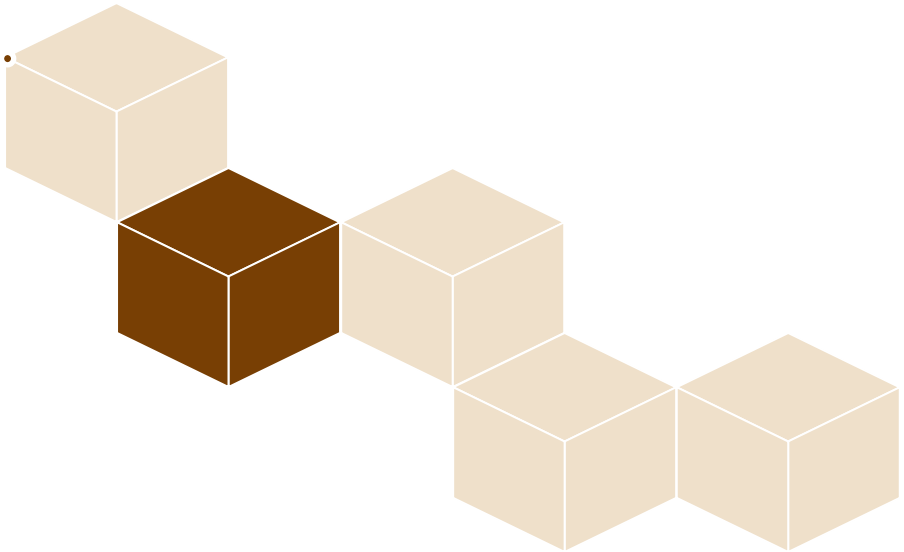
Session 2



rowSums()

In R, the function `rowSums()` conveniently calculates the totals for each row of a matrix. This function creates a new vector:

```
rowSums(my_matrix)
```

cbind() and rbind()

You can add a column or multiple columns to a matrix with the cbind() function, which merges matrices and/or vectors together by column. For example:

```
big_matrix <- cbind(matrix1, matrix2, vector1 ...)
```

Just like every action has a reaction, every cbind() has an rbind().

Selection of matrix elements

Similar to vectors, you can use the square brackets `[ ]` to select one or multiple elements from a matrix. Whereas vectors have one dimension, matrices have two dimensions. You should therefore use a comma to separate the rows you want to select from the columns. For example:

```
my_matrix[1,2] selects the element at the first row and second column.
my_matrix[1:3,2:4] results in a matrix with the data on the rows 1, 2, 3 and columns 2, 3, 4.
```

Arithmetic with matrices

Similar to what you have learned with vectors, the standard operators like `+`, `-`, `/`, `*`, etc. work in an element-wise way on matrices in R.

For example, `2 * my_matrix` multiplies each element of `my_matrix` by two.

Exercise

- Create a vector `u` with values ranging from 26 to 50
- Create a vector `v` with values ranging from 51 to 75
- Create a matrix `w` and assign the values of `x` and `y`, divide the matrix into 5 columns

Factors

The term factor refers to a statistical data type used to store categorical variables. The difference between a categorical variable and a continuous variable is that a categorical variable can belong to a limited number of categories. A continuous variable, on the other hand, can correspond to an infinite number of values.

What's a factor and why would you use it?

To create factors in R, you make use of the function `factor()`. First thing that you have to do is create a vector that contains all the observations that belong to a limited number of categories. For example, `sex_vector` contains the sex of 5 different individuals:

```
sex_vector <- c("Male","Female","Female","Male","Male")
```

It is clear that there are two categories, or in R-terms `'factor levels'`, at work here: `"Male"` and `"Female"`.

The function `factor()` will encode the vector as a factor:

```
factor_sex_vector <- factor(sex_vector)
```

Factor levels

When you first get a data set, you will often notice that it contains factors with specific factor levels. However, sometimes you will want to change the names of these levels for clarity or other reasons. R allows you to do this with the function `levels()`:

```
levels(factor_vector) <- c("name1", "name2",...)
```

Summarizing a factor

`summary` is a generic function used to produce result summaries of the results of various model fitting functions. The function invokes particular methods which depend on the class of the first argument.

```
summary(my_var)
```

Ordered factors

To create an ordered factor, you have to add two additional arguments: ordered and levels.

```
factor(some_vector, ordered = TRUE, levels = c("lev1", "lev2" ...))
```

By setting the argument ordered to `TRUE` in the function `factor()`, you indicate that the factor is ordered. With the argument `levels` you give the values of the factor in the correct order.

Comparing ordered factors

The fact that factor is now ordered enables us to compare different elements. You can simply do this by using the well-known operators.

```
factor_ordered_1 > factor_ordered_2
```

Break

15 minutes

Data frames

The function `data.frame()` creates data frames, tightly coupled collections of variables which share many of the properties of matrices and of lists, used as the fundamental data structure by most of R's modeling software.

Creating a data frame

You construct a data frame with the `data.frame()` function. As arguments, you pass vectors: they will become the different columns of your data frame. Because every column has the same length, the vectors you pass should also have the same length. But don't forget that it is possible (and likely) that they contain different types of data.

Selection of data frame elements

Similar to vectors and matrices, you select elements from a data frame with the help of square brackets [ ]. By using a comma, you can indicate what to select from the rows and the columns respectively. For example:

- `my_df[1,2]` selects the value at the first row and second column in `my_df`.
- `my_df[1:3,2:4]` selects rows 1, 2, 3 and columns 2, 3, 4 in `my_df`.

Sometimes you want to select all elements of a row or column. For example, `my_df[1, ]` selects all elements of the first row.

Sorting

In data analysis you can sort your data according to a certain variable in the data set. In R, this is done with the help of the function `order()`.

`order()` is a function that gives you the ranked position of each element when it is applied on a variable, such as a vector for example:

```
a <- c(100, 10, 1000)
order(a)
[1] 2 1 3
```

Exercise

- Create height and weight vectors of 5 people in the `height` and `weight` variables respectively.
- Create vector `gender` that stores the sex of the 5 people (`male`, `female`).
- Create a data frame with `height`, `weight`, and `gender` vectors.
- Print the data frame.
- Change the `gender` column of the data frame to a factor.
- Order the `gender` factor as follows: `female`, `male`.
- Print `gender` column

Homework 1

Parte I: En un notebook de R desarrolle los siguientes pasos:

- Construya una matriz de 5 x 5 con valores aleatorios decimales que vayan en un rango de 160 a 200. Esta matriz contendrá los pesos (en libras) de varias personas. Asigne un nombre descriptivo a esta matriz.
- Asigne los meses enero a mayo como nombre de cada columna.
- Asigne 5 nombres de persona, que sean aleatorios y que se utilicen como nombre de cada fila.
- Construya un vector que contenga el peso promedio de cada persona durante los meses de enero a mayo. (Investigue una función similar a `rowSums` pero para promediar)
- Construya un vector que contenga el peso promedio en cada mes para todas las personas. (Investigue una función similar a `colSums` pero para promediar)
- Utilice `cbind` y `rbind` para agregar la columna y fila promedio.
- Por último imprima la matriz resultante.

Homework 1

Parte II

Entregar en formato PDF una pequeña investigación de GIT que incluya lo siguientes apartados:

- Que es GIT
- Control de versiones con GIT
- Estados de un archivo en GIT
- Como se configura un repositorio
- Comandos en GIT

El entregable de la tarea será el siguiente:

- Crear una cuenta en Github
- Crear un repositorio con el nombre **Homeworks-R**
- Dentro del repositorio crear la carpeta **Homework-1**
- Cargar el R notebook y la investigación en PDF

Enviar enlace del repositorio a más tardar: **viernes 26-mayo a las 23:59**