SysBio 2020 - Advanced Computational Systems Biology course

The following exercises are meant to be performed along the afternoon practical session.

You're encouraged to follow good coding practices and learn to find solutions to the problems independently (hints and examples will be given). You are also encouraged to set up a local `git` repository to keep track of your changes. You may do that by opening a terminal on your working directory and typing:

```
git init [name_of_your_repository]
```

You can commit your changes as you see fit (every functional change) or after each exercise for example, with:

```
git add [name_of_the_file/path]
git commit -m "[commit_message]"
```

Example solutions will be provided at the end of the day.

Exercises day 2 - Exploratory analysis and visualization of data

Today we will focus on basic summary-statistics and data visualization using R.

1) Loading packages and data

Load the expression and annotation data we prepared in yesterday's practice.
Also, install and load the packages `Rtsne` and `RColorBrewer`.
Note that these packages are not from Bioconductor but from CRAN (general R package repository). Therefore must be installed with `install.packages()` function. Remember to load these packages in your environment once installed with the `library()` function.

For today's exercise we will use the data we prepared yesterday: the expression data as well as the sample annotations. You can load these files with the `read.csv()` function.

2) Subsetting the data

For today's purposes, we will only work with the treated expression data (e.g. only the 10000nM drug dose). Subset the expression and the annotation data to contain only the samples corresponding to that dose. *You can easily achieve this by copy-pasting a fragment of yesterday's code (exercise 4) with some slight modifications*.

3) Preparing a color palette

Data visualization (e.g. plots) is a very important aspect to show any kind of data analysis result. Corollary, selecting a proper color scheme is also important. In the following exercises, we will try to visualize the different

tissue origins from our cell-lines (categorical data). Therefore, we need a palette of colors for each tissue as differentiable as possible (instead of some color gradient for continuous variables like expression for example). To achieve this, use the pre-defined `'Set1'` color palette provided by the package `RColorBrewer` to generate a list of colors with the same size as tissues we have in our data. *To get this color palette, use the function* `brewer.pal()` *from the aforementioned package*.

4) Generating barplots

As mentioned, one interesting categorical feature of our data set is the tissue origin of the cell-lines. Generate a barplot (with `barplot()` function) summarizing the number of samples we have for each tissue (colored).

5) Basic statistics

Compute some basic descriptive statistics from our samples (mean, median and SD) and plot them. *As you may know, this is usually shown as mean/median along with error bars +/- SD*.

6) Boxplots

You can show similar information along with quantiles and outliers using boxplots (function `boxplot()`). If you didn't before, now do this with the tissue colors for each sample.

7) Histogram and density plots

Select one of the samples (e.g. first one) and plot the histogram and the density function of its expression data. Discuss about data distribution. *You can plot the histogram with the* `hist()` *function and add the density function on top with* `lines(density())`

8) Heatmap with hierarchical clustering

Plot the expression profiles of our samples using a heatmap (`heatmap()` function). Luckily, this base R function also computes de hierarchical clustering of the data automatically adding the dendogram to the plot (hierarhical clustering by default). Add on top of it the colours for each tissue of origin to add more information. *For better visualization of continuous data you can choose your own color palette for the continuous expression data (e.g.* `col=brewer.pal(11, 'RdTlGn')` *)*

9) Correlation and scatter plots

From the previously generated plots, Leukemia-origin cell-lines seem to cluster together (and we have exactly 2 samples). Generate a scatter plot of these two samples and compute their correlation.

10) Principal Component Analysis

PCA is a **dimensionality reduction** algorithm (not clustering). Even though it's commonly used to explore underlying structures in multidimensional data ("clusters"). Plot the first two PC's and color the sample points by their tissue of origin to see if there's some underlying structure about them in the expression data. *We will define our own function for this based on the result of base R's function* `prcomp()`*. NOTE: you must transpose the dataframe*.

11) t-SNE

Similarly, t-SNE [1] is also a dimensionalty reduction algorithm which is being widely used nowadays to find potential structures in the data. One of its major pros is the ability to find non-linear relationships across multi-dimensional data points. In order to apply this algorithm use the package `Rtsne` which implements such algorithm and plot the first two dimensions colored by tissue of origin. You may run it a couple of times and with different perplexity parameter values to see the different results. *As with the PCA, the dataframe must be transposed*

> You can find a nice web to visualize the effect of parameter choices here:
> https://distill.pub/2016/misread-tsne/

References

1. Maaten L et al. Visualizing data using t-SNE. J Mach Learn Res 2008 Nov; 9:2579-2605.