
  
liv\_volc + geom\_point (data = data\_s\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val)),   
 colour = our\_palette) +  
 geom\_text(data = data\_s\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val),   
 label = swissprot\_acc),   
 colour = our\_palette,  
 vjust = 0, nudge\_y = 0.2)



liv\_volc + geom\_point (data = data\_s\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val)),   
 colour = our\_palette) +  
 geom\_text(data = data\_s\_2label,   
 aes(x = logFC,   
 y = -log10(p\_val),   
 label = names),   
 colour = our\_palette,  
 vjust = 0, nudge\_y = 0.1)



### Review what we have learned

* We have extracted and filtered data with the dplyr package
  + select() function to select columns from a dataframe
  + filter() function to rows with properties we want
* We have made a volcano plot
  + coloured by a threshold
  + transparency set with geom\_point(alpha=0.3)
* We have used more data and our plot to make a new plot
  + volc\_c <- volc\_d %+% drug\_c
* We have explored labelling points and colour
  + making colour palettes with brewer.pal() from RColorBrewer
  + using geom\_text()
  + geom\_label also shown

### Resources

* Try another script available on R for Biochemist which explores a published proteomic data set - it does a little clustering and makes another volcano plot. <http://rforbiochemists.blogspot.co.uk/2015/09/downloading-and-manipulating-published.html>
* There are details the analysis of the Affymetrix drug treatment data was generated here and a heatmap. <http://rforbiochemists.blogspot.co.uk/2016/03/gene-expression-analysis-and_7.html>
* Introduction to dplyr <https://cran.r-project.org/web/packages/dplyr/vignettes/dplyr.html>
* More about dplyr: <http://dplyr.tidyverse.org/>
* Documentation about the geom\_text( ) function <http://ggplot2.tidyverse.org/reference/geom_text.html>
* There is a list of colour names here: <http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf>
* More about colorbrewer here: <http://colorbrewer2.org/#type=sequential&scheme=BuGn&n=3>