<u>**Exercise 9**</u>
Basic graphs

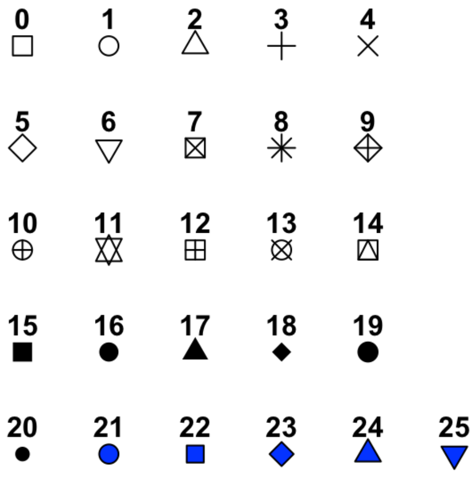
Exercise 9a – scatter plot

1. Read in file "/users/bi/public-docs/sbonnin/Rcourse/gene_counts.txt" in object **genes.**
Note: this file contains a header.

2. Create a scatter plot showing **sample1** (x-axis) vs **sample2** (y-axis) of **genes**.
plot()

3. Change the point type and colors.
*Note: see options **pch** and **col.***

*example of **pch:***



*For **col**, you can use:*
- integers (1, 2, 3 etc.) but limited to 8 different colors!
*- or pick up the name of a color using: **colors()**; for color picked randomly, you can use sample(colors(), 1).*

4. Change axis labels to "**Sample 1**" and "**Sample 2**".
*Note: see options **xlab** and **ylab**.*

5. Add a title to the plot.
*Note: see option **main**.*

6. Add a **red** vertical line at the **median** expression value of sample 1.
    a. calculate the median expression of genes in sample 1.
    median()
    b. plot a vertical line using **abline**().
*Note: **plot(....)** must be called before **abline()** is called. abline is not an argument but a function!*
See abline help page.

Exercise 9b – bar plot + pie chart

1. Read in file “/nfs/users/bi/public-docs/sbonnin/Rcourse/genes_counts_significance.txt” in object **de**.
Note: this file contains a header.

2. The column **updown** specifies whether a gene is up- or down- regulated, or neither of them (none).
Produce a **barplot** displaying this information: how many genes are up- or down-regulated, or neither of them.
barplot(table())

3. Color the bars of the boxplot, each in a different color: up in red, down in blue, none in grey.

4. Produce the same plot, but this time set argument **space** to 0.
What is **space** used for?

5. Now use option **names.arg** in **barplot**() to rename the bars.

6. The **las** argument allows you to rotate labels for a better visibility.
Try it!
Note: try values 0, 1, 2, 3. Observe how labels on both axis are oriented.

4. Create a pie chart of the same information (up, down, none)
pie(table())
*Note: Try arguments **color**, **main** and **labels**.*

Exercise 9c – histogram

1. Using **genes** object from exercise 9a, create a **histogram** of the gene expression distribution of sample 1.
hist()

2. Repeat the histogram but change argument **breaks** to 50.
What is the difference?

3. Color this histogram in light blue.
Note: there is color called “lightblue”

4. Overlap a second histogram to that first one, corresponding to sample 2 expression values.
*Note: you should call **hist()** once for sample 1, once for sample 2. The second histogram called should have option **add** set to **TRUE**. Read help page of **hist**.*

5. “Zoom” in the histogram: show only the distribution of expression values from 7 to 12 (x-axis).
*Note: use **xlim** option. Adjust also **ylim** if necessary for a better visibility.*

6. Save plot in a pdf file.
a. Try with RStudio Plots window (Export)

b. Try in the console:
pdf(...)
hist(...)
dev.off()

__Exercise 10.__
Introduction to ggplot2.

Exercise 10a – scatter plot

1. Load **ggplot2** package.

2. Using **de** object from Exercise 9, create a simple scatter plot for plotting gene expression of sample 1 and sample 2.
Note: remember the structure:
ggplot(dataframe, aes(x=, y=)) + geom_point()

3. Color points according to the **updown** column. Save in object **p**.
Note: remember the structure:
p <- ggplot(dataframe, aes(x=, y=, ***color=****)) + geom_point()*

4. Change colors of the points to blue, grey and red. Save to p2.
p2 <- p + ***scale_color_manual****()*

5. Save p2 into a jpeg file.
a. Try with RStudio Plots window (Export)
b. Try in the console:
jpeg(...)
plot(...)
dev.off()

Exercise 10b – box plot

1. Convert **de** from a wide format to a long format, save in **de_long**.
Note: remember ***melt*** *function from* ***reshape2*** *package.*
de_long <- melt(de)

2. Produce a boxplot of the expression of sample 1 and sample 2 (each sample should be represented by a box)
ggplot(dataframe, aes(x, y)) + geom_boxplot()

3. Modify the previous boxplot so as to obtain 3 "sub"-boxplots <u>per sample</u>, each representing the expression of either UP, DOWN or NONE genes.
ggplot(dataframe, aes(x, y, col)) + geom_boxplot()

Exercise 10c – bar plot

1. Produce a bar plot of how many UP/DOWN/NONE genes are in **de**.
ggplot(de, aes(x=)) + geom_bar()

2. Add an horizontal line at counts **1000.**
geom_hline()

3. Swap x and y axis.
coord_flip()

4. Add a title to the graph.
ggtitle()

Exercise 10d – histogram

1. Create a simple histogram using **de_long**.
ggplot(de_long, aes(x=)) + geom_histogram()

2. Notice that you get the following warning message *"stat_bin()` using `bins = 30`. Pick better value with `binwidth`."*
Change **bins** parameter in **geom_histogram**() to **50**.

3. This histogram plots expression values for both sample1 and sample2.
Change the plot so as to obtain 2 histograms on the same plot: one corresponding to sample1, one corresponding to sample2.
*ggplot(...**fill**=) + geom_histogram()*

4. By default, geom_histogram produces a stacked histogram.
Change the position of the bars to **dodge**.
ggplot(...fill=) + geom_histogram(position=...)

5. Zoom in the plot: reduce the x-axis to values from 7 to 12.
xlim() layer.

6. Finally, change the colors of the bars to colors of your choice.
scale_fill_manual()
Note: Try the rainbow() function for coloring!