Case: Golub data

Background

The Golub data set contains Gene expression values from 3051 genes taken from tumors sampled from 38 leukemia patients. Twenty-seven of the patients were diagnosed with *acute lymphoblastic leukemia* (ALL) and the remaining eleven patients with *acute myeloid leukemia* (AML).

We will investigate which gene expressions that significantly differ between the two tumor populations. This data set is well-known in bioinformatics and is often used for classification studies.

Variables

variable name	description
`golub`	gene expression values

As we are not interested in the actual genes, data is stored as a raw matrix with 3051 rows and 38 columns. Columns 1-27 are tumors from ALL patients, and columns 28-38 are tumors for AML patients.

Exercise

We consider the null hypotheses

$$\begin{aligned} H_1 &: \mu_{\text{AML},1} = \mu_{\text{ALL},1} \\ H_2 &: \mu_{\text{AML},2} = \mu_{\text{ALL},2} \\ &\vdots \\ H_{3051} &: 1\mu_{\text{AML},3051} = \mu_{\text{ALL},3051} \end{aligned}$$

where $\mu_{a,i} = \text{E}[\text{gene expression of gene } i \text{ for group } a]$.

- Use `par(mfrow = c(1,2))` to make R split the plot window into two panes. Randomly select a few genes, make a boxplot of gene expressions for AML and ALL tomurs in the left and right windows, respectively.

One can discuss which test to use, here we will use a standard t-test.

- If there were no differences between the two populations, what is the expected number of rejected hypotheses on a 5% significance level?
- Perform a standard t-test for each of the hypotheses.
- Sort the p-values and plot them from smallest to largest. Look at shape of the curve. Is there evidence for difference between ALL and AML tumors? (Hint: if there were no difference between the populations, how would the p-values behave?)
- Adjust the p-values using the Benjamini-Hochberg method.
- Report the number of rejected hypotheses on 10%, 5%, 1% and 0.1% significance levels, before and after adjustment. Store these numbers into a table in a text document.

Inspiration:

```
# To get the p-value from a t-test:
t.test([...], [...], var.equal = TRUE)$p.value

# Split the plot window into two panes:
par(mfrow = c(1,2))

plot([...]) # Left pane
plot([...]) # right pane

par(mfrow = c(1,1)) # Only one pane

# Sample m numbers from 1:n:
sample(n, m)

# Adjust p values
?p.adjust # Help page
```