A gentle introduction to big data analysis

Nicola Romano

Introduction

In this workshop we will have some hands-on practice with dealing with *big data*. We will use a fairly large dataset containing gene expression data from several cell lines, which are part of the NCI-60 panel. This is a panel of 60 cancer cell lines developed in the 1990s by the National Cancer Institute[1].

The original dataset can be retrieved from NCBI[2]; today we will use a smaller dataset, containing expression data for only some of the cell lines.

[1] See `https://en.wikipedia.org/wiki/NCI-60` and `https://dtp.cancer.gov/discovery_development/nci-60/cell_list.htm` for more info

[2] `https://www.ncbi.nlm.nih.gov/sites/GDSbrowser?acc=GDS4296`

Preparing the data

There are two files called `NCI60.csv.gz` and `info.csv`. The first contains gene expression data for >50000 probes from a microarray experiment[3]), while the latter contains information about the samples.

The first step is to load the files into R using the `read.csv` function[4].

Start by setting the "working directory" to where you saved the files on the computer. In RStudio you can use *Session->Set Working Directory->Choose Directory...* to do that.

Now try the following commands[5]

```
expr <- read.csv("NCI60.csv.gz")
info <- read.csv("info.csv")
```

This has created two variables, `expr` containing the expression data and `info` containing informations about the samples

"Explore" the data using the `head` and `tail` functions, which show the beginning and end of a variable. For example

```
head(expr)
head(info, 20)
```

What does the `20` in the second command do?[6]

We can access a specific row/column of a dataset using the `[]` operator. For example, to look at row 5, column 10 of the gene expression matrix we can do

```
expr[5, 10]
```

```
## [1] 2.76879
```

[3] Note that the human genome has only ~20000 genes; in a microarray experiment, multiple probes are used to measure transcript for the same genes, that is why the number of probes here is so high.

[4] Note: .gz files are compressed files. However R is able to read these files directly using `read.csv`.

[5] R will look for the files in the working directory.

[6] Try and change it!

If the data has an header (like in this case) the columns will be named and we can also refer to them with their name, using the `$` operator (RStudio will helpfully suggest column names).

```
head(info$tissue)
```

```
## [1] leukemia leukemia leukemia leukemia
## [5] leukemia leukemia
## 4 Levels: breast colon ... melanoma
```

The first column of our matrix contains the gene identifiers, we can save that into a separate variable by doing[7]

[7] Note that we do not specify a row number, so R gets all the rows

```
gene.names <- expr[, 1]
```

We now remove column 1 from the expression matrix. This will make things easier later on.

```
expr <- as.matrix(expr[, -1])
```

Normally we arrange our data to have observations on the rows and variables on the columns. However, our expression matrix has cell lines as columns and genes as rows. We can swap rows and columns in our matrix using the `t` function[8]. For example[9]:

[8] The mathematical term for this is to *transpose the matrix*

[9] After doing this, use `head` to look at the matrix. Notice the changes?

```
expr <- t(expr)
```

We now have:

- one variable with the gene names
- one matrix with the expression levels (use `as.matrix` to convert the data frame to a matrix)
- one variable with the sample information

Visualising the data

You may now start by visualising/exploring the data.

For example we may want to ask these questions:

- How many type of cancers are represented in the dataset?
- How many cell lines from each type?

We can use `table` to summarise our data. For instance:

```
table(info$tissue)
```

```
##
##   breast    colon leukemia melanoma
##       15       21       18       26
```

Now, imagine we wanted to look at whether levels of the Ras gene (HRAS) are changed between different groups?

We can plot the data to find that out. Recall that we have genes in columns and samples in rows. Let's find which column corresponds to our gene of interest[10].

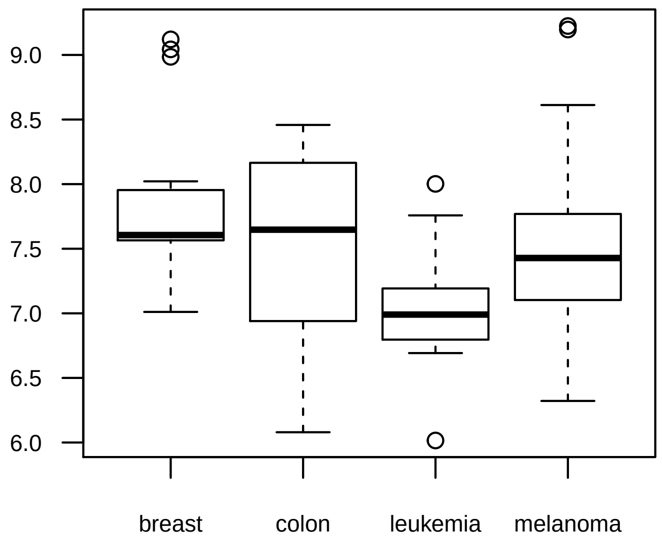
[10] Note that to compare two values in R we need to use == rather than =

```
hras.column <- which(gene.names == "HRAS")
```

So HRAS levels are in column 22287.

The boxplot function allows us to plot a box plot. We can use the ~ operator to divide the plot into groups.

```
boxplot(expr[, hras.column] ~ info$tissue, las = 1)
```



What can you conclude from the plot?

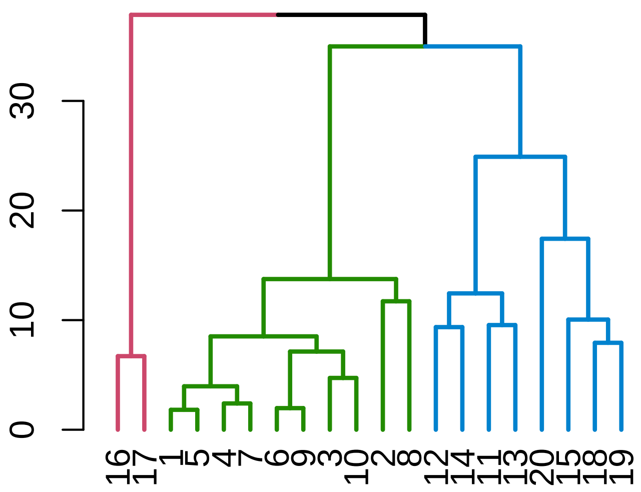
Now, what if I wanted to know what genes are different between leukemia and melanoma? Clearly, trying one gene at a time is not the answer; furthermore the difference may lie in the specific pattern of expression of multiple genes…

How do we go about solving this problem? First of all we may want to try some way of visualising all of our data together.

Clustering our data

The first thing we can see is whether there is some structure to the data. We can use a *dendrogram*. This is a way of displaying how distant data points are in our multi-dimensional parameter space.

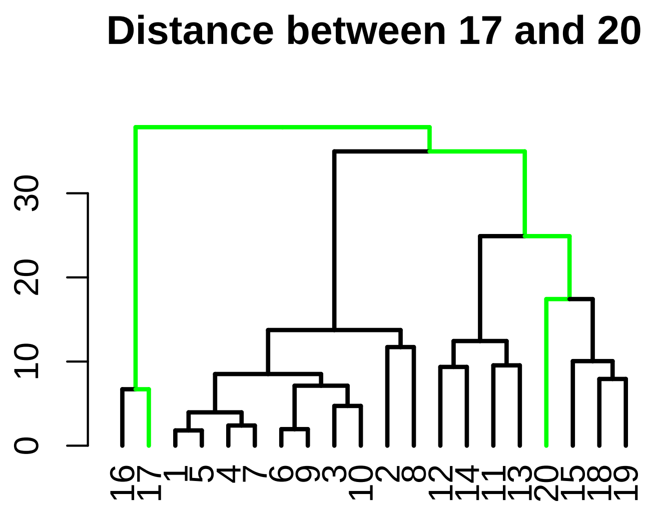
For example, consider this plot:



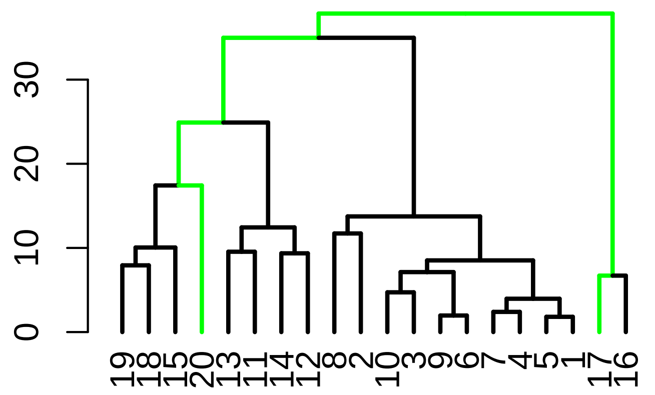
This plot shows 20 samples, that cluster in 3 main branches[11]. Samples in the same branch are more similar to themselves then to samples in other branches. To see how distant two samples are we can start from the *leaf node* corresponding to one sample and "walk" our way to the other sample.

[11] Note: this is a type of *unsupervised clustering*. To generate this plot, I have not told R that there were 3 groups!

For example, the distance between 17 and 20 is marked by the green line in the following plot.

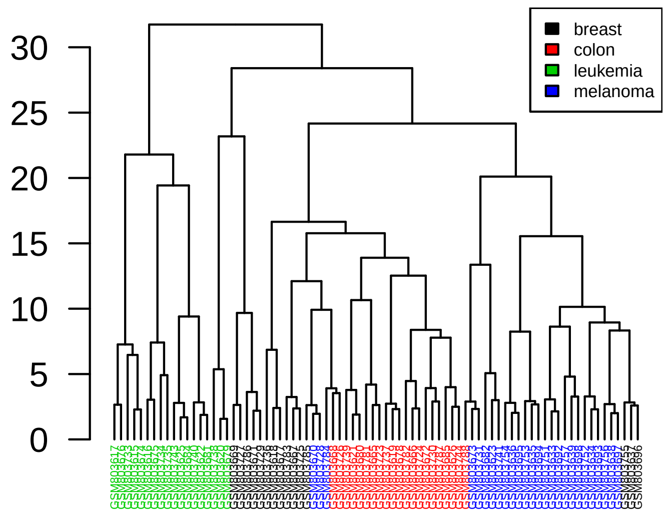


Note that the order of the branches is arbitrary. The following dendrogram is equivalent to the one above.



As an example, here is a dendrogram of our data [12].

[12] Generating this plot is fairly complicated and beyond the point of this workshop



Try to interpret the dendrogram. What does it tell us? Can you think of some interesting experiment that it may prompt you to do?

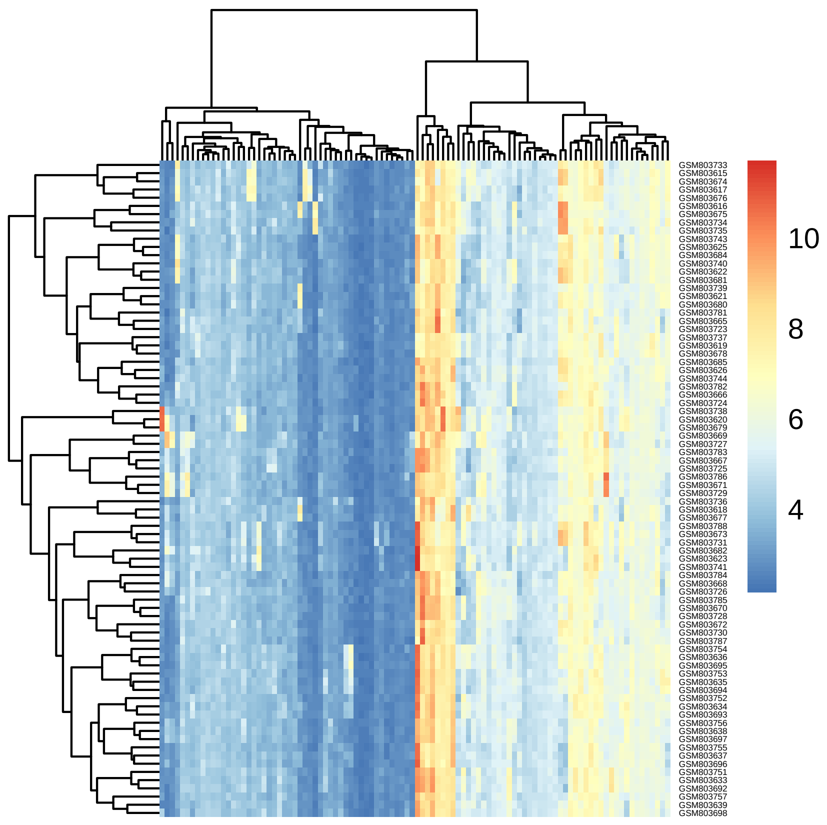
Another common way to represent this type of data is a *heatmap*. A heatmap consists of many little squares, one for each observation. We can use the `pheatmap` function in the `pheatmap` package [13].

We will plot a heatmap of the first 100 probes in the dataset[14]. `pheatmap` automatically clusters and draws dendrograms for both the cell lines and the genes.

```
library(pheatmap)
# fontsize_row: the font size for the row
# (cell lines) labels border_color: the color
# of the borders, we set it to NA for no
# border for visual clarity
pheatmap(expr[, 1:100], fontsize_row = 3, border_color = NA)
```

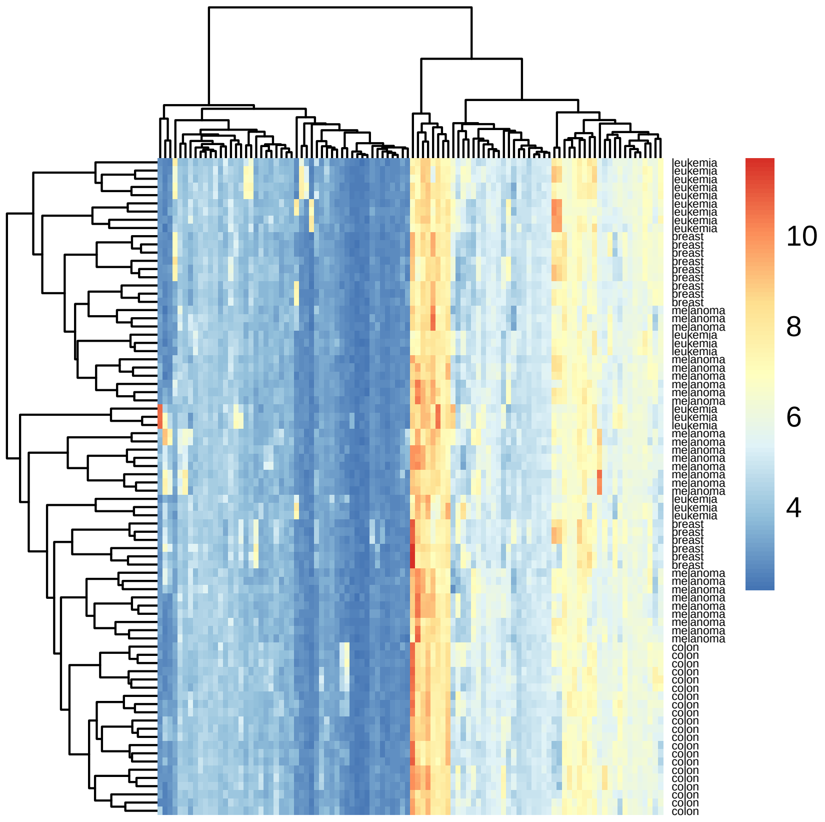
[13] You should first install it using `install.packages("pheatmap")`

[14] **ATTENTION**: do not attempt to create a heatmap of the whole dataset, it is likely to crash your computer. You should be able to safely plot 1000-2000 genes though! Feel free to experiment with this, but save your work beforehand!



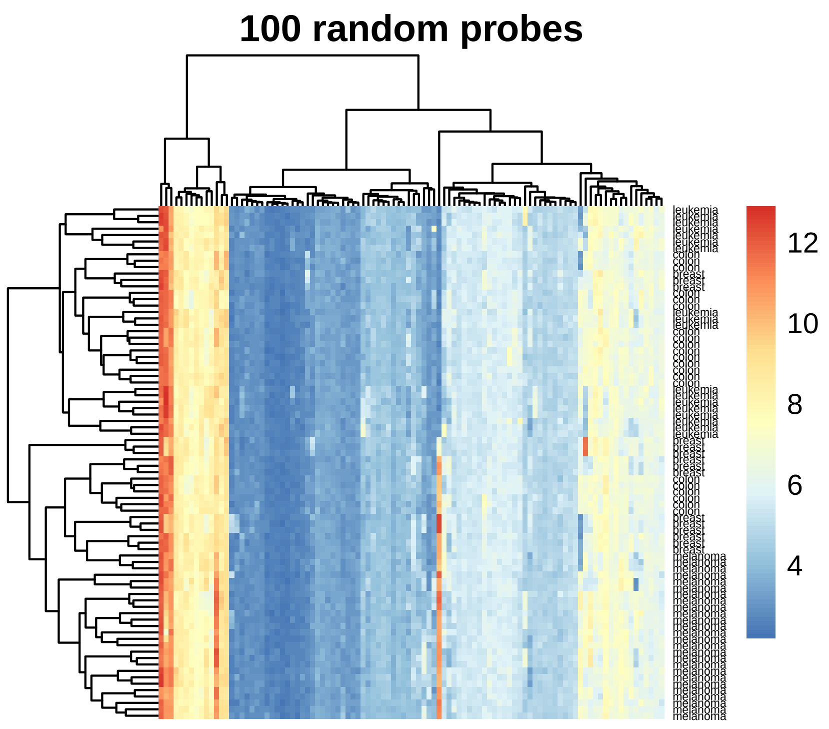
We can print the tissue of origin instead of the cell line name to make the plot more useful.

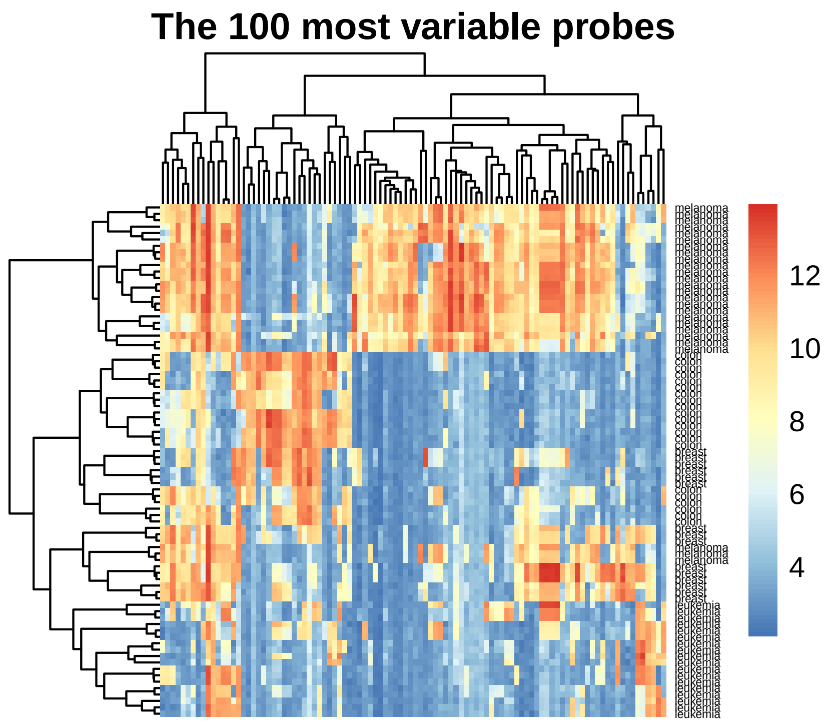
```r
rownames(expr) <- info$tissue
pheatmap(expr[, 1:100], fontsize_row = 4, border_color = NA)
```

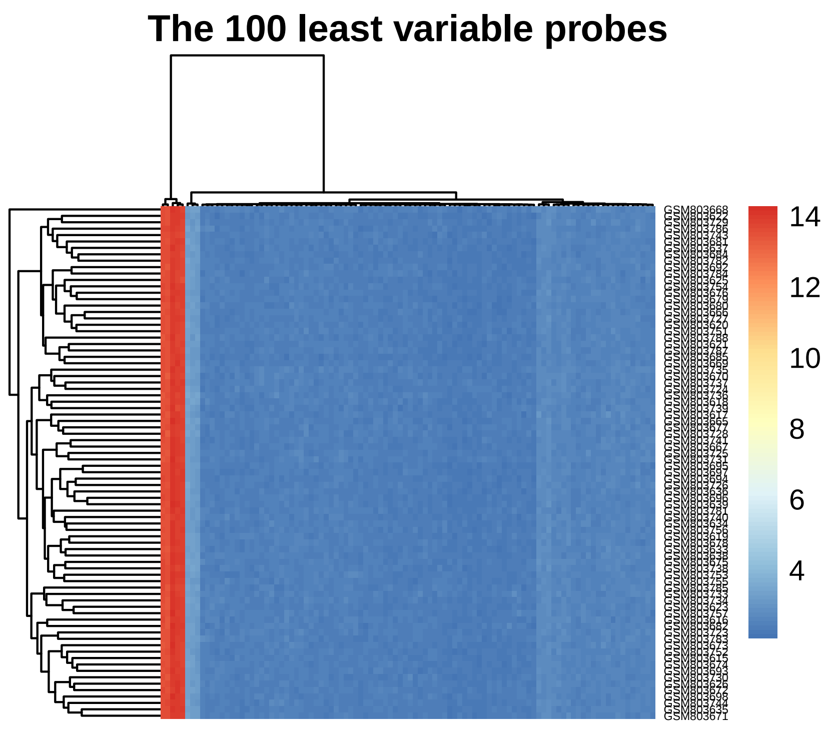


```r
# Revert to sample names
rownames(expr) <- info$sample
```

- What can you conclude from this plot?
- Following are similar plots done using a set of 100 random probes, or the 100 probes with more (or less) variability across samples. Is any of these plots more useful? Why? How do you interpret them?


100 random probes
12
10
8
6
4
leukemia
colon
breast
melanoma


The 100 most variable probes
12
10
8
6
4
melanoma
colon
breast
leukemia



PCA

So far, we have only used some of the probes from our dataset which, in this particular case, happened to give quite satisfactory results; however, what if some interesting difference is present in the other >50000 columns that we did not consider?

You have learned about dimension reduction techniques, specifically about PCA. Let's try it on our data!

Running PCA in R is extremely simple, using the `prcomp` function. Remember that our data needs to be centered around 0; we could do this manually, or ask `prcomp` to do it. We can also ask it to scale the data, so that each variable (gene) has a variance of 1. This is generally helpful to avoid variables with sustantially bigger variance to have more weight in the PCA output.

```r
# Note that we are using the whole dataset
# this time!
pca <- prcomp(expr, center = TRUE, scale = TRUE)
```

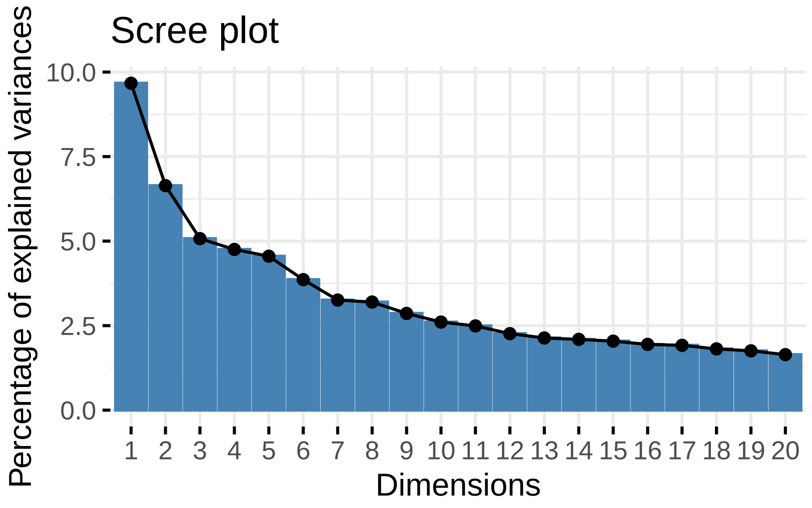
To plot the result of our PCA we will use the `factoextra` package. This provides a series of visualization functions (`fviz_...`) to create nice PCA plots[15].

We will start by looking at the scree plot[16] of the first 20 principal components.

[15] As before, install it using `install.packages(factoextra)`

[16] Look back at the lecture if you do not remember what that is!

```r
library(factoextra)
# Scree plot
fviz_eig(pca, ncp = 20)
```

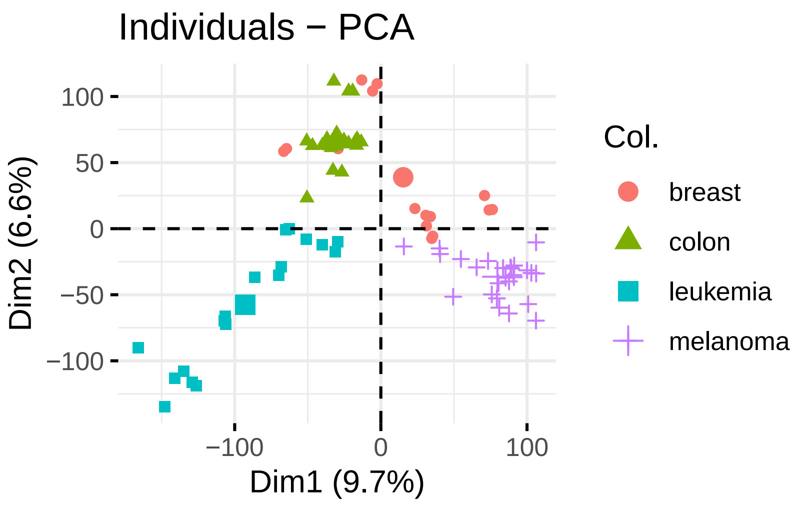


- How do you interpret this plot?
- How much variance is explained by the first 10 PCs[17]?.

[17] You can find the variance associated to each PC by squaring `pca$sdev`

Now let's plot the first two components.

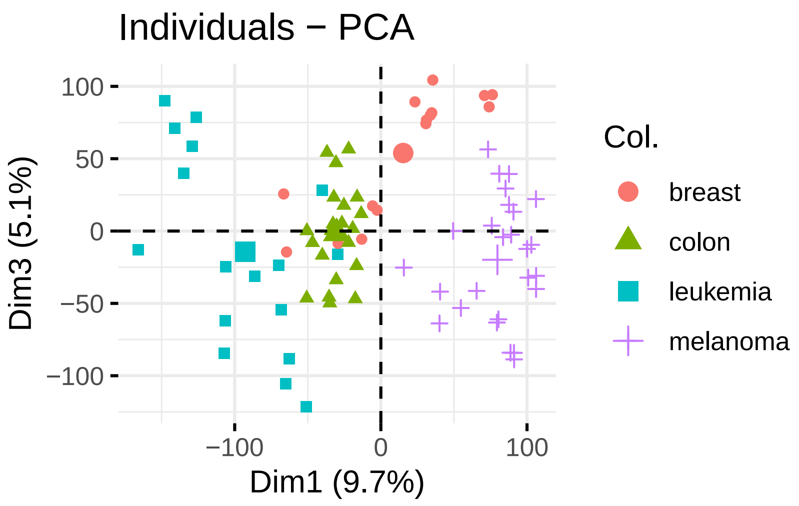
```r
fviz_pca_ind(pca, col.ind = info$tissue, geom.ind = "point",
    axes = c(1, 2))
```



You can see that cells mostly cluster in separate groups, with the exception of some breast cancer lines. The function also plots the center of each group as a data point with bigger size; this is helpful if you want to look for general changes in your data.

We can visualise other components as well

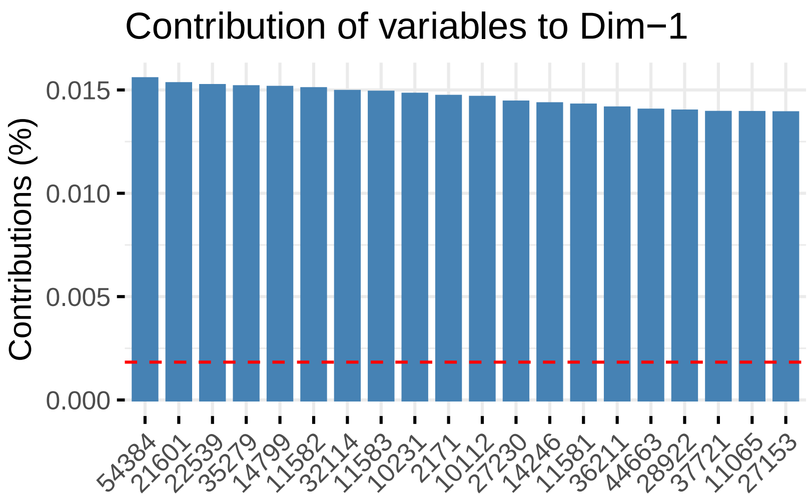
```
fviz_pca_ind(pca, col.ind = info$tissue, geom.ind = "point",
    axes = c(1, 3))
```



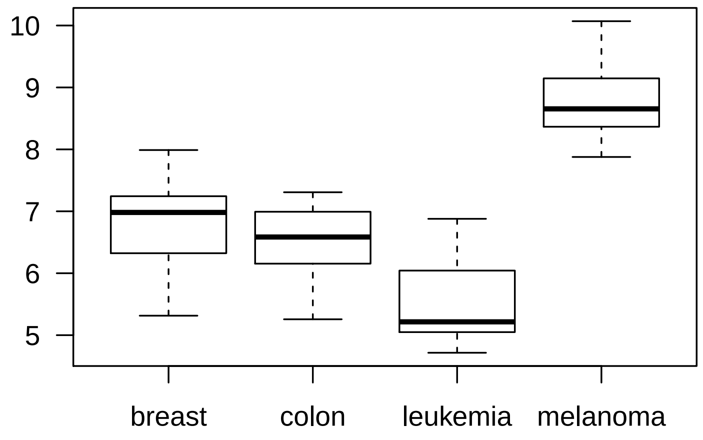
- How does this plot compare to the previous one?
- Try to plot other PCs. What do you observe when plotting PCs with higher number? Why?

Finally, we can also ask R to tell us which variables are the ones that most contribute to a certain PC.

```
fviz_contrib(pca, choice = "var", top = 20, axes = 1)
```



```
boxplot(expr[, 54384] ~ info$tissue, las = 1)
```



In this example, variable 54384 (corresponding to gene SASH1) is the one that most discriminates over PC1, albeit accounting for just a very small percentage of the variance. By plotting it we see that it is indeed quite different between the different groups.

Further work

Further analysis of genomic data (microarray, RNA-seq, CHIP-seq, etc) can be quite complex and definitely out of the scope of this workshop. It may involve differential gene expression analysis, to identify genes that are over- or underexpressed in specific clusters, as well as pathway analysis to see if component of specific molecular pathways are expressed in certain clusters or conditions. This can invove using gene ontology (GO) descriptors[18].

Examples of how to perform these can be found here[19]:

- `http://bioconductor.org/packages/devel/bioc/vignettes/DESeq2/inst/doc/DESeq2.html`
- `https://bioconductor.org/packages/release/bioc/vignettes/topGO/inst/doc/topGO.pdf`

[18] See `https://en.wikipedia.org/wiki/Gene_ontology` and `http://geneontology.org/`

[19] Note that these are just some of the possible analysis that can be performed, there is a lot more!