

# An introduction to the tidyverse

Ziv Shkedy et al. (2020)

# **Part 1: Introduction**

Focus: a quick introduction to tidyverse

# The tidyverse

- The `tidyverse` is a collection of R packages that work in data frames in a tidy format.
- All packages are uploaded in CRAN, and can be installed using `install.packages`.
- In this chapter of the interactive book, we cover materials at an introduction level and follow closely the topics presented in Chapter 4 in the book Data Analysis and Prediction Algorithms with R by Rafael A. Irizarry.

# What do we cover in this chapter ?

- The chapter is developed at a beginner level.
- We cover few functions from the `tidyverse` packages and illustrate the basic concepts using different examples of the following functions:
  - `mutate()`
  - `filter()`
  - `select()`
  - The pipe: `% > %`
  - `summarize()`
  - `group_by()`
  - `arrange()`
  - `top_n()`

# ggplot2?

- Our aim in this tutorial is not to teach `ggplot2` (this will be done in a different chapter).
- Some functions of the package are used to visualize the main patterns in the datasets we used to illustrate the examples presented in this chapter.
- The following graphical functions are used for visualization:
  - `qplot()`
  - `ggplot() + geom_jitter()`
  - `ggplot() + geom_point()`
  - `gplot() + geom_density()`
  - `stripplot()`

# Online references (I)

- Materials about `tidyverse` are widely available online.
  - For a YouTube tutorial about tidyverse in R by Mark Gingrass see [YTtidyverse1](#).
  - For a YouTube tutorial about tidyverse in R by Garreer Grolemund see [YTtidyverse2a](#).
  - For a YouTube tutorial about tidyverse in R by Garreer Grolemund see [YTtidyverse2b](#).
  - For a YouTube tutorial about tidyverse in R by Ben Stenhaug see [YTtidyverse3](#).

# Online references (II)

- Interactive book for the course: [Booktidyverse1](#)
- Chapter 4 in the book: Data Analysis and Prediction Algorithms with R by Rafael A. Irizarry see [Chapter4](#).

# Part 2: the tidyverse package and tidy data

Focus: the center of the distribution

Graphical and numerical summaries

# Tidy data

- Tidy data is a data format in which each row represents one measurement for one observation and columns are, as usual, the variables in the data.

# The murders data

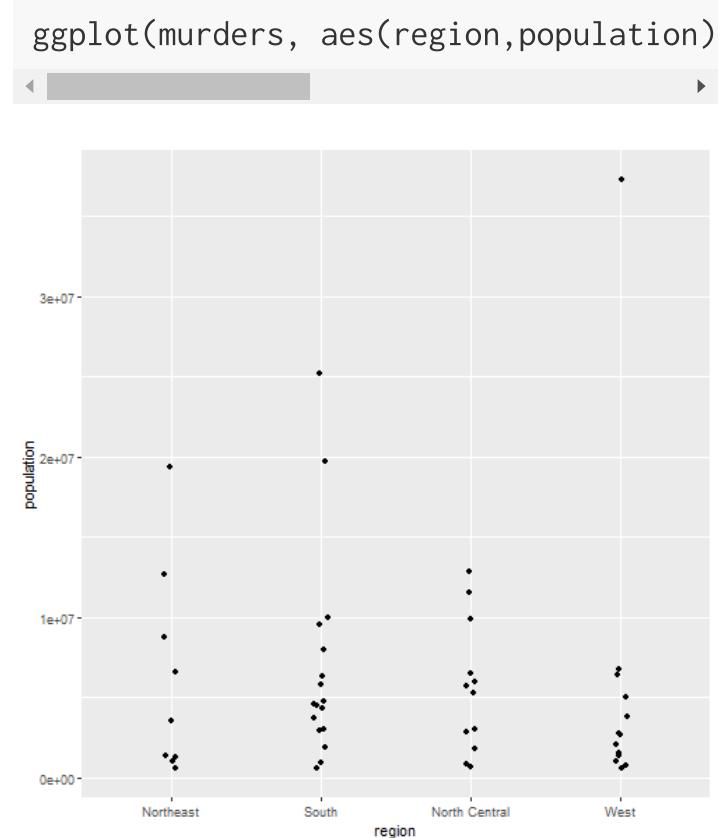
- The murders is an example of a tidy data.
- The information for each state is given in one line.
- Note that in this case, each observation (=state) has information in one data line.

```
data("murders")
head(murders)
```

```
##          state abb region population total
## 1      Alabama  AL   South    4779736   135
## 2      Alaska  AK   West     710231    19
## 3      Arizona  AZ   West    6392017   232
## 4      Arkansas AR   South    2915918    93
## 5 California CA   West    37253956  1257
## 6 Colorado CO   West    5029196    65
```

# The murders data

- The murder rate by region is shown in the stripplot.
- It shows clearly that in the west, the murder rate is the lowest.



# The ChickWeight data

- The Chicken Weight data gives information about weights of chicken in different diet groups.
- Each observation is a chick and it was measured in 12 time points.

```
names(ChickWeight)
```

```
## [1] "weight" "Time"   "Chick"   "Diet"
```

```
unique(ChickWeight$Time)
```

```
## [1]  0  2  4  6  8 10 12 14 16 18 20 21
```

- In the data frame, each measurement is presented in one data line.
- Hence, the ChickWeight is a tidy data.

# The ChickWeight data

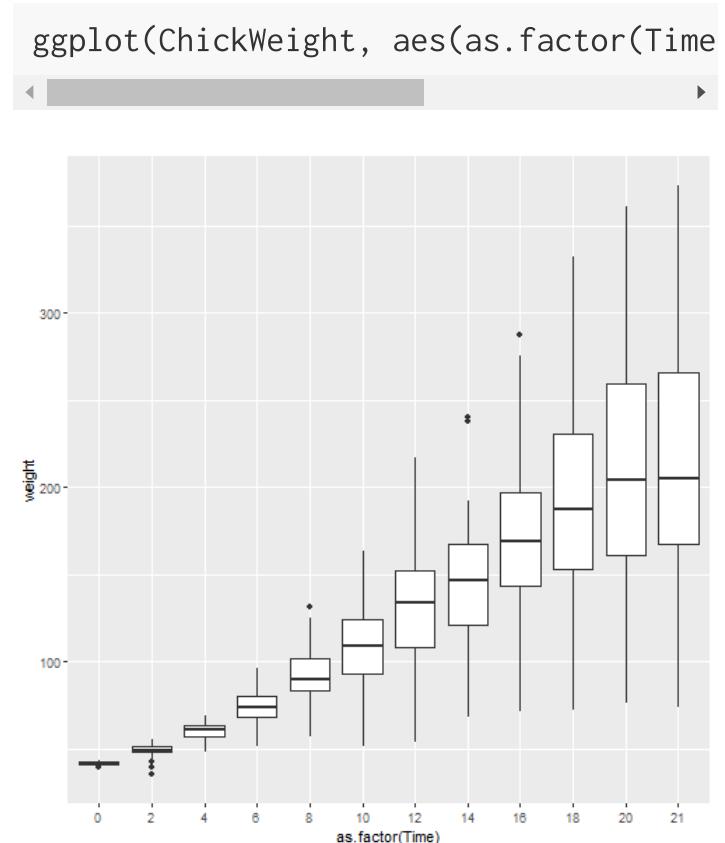
- This implies that that the data for each observation is presented in 12 lines.
- note that NOT ALL subjects were measured in 12 times.
- Data for the first 6 time points of the first chick are listed below.

```
head(ChickWeight)
```

```
## Grouped Data: weight ~ Time | Chick
##   weight Time Chick Diet
## 1     42     0     1     1
## 2     51     2     1     1
## 3     59     4     1     1
## 4     64     6     1     1
## 5     76     8     1     1
## 6     93    10     1     1
```

# The ChickWeight data

- The boxplot of the chicken weights by time point.
- It shows the increasing trend of the weight over time.



# **Part 3: the mutate()**

Focus: adding a column(s)

# Adding a variable (column)

- Suppose that we would like to calculate the murder rate per 100000 people that is

$$rate = \frac{total}{population} \times 100000.$$

- This can be done using the `mutate()` function.
- The function has the general call of: `mutate(data frame , new variable)` .

# The murders data

- For the `murders` data we have

```
data("murders")
murders <- mutate(murders, rate = total / population * 100000)
```

- Note that after calculating the murder rate, the `murders` has an extra column (=variable) for the `rate`.

```
head(murders)
```

```
##      state abb region population total      rate
## 1    Alabama  AL   South     4779736   135 2.824424
## 2     Alaska  AK    West      710231    19 2.675186
## 3   Arizona  AZ    West     6392017   232 3.629527
## 4  Arkansas  AR   South     2915918    93 3.189390
## 5 California  CA    West    37253956  1257 3.374138
## 6 Colorado  CO    West     5029196    65 1.292453
```

# The NHANES data

- The BMI of a person is given by

$$BMI = \frac{weight}{height^2}$$

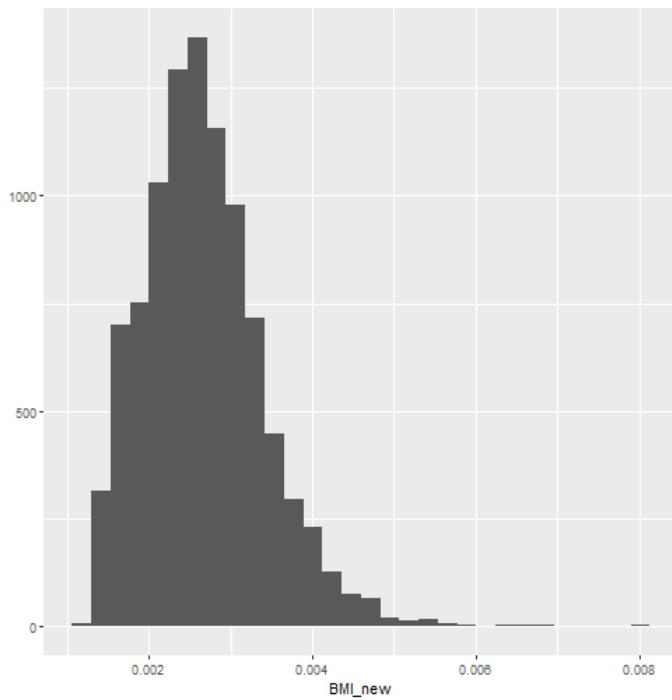
- To calculate the BMI in the NHANES we use

```
data("NHANES")
Data_new <- mutate(NHANES, BMI_new = Weight / (Height*Height))
```

# The NHANES data

- The histogram of the BMI.

```
qplot(BMI_new , data=Data_new, geom="h
```



# Practical session

- For the cars data ( `mtcars`):
  - Define a new variable: mile per gallon / weight.
  - Produce a boxplot for the variable mile per gallon / weight by number of number of cylinders.
- For the murders data :
  - Produce a boxplot for the murder rate by region.

# **Part 4: the filter() function**

Focus: selection of cases

# Filtering

- Selection of observations (=cases) from the data.
- A subset of the data.
- can be done using the function `filter()` .
- The new data: has the same variables.

# The murders data

- For murder dataset, we can select all the states with murder rate  $\leq 0.71$  by

```
filter(murders, rate <= 0.71)
```

```
##           state abb      region population total      rate
## 1        Hawaii HI       West    1360301     7 0.5145920
## 2         Iowa IA North Central 3046355    21 0.6893484
## 3 New Hampshire NH Northeast 1316470     5 0.3798036
## 4  North Dakota ND North Central 672591     4 0.5947151
## 5    Vermont VT Northeast 625741     2 0.3196211
```

# The cars data

- For cars data, suppose that we would like to plot the cars' weight versus the cars mpg for cars with weight  $\leq 3$ .
- We first select all cars with weight smaller or equal to 3:

```
dim(mtcars)
```

```
## [1] 32 11
```

```
mtcars1<-filter(mtcars, wt <= 3)
dim(mtcars1)
```

```
## [1] 12 11
```

# The cars data

- The new data frame `mtcars1` contains the information for all cars with weight lower than 3,

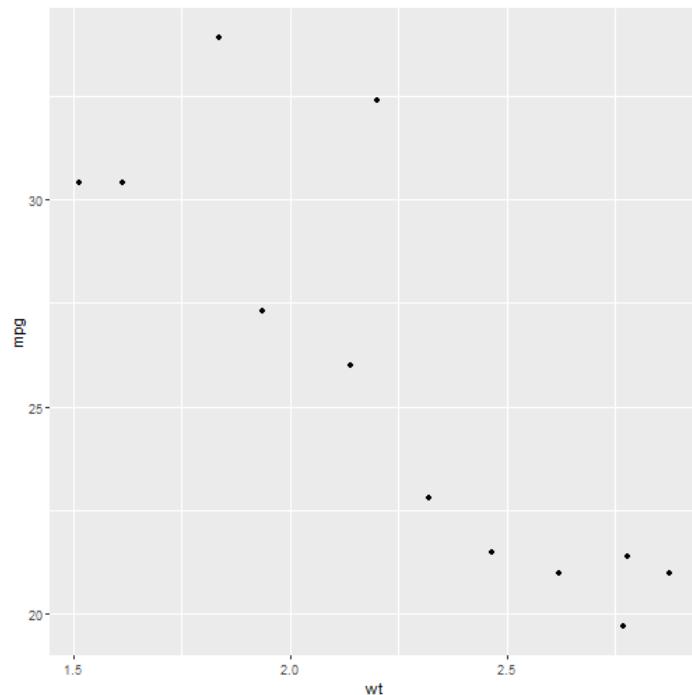
```
mtcars1
```

```
##          mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4   21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710  22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Fiat 128    32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona 21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Fiat X1-9    27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2 26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa 30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ferrari Dino 19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Volvo 142E   21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

# The cars data

- The scatterplot below of the weight versus the mpg can be produced using the following code.

```
ggplot(mtcars1, aes(x=wt, y=mpg)) +  
  geom_point()
```



# Practical session

- For the NHANES data:
  - Select all the observations for which the BMI is greater or equal to 30.
  - Produce the histogram for the variable BMI for all male.
- For the chick weight data (`chickwts`):
  - Select all chicks with weight smaller than 158.
  - Select all chicks from the horsebean feed group.

# **Part 5: the select() function**

Focus: selection of variables

# Part 6: the pipe % > %

Focus: multiple startments in one run

# Part 7: the summarize() function

Focus: summary statistics

# The summarize function

- The function `summarize()` allows us to produce summary statistics for variables in the data frame.

# The heights data

- The height data frame gives the height by gender for 1050 individuals

```
library(dplyr)
library(dslabs)
data(heights)
dim(heights)
```

```
## [1] 1050      2
```

- The first 6 observations in the data are listed below.

```
head(heights)
```

```
##      sex height
## 1   Male    75
## 2   Male    70
## 3   Male    68
## 4   Male    74
## 5   Male    61
## 6 Female   65
```

# Mean and standard deviation

- We can calculate the mean and standard deviation for female using the function `summarize ()` .
- Note that we first filter the data and define and subgroup contains the data for female

```
s <- heights %>% filter(sex == "Female") %>%
  summarize(average = mean(height), standard_deviation = sd(height))
```

- The object `s` stores the results

```
s
```

```
##      average standard deviation
## 1 64.93942          3.760656
```

# Mean and standard deviation

- We can define a vector that contains the results

```
c(s$average,s$standard_deviation)
```

```
## [1] 64.939424 3.760656
```

# Summary statistics by group

Alternatively, we can define a vector with the female heights ( `height.female` ) and calculate the mean and standard deviation for this vector.

```
height.female<-heights$height[heights$sex == "Female"]  
mean(height.female)
```

```
## [1] 64.93942
```

```
sd(height.female)
```

```
## [1] 3.760656
```

# Quantiles

- The median, minimum and maximum height for female

```
heights %>%
  filter(sex == "Female") %>% summarize(median = median(height), minimum = min(hei,
                                             maximum = max(height))

##      median minimum maximum
## 1 64.98031      51       79
```

# Quantiles

- These summary statistics can be also calculate can the function `quantile` .

```
heights %>% filter(sex == "Female") %>%
  summarize(range = quantile(height, c(0, 0.5, 1)))
```

```
##      range
## 1 51.00000
## 2 64.98031
## 3 79.00000
```

# The chicks data

- To calculate the mean and standard deviation for the chick weights we use

```
s <- chickwts %>% summarize(average = mean(weight), standard_deviation = sd(weigh  
s  
##   average standard deviation  
## 1 261.3099      78.0737
```

- Note that for this example we ignore the diet group.

# Practical session

- For the NHANES data:
  - Calculate the mean, median and standard deviation of the variable BMI.
  - Calculate the mean, median and standard deviation of the variable BMI only for male.
- For the cars data:
  - Calculate the mean, median and standard deviation of the variable weight.
  - Calculate the mean, median and standard deviation of the variable weight only for cars with automatic transmission.

# **Part 8: the group\_by() function**

Focus: summary statistics by group

# Analysis by group

- In this section we focus on an analysis in which the analysis is done across a level of a factor in the data frame.
- For example, the diet group in the chick data frame etc.

# The heights data

- The mean and standard deviation for the height by gender can be calculate by adding the function `group_by(sex)`

```
heights %>%
  group_by(sex) %>%
  summarize(average = mean(height), standard_deviati
```

```
## # A tibble: 2 x 3
##   sex     average standard_deviati
##   <fct>    <dbl>        <dbl>
## 1 Female    64.9         3.76
## 2 Male      69.3         3.61
```

# The heights data

- The same results can be obtained using the function `tapply` .

```
tapply(heights$height, heights$sex, mean)
```

```
##   Female     Male  
## 64.93942 69.31475
```

```
tapply(heights$height, heights$sex, sd)
```

```
##   Female     Male  
## 3.760656 3.611024
```

# The murders data

- The median murder rate by region using the `group_by(region)` and the `summarize ()` functions

```
murders %>% group_by(region) %>%
  summarize(median_rate = median(rate))
```

```
## # A tibble: 4 x 2
##   region      median_rate
##   <fct>        <dbl>
## 1 Northeast     1.80
## 2 South         3.40
## 3 North Central 1.97
## 4 West          1.29
```

# The murders data

The median murder rate by region using the `tapply()` function.

```
tapply(murders$rate,murders$region,median)  
  
##      Northeast          South       North    Central        West  
##      1.802179      3.398069      1.971105      1.292453
```

# The chicks data

- Summary statistics by diet group

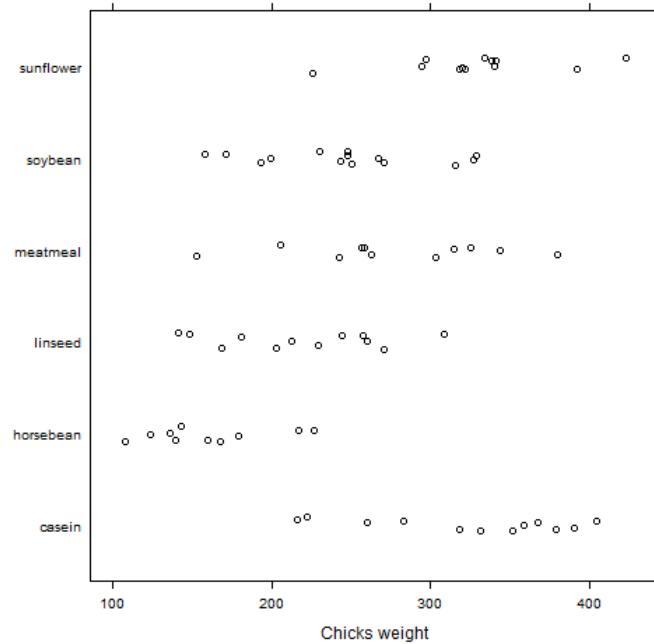
```
chickwts %>%  
  group_by(feed) %>%  
  summarize(average = mean(weight), standard_deviation = sd(weight))
```

```
## # A tibble: 6 x 3  
##   feed      average standard deviation  
##   <fct>     <dbl>          <dbl>  
## 1 casein     324.           64.4  
## 2 horsebean   160.           38.6  
## 3 linseed     219.           52.2  
## 4 meatmeal    277.           64.9  
## 5 soybean     246.           54.1  
## 6 sunflower   329.           48.8
```

# The chicks data

- The stripplot reveals that the weights in the horsebean group are, in general, the smallest in the sample.

```
stripplot(feed ~ jitter(weight),  
         data = chickwts,  
         aspect = 1, jitter = T,  
         xlab="Chicks weight", col
```



# Practical session

- For the NHANES data:
  - Calculate the mean BMI by gender.
  - Calculate the mean age by diabetes status (the variable Diabetes).
- For the cars data:
  - Calculate the mean, median and standard deviation of the variable weight by transmission type (automatic vs. manual).
  - Calculate the mean, median and standard deviation of the variable miles per gallon by number of cylinders.

# Part 9: the arrange() function

Focus: sorting the data

# The murders data

- We can sort a data frame by a variable `x` using the function `arrange(x)` .
- For the murder data frame, we sort the data by the population size

```
murders %>%  
  arrange(population) %>% head()
```

```
##           state abb      region population total      rate  
## 1        Wyoming WY        West     563626    5 0.8871131  
## 2 District of Columbia DC       South    601723   99 16.4527532  
## 3        Vermont VT Northeast  625741     2 0.3196211  
## 4       North Dakota ND North Central 672591     4 0.5947151  
## 5        Alaska AK        West    710231    19 2.6751860  
## 6       South Dakota SD North Central 814180     8 0.9825837
```

# The murders data

- The same sorting can be implemented using the function `order()` .
- In this case the rows will be presented in the order of the population.

```
data1<-murders[order(murders$population), ]  
head(data1)
```

```
##           state abb      region population total      rate  
## 51        Wyoming WY       West     563626    5 0.8871131  
## 9  District of Columbia DC       South    601723   99 16.4527532  
## 46        Vermont VT      Northeast  625741    2 0.3196211  
## 35        North Dakota ND  North Central 672591    4 0.5947151  
## 2         Alaska AK       West    710231   19 2.6751860  
## 42        South Dakota SD  North Central 814180    8 0.9825837
```

# Sorting and changing the order

- We sort the data frame by rate from the lowest to the highest rate

```
murders %>%  
  arrange(rate) %>%  
  head()
```

```
##           state abb      region population total      rate  
## 1       Vermont VT Northeast    625741     2 0.3196211  
## 2 New Hampshire NH Northeast   1316470     5 0.3798036  
## 3       Hawaii HI        West  1360301     7 0.5145920  
## 4 North Dakota ND North Central  672591     4 0.5947151  
## 5       Iowa IA North Central 3046355    21 0.6893484  
## 6       Idaho ID        West 1567582    12 0.7655102
```

# Sorting and changing the order

- We can change the order using the function `desc()` so the data are presented from the highest to the lowest rate in a decreasing order.

```
murders %>%
  arrange(desc(rate)) %>%
  head
```

```
##           state abb      region population total      rate
## 1 District of Columbia DC        South    601723   99 16.452753
## 2 Louisiana LA        South    4533372   351  7.742581
## 3 Missouri MO North Central  5988927   321  5.359892
## 4 Maryland MD        South    5773552   293  5.074866
## 5 South Carolina SC        South    4625364   207  4.475323
## 6 Delaware DE        South    897934    38  4.231937
```

# The Chicken Weight data

- The first 6 observations in the chicken weight data belongs to the first chick at time point 0 to 10.

```
head(ChickWeight)
```

```
## Grouped Data: weight ~ Time | Chick
##   weight Time Chick Diet
## 1     42     0     1     1
## 2     51     2     1     1
## 3     59     4     1     1
## 4     64     6     1     1
## 5     76     8     1     1
## 6     93    10     1     1
```

# The Chicken Weight data

- We sort the data frame according to the `Time` variable. After sorting, the first 6 lines in the data frame are the measurements for chick 1-6 at day 21.

```
ChickWeight %>%
  arrange(desc(Time)) %>%
  head
```

```
## Grouped Data: weight ~ Time | Chick
##   weight Time Chick Diet
## 1    205    21     1     1
## 2    215    21     2     1
## 3    202    21     3     1
## 4    157    21     4     1
## 5    223    21     5     1
## 6    157    21     6     1
```

# The Chicken Weight data

- We can reverse the order, in this case the first 6 lines are the measurements for check 1-6 at baseline ( `Time =0` ).

```
ChickWeight %>%
  arrange(Time) %>%
  head

## Grouped Data: weight ~ Time | Chick
##   weight Time Chick Diet
## 1     42    0     1     1
## 2     40    0     2     1
## 3     43    0     3     1
## 4     42    0     4     1
## 5     41    0     5     1
## 6     41    0     6     1
```

# Practical session

- Sort the NHANES data according to the subjects' BMI.
- For the NHANES data, select all subjects with diabetes (the variable Diabetes) and sort according to the subjects' age.
- Sort the cars data according to the cars' mpg.

# **Part 10: Nested sorting**

Focus: sorting with two variables

# Nested sorting

- Suppose that we want to present the data in an increasing order of  $x$  across a level of a factor  $y$  .
- We can sort the data frame by a variable  $x$  within the factor levels using the function `arrange(y,x)` .

# the murder data

- For the murder data frame, we sort the data by murder rate within the region

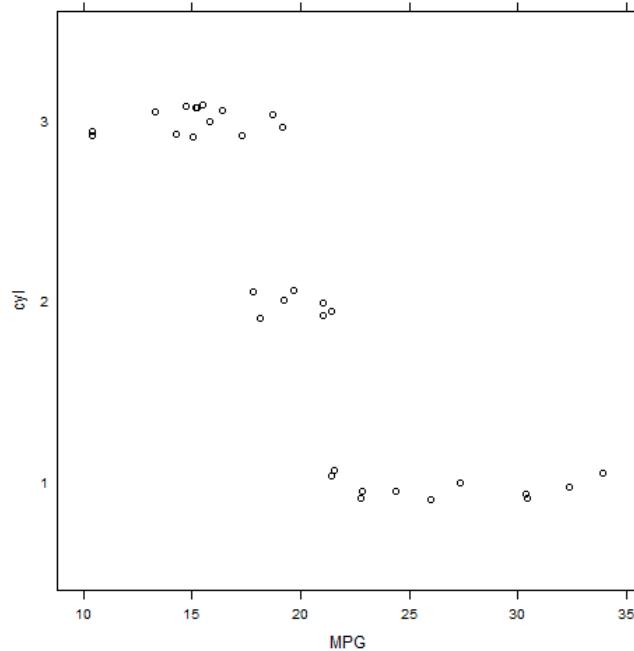
```
murders %>%
  arrange(region, rate) %>%
  head()
```

```
##           state abb   region population total      rate
## 1       Vermont VT Northeast    625741     2 0.3196211
## 2 New Hampshire NH Northeast   1316470     5 0.3798036
## 3       Maine ME Northeast   1328361    11 0.8280881
## 4 Rhode Island RI Northeast   1052567    16 1.5200933
## 5 Massachusetts MA Northeast   6547629   118 1.8021791
## 6       New York NY Northeast  19378102   517 2.6679599
```

# The cars data

- The figure shows that mpg as the number of cylinders decreases.

```
stripplot(cyl ~ jitter(mpg),  
         data = mtcars,  
         aspect = 1, jitter = T,  
         xlab="MPG", col = 1)
```



# The cars data

- We can sort the cars according to their mpg (in an increasing order) by the number of cylinders

```
mtcars %>%  
  arrange(cyl, mpg)
```

|                        | mpg  | cyl | disp  | hp  | drat | wt    | qsec  | vs | am | gear | carb |
|------------------------|------|-----|-------|-----|------|-------|-------|----|----|------|------|
| ## Volvo 142E          | 21.4 | 4   | 121.0 | 109 | 4.11 | 2.780 | 18.60 | 1  | 1  | 4    | 2    |
| ## Toyota Corona       | 21.5 | 4   | 120.1 | 97  | 3.70 | 2.465 | 20.01 | 1  | 0  | 3    | 1    |
| ## Datsun 710          | 22.8 | 4   | 108.0 | 93  | 3.85 | 2.320 | 18.61 | 1  | 1  | 4    | 1    |
| ## Merc 230            | 22.8 | 4   | 140.8 | 95  | 3.92 | 3.150 | 22.90 | 1  | 0  | 4    | 2    |
| ## Merc 240D           | 24.4 | 4   | 146.7 | 62  | 3.69 | 3.190 | 20.00 | 1  | 0  | 4    | 2    |
| ## Porsche 914-2       | 26.0 | 4   | 120.3 | 91  | 4.43 | 2.140 | 16.70 | 0  | 1  | 5    | 2    |
| ## Fiat X1-9           | 27.3 | 4   | 79.0  | 66  | 4.08 | 1.935 | 18.90 | 1  | 1  | 4    | 1    |
| ## Honda Civic         | 30.4 | 4   | 75.7  | 52  | 4.93 | 1.615 | 18.52 | 1  | 1  | 4    | 2    |
| ## Lotus Europa        | 30.4 | 4   | 95.1  | 113 | 3.77 | 1.513 | 16.90 | 1  | 1  | 5    | 2    |
| ## Fiat 128            | 32.4 | 4   | 78.7  | 66  | 4.08 | 2.200 | 19.47 | 1  | 1  | 4    | 1    |
| ## Toyota Corolla      | 33.9 | 4   | 71.1  | 65  | 4.22 | 1.835 | 19.90 | 1  | 1  | 4    | 1    |
| ## Merc 280C           | 17.8 | 6   | 167.6 | 123 | 3.92 | 3.440 | 18.90 | 1  | 0  | 4    | 4    |
| ## Valiant             | 18.1 | 6   | 225.0 | 105 | 2.76 | 3.460 | 20.22 | 1  | 0  | 3    | 1    |
| ## Merc 280            | 19.2 | 6   | 167.6 | 123 | 3.92 | 3.440 | 18.30 | 1  | 0  | 4    | 4    |
| ## Ferrari Dino        | 19.7 | 6   | 145.0 | 175 | 3.62 | 2.770 | 15.50 | 0  | 1  | 5    | 6    |
| ## Mazda RX4           | 21.0 | 6   | 160.0 | 110 | 3.90 | 2.620 | 16.46 | 0  | 1  | 4    | 4    |
| ## Mazda RX4 Wag       | 21.0 | 6   | 160.0 | 110 | 3.90 | 2.875 | 17.02 | 0  | 1  | 4    | 4    |
| ## Hornet 4 Drive      | 21.4 | 6   | 258.0 | 110 | 3.08 | 3.215 | 19.44 | 1  | 0  | 3    | 1    |
| ## Cadillac Fleetwood  | 10.4 | 8   | 472.0 | 205 | 2.93 | 5.250 | 17.98 | 0  | 0  | 3    | 4    |
| ## Lincoln Continental | 10.4 | 8   | 460.0 | 215 | 3.00 | 5.424 | 17.82 | 0  | 0  | 3    | 4    |

# Practical session

- Sort the NHANES data according to the subjects' BMI by Gender.
- Sort the NHANES data according to the subjects' age by smoking status (the variable `Smoke100`).
- Sort the Chicks Weights data (`chickwts`) according to the chicks' weight by diet group.

# **Part 11: The top\_n() function**

Focus: selection of the top K observations

# Top n

- To print the top  $n$  observations according to the variable  $x$  we can use the function `top_n(n, x)` .

# The murders data

- For the murders data, we print the top 5 states with the highest murder rate

```
murders %>% top_n(5, rate)
```

```
##           state abb      region population total      rate
## 1 District of Columbia DC        South    601723    99 16.452753
## 2 Louisiana LA        South   4533372   351  7.742581
## 3 Maryland MD        South   5773552   293  5.074866
## 4 Missouri MO North Central 5988927   321  5.359892
## 5 South Carolina SC        South   4625364   207  4.475323
```

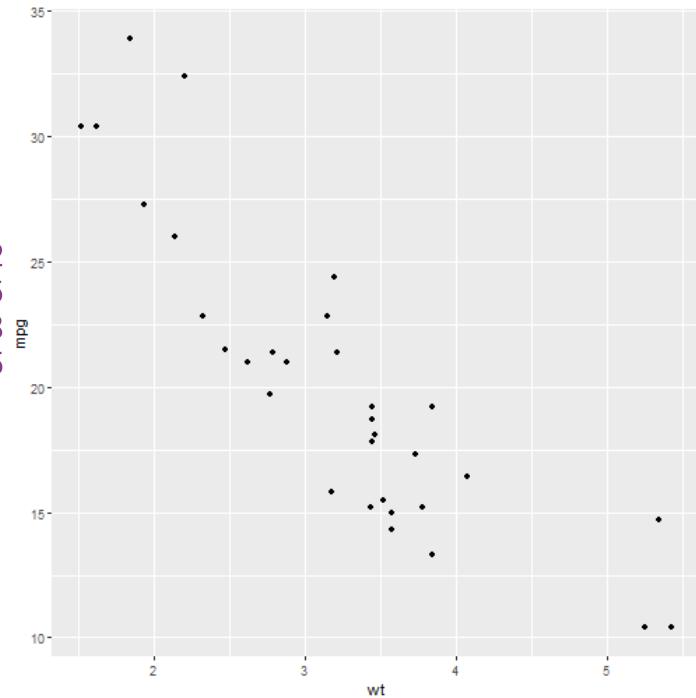
# The cars data

- The scatterplot of the cars' weight versus the cars' mpg.
- The top 4 cars, with the highest mpg are given below

```
mtcars %>% top_n(4,mpg)
```

```
##          mpg cyl disp  hp drat
## Fiat 128 32.4   4 78.7 66 4.08
## Honda Civic 30.4   4 75.7 52 4.93
## Toyota Corolla 33.9   4 71.1 65 4.22
## Lotus Europa 30.4   4 95.1 113 3.77 1.5
```

```
ggplot(mtcars, aes(x=wt, y=mpg)) +  
  geom_point()
```



# Practical session

- Print the 10 observations in the NHANES data with the highest BMI.
- Create a new data frame that contains the top 5 observations with the older age in in the NHANES data.

