Exercise 2

No Fear of Numbers: Introduction to Quantitative Data Analysis in R

Dr. Susanne de Vogel, Data Science Center, University of Bremen

2 December 2025

Research Question: Is there a Life beyond the PhD?

In this workshop, we will work with one coherent example throughout all four parts.We focus on two outcomes of interest:

- `bpsy01`: overall life satisfaction (10 point scale)
- `blcd06`: Children (yes/no)

We will examine how these outcomes are related to different aspects of doctoral researchers' lives and backgrounds:

PhD and Work conditions

- `adbi01`: Status of the doctorate
- `adbi15`: Discipline
- `bdcd17`: Emotional Support during PhD
- `bwdr12`: Perceived scientific pressure
- `bemp81`: Gross income

Attitudes and Well-Being

- `bldc12`: Satisfaction with work-life balance
- `apar15b`: Relationship with parents
- `bpsy05`: Self-Efficacy

Demographics

- `adem01`: Gender
- `adem02`: Age in years
- `apar10`: Highest vocational degree of parents
- `adem03`: Country of birth
- `blcd01`: Relationship status

Univariate Analysis

In the previous exercise, we got to know our dataset and the main variables. In this sheet, we focus on univariate analysis looking at **one variable at a time**. Univariate analysis is the starting point of any data analysis. It helps us **understand distributions**, typical values, and strange values and is the basis for all further steps: bivariate relationships and regression models.

In the context of our research story, we are interested in **overall life satisfaction** (`blcd01`) and **having children (yes/no)** (from `blcd06`). Before we relate these outcomes to other factors, we first want to understand

- How are these variables distributed in our sample?
- How are other key variables (e.g. gender, age, burden, work–life compatibility) distributed?

In this exercise, you will:

- transform categorial variables to factors
- describe nominal variables using absolute and relative frequencies, mode, and bar charts,
- describe ordinal variables using cumulative frequencies, median, percentiles,
- describe metric variables using mean, standard deviation, percentiles, boxplots, histograms, skewness and kurtosis, standard error, confidence intervals

2.1. Set-up: Load packages and dataset

First, we (install and) load the packages `tidyverse` and `haven`, as well as `janitor`, `sjmisc`, `scales`, `skimr` and `moments`. These packages are tidyverse-friendly but not part of the core tidyverse packages, so they need to be installed and loaded separately.

1. Install `tidyverse`, `haven`, `janitor`, `sjmisc`, `scales`, `skimr` and `moments` if not already installed and load these packages.

Solution:

```
# Install tidyverse
if (!requireNamespace("tidyverse", quietly = TRUE))
  install.packages("tidyverse")

# Install haven
if (!requireNamespace("haven", quietly = TRUE))
  install.packages("haven")

# Install readxl
if (!requireNamespace("janitor", quietly = TRUE))
  install.packages("janitor")

# Install skimr
if (!requireNamespace("skimr", quietly = TRUE))
  install.packages("skimr")

# Install scales
if (!requireNamespace("scales", quietly = TRUE))
  install.packages("scales")

# Install moments
if (!requireNamespace("moments", quietly = TRUE))
  install.packages("moments")
```

```
# Install sjmisc
if (!requireNamespace("sjmisc", quietly = TRUE))
  install.packages("sjmisc")

## load packages
library(tidyverse)
library(haven)
library(sjmisc)
library(skimr)
library(janitor)
library(scales)
library(moments)
```

2. For this exercise, we will use the file *02_qa_uni_data.sav* in the *exercises* folder. Define the *exercise* folder as your working directory with the function `setwd("path")`.

3.Choose `mydata2` as a name for your data frame and import it with the following structure: `chosen_name <- read_sav("filename")`.

Solution:

```
# set working directory
setwd(⌋
↪  "C:/Users/Susanne/Nextcloud/share_DSC/003_Trainings/001_Workshops/2025/2025-12-02_SdV_Data_Analysis/Materials/E
↪  )

# Load the Nacaps data set and name this data frame "mydata2"
mydata2 <- read_sav("02_qa_uni_data.sav")
```

On the right side, in the Environment pane, the data frame `mydata2` should now appear under **Data**.

2.2. Nominal variables: Children and gender

1. Calculate absolute and relative frequencies for `blcd06` children (1 = yes/ 2 = no).

Use the function `tabyl()` from the *janitor* package to create a frequency table for `blcd06`. Look at the counts and percentages in the output, what does it tell you?

Solution:

```
# calculate absolute and relative frequencies with tabyl function
tabyl(mydata2$blcd06)
```

```
##  mydata2$blcd06    n   percent valid_percent
##               1  423 0.1796941      0.192623
##               2 1773 0.7531861      0.807377
##              NA  158 0.0671198            NA
```

`tabyl` reports the absolute frequencies (`n`), relative frequencies of all categories incl missings (`percent`, e.g. 0.75 = 75%) and relative frequencies excl. missings (`valid_percent`).

In our sample, 80% of doctoral researchers have no children (code 2), and 20% have at least one child.

2. Alternative: Calculate Absolute and relative frequencies for `adem01` gender *with labels*

`tabyl(adem01)` will show the **numeric codes**, which is not very intuitive. To make the table easier to read, we want to display the **variable and value labels** instead.

2a. To do so, first convert `adem01` into a *factor* with labels.

Use `mutate()` and `adem01 = as_factor(adem01, levels="labels")` to transform `adem01` into a factor variable. Save the result again as `mydata2`. The code should look like this:

```
# convert the labelled variable into a factor whose levels are the value labels (e.g. "female",
↪  "male", …)
mydata2 <- mydata2 %>%
  mutate(
    adem01 = as_factor(adem01, levels="labels")
  )
```

2b. Create a frequency table for `adem01` using `tabyl()`.

Compare this to the table for `blcd06` with numeric codes. Decide which version is easier to read.

Solution:

```
# calculate absolute and relative frequencies with tabyl function
tabyl(mydata2$adem01)
```

```
##  mydata2$adem01    n      percent valid_percent
##          female 1153 0.4898045879     0.4900127
##            male 1068 0.4536958369     0.4538887
##           other  132 0.0560747664     0.0560986
##            <NA>    1 0.0004248088            NA
```

3. Identify the *mode* of `alcd06`and `adem01`

The modus is the category that appears most often. As there are variables with little categories, we can easily identify it by looking at the frquency tables we created with `tabyl()`. For variables with many categories, you can identify the mode by sorting the table so that the category with the highest `n` comes first.

For gender `adem01` the categories are accidentally sorted from largest to smallest. The most frequent category is `female`.

For `alcd06`, create a `%>% pipe` where you first create a table with the `tabyl()` function for `alcd06` and then sort this table by `n` (from largest to smallest) using the `arrange(desc(n))` function. The code should look like this:

```
# identify mode
mydata2 %>%
  tabyl(blcd06) %>% # frequency table
  arrange(desc(n)) # sort in descending order
```

```
##  blcd06    n   percent valid_percent
##       2 1773 0.7531861      0.807377
##       1  423 0.1796941      0.192623
##      NA  158 0.0671198            NA
```

What is the most frequent category?

Solution:

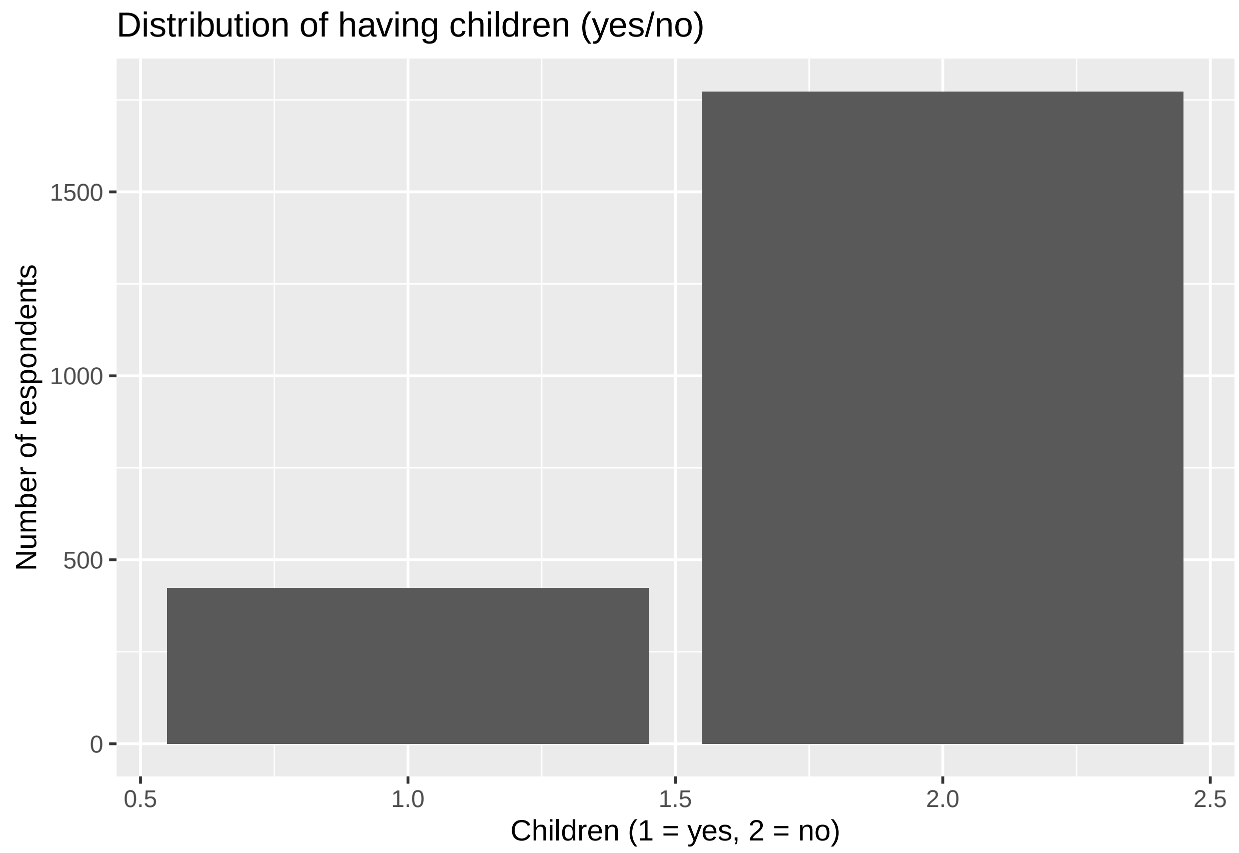
Most people in the sample do not have children.

4. Visualize the distribution of `blcd09` children (yes/no) in a bar chart.

Draw a bar chart with `blcd06` on the x-axis using `ggplot()` function. The height of the bars should show how many people are in each category (1/2). The code should look like this:

```r
mydata2 %>% # take mydata2 as input
  ggplot(aes(x = blcd06)) + # put blcd06 on x-axis
  geom_bar() + # draw bars, counting tows per category
  labs( # add labels and title
    x = "Children (1 = yes, 2 = no)", # label for x-axis
    y = "Number of respondents", # label for y-axis
    title = "Distribution of having children (yes/no)" # main title of the plot
  )
```

```
## Warning: Removed 158 rows containing non-finite outside the scale range
## (`stat_count()`).
```



What do you notice?

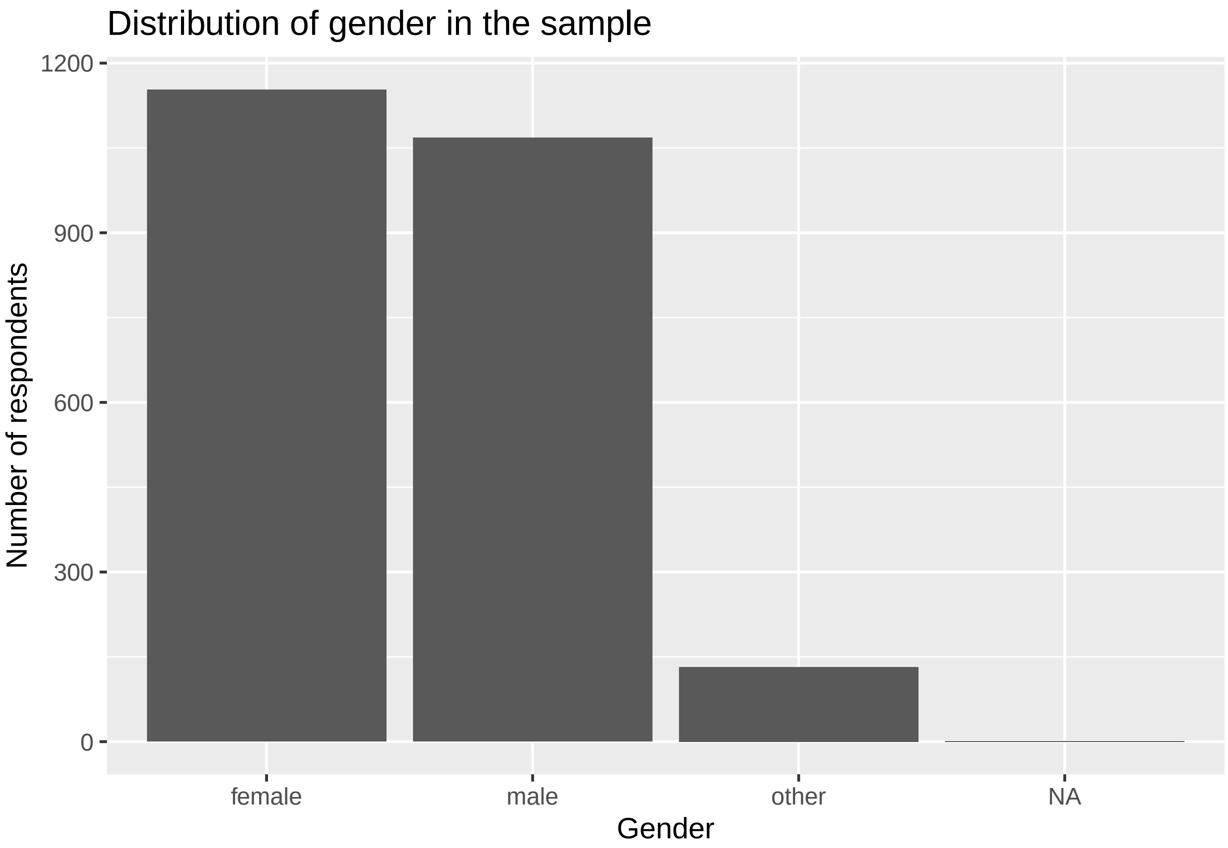
Solution:

The categories on the x-axis don't make any sense because R does not know that it should treat the variable like a categorical variable.

5. Create a similar bar chart for the distribution of gender `adem01`. What do you notice?

Solution:

```
mydata2 %>%
  ggplot(aes(x = adem01)) +
  geom_bar() +
  labs(
    x = "Gender",
    y = "Number of respondents",
    title = "Distribution of gender in the sample"
  )
```

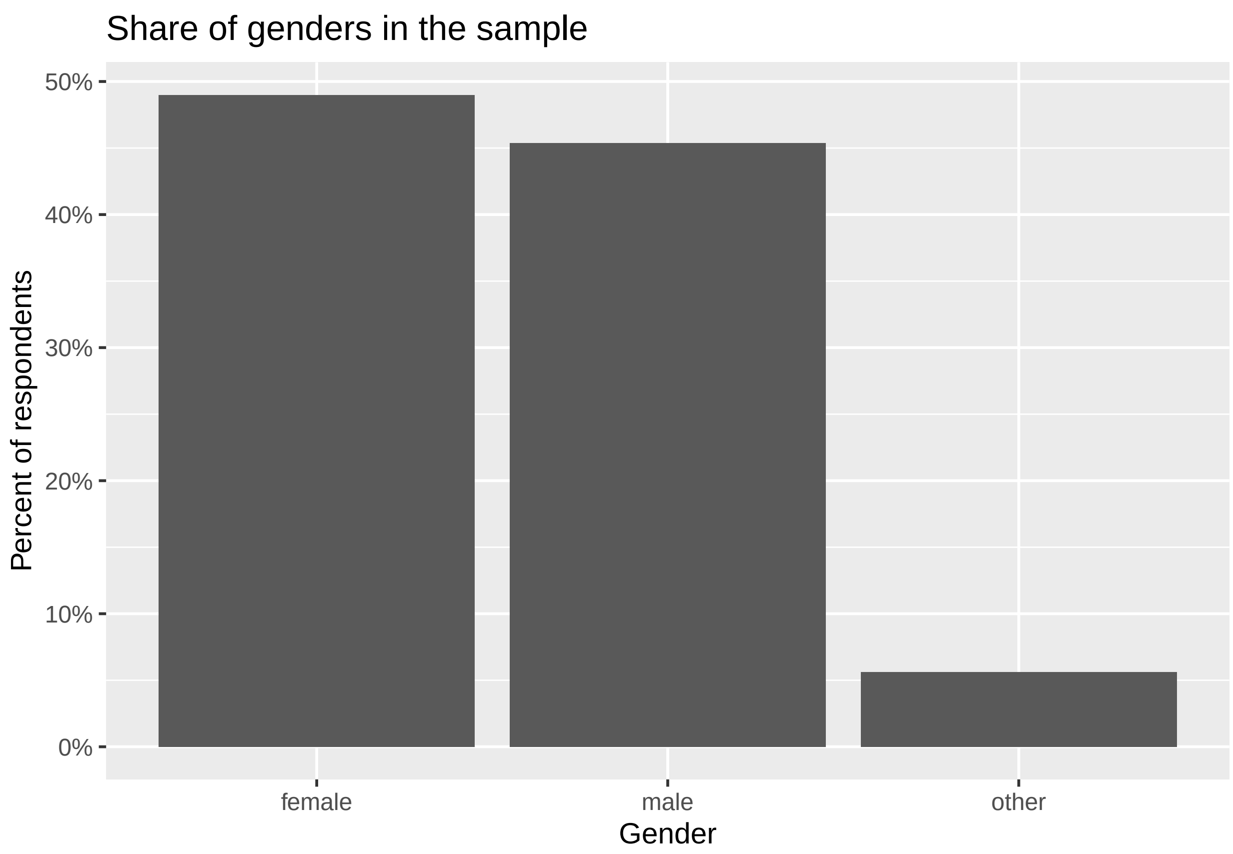


One the variable is transformed into a factor, the categories on the axes make much more sense now.

6. Modify the graph so 1.) missings do not appear and 2.) percentages are plotted.

To do so, you need to remove missing values out of `adem01`, count the cases in the category and use these information for plotting the bar chart. Modify the `ggplot()` function like this:

```
mydata2 %>%
  filter(!is.na(adem01)) %>%             # remove missing values of adem01
  count(adem01) %>%                      # count how many cases in each category
  mutate(prop = n / sum(n)) %>%          # compute proportion (relative frequency)
  ggplot(aes(x = adem01, y = prop)) +    # map category to x-axis, proportion to y-axis
  geom_col() +                           # draw bars with given heights (prop)
  scale_y_continuous(labels = percent) + # scales package: show y-axis as percentages
  labs(
    x = "Gender",                        # x-axis label
    y = "Percent of respondents",        # y-axis label
    title = "Share of genders in the sample"
  )
```



2.3. Ordinal variable: Highest vocational degree of parents

Variable `apar10` measures the **highest vocational degree of the respondents' parents**. It is a categoricial variable with three categories.

1. Use `mutate()` and `apar10 = as_factor(adem01, levels="labels")` to transform `apar10`into a factor variable. Save the result again as `mydata2`.

Solution:

```
# convert the labelled variable into a factor whose levels are the value labels
mydata2 <- mydata2 %>%
  mutate(
    apar10 = as_factor(apar10, levels="labels")
  )
```

2. Use the `frq()` function from the *sjmisc* package to calculate a table with *absolute, relative and cumulative frequencies.* Check also the *percentiles.*

Solution:

```
# frequency table with frq function
frq(mydata2$apar10)
```

```
## Parents' highest level of vocational training (4 categories, highest of mother/father, aggregated) (
## # total N=2354 valid N=2346 mean=2.26 sd=0.75
```

```
##
## Value                          |    N | Raw % | Valid % | Cum. %
## ---------------------------------------------------------------
## PhD/doctorate                  |  375 | 15.93 |   15.98 |  15.98
## Bachelor / Master degree       | 1033 | 43.88 |   44.03 |  60.02
## Other vocational qualification |  882 | 37.47 |   37.60 |  97.61
## Unclassifiable                 |   56 |  2.38 |    2.39 | 100.00
## <NA>                           |    8 |  0.34 |    <NA> |   <NA>
```

We find that 16% of the participants come from a household where at least one parent also got a PhD. Around 44% of the participants have a background where at least one parent have another higher education degree. The cumulative percentage tell us that 60 percent of the participants have parents with an academic background.

You can use these **cumulative percentages** to read off percentiles for an ordinal variable.

In the `frq()` output, look for the column with cumulative (valid) percent. Then:

- The **25th percentile (P25)** is the first category where the cumulative percent is greater than or equal to 25%. In this case, it's in the 'Bachelor/Master degree category.
- The **50th percentile (P50, median)** is the first category where the cumulative percent is 50%.
- The **75th percentile (P75)** is the first category where the cumulative percent is 75%.

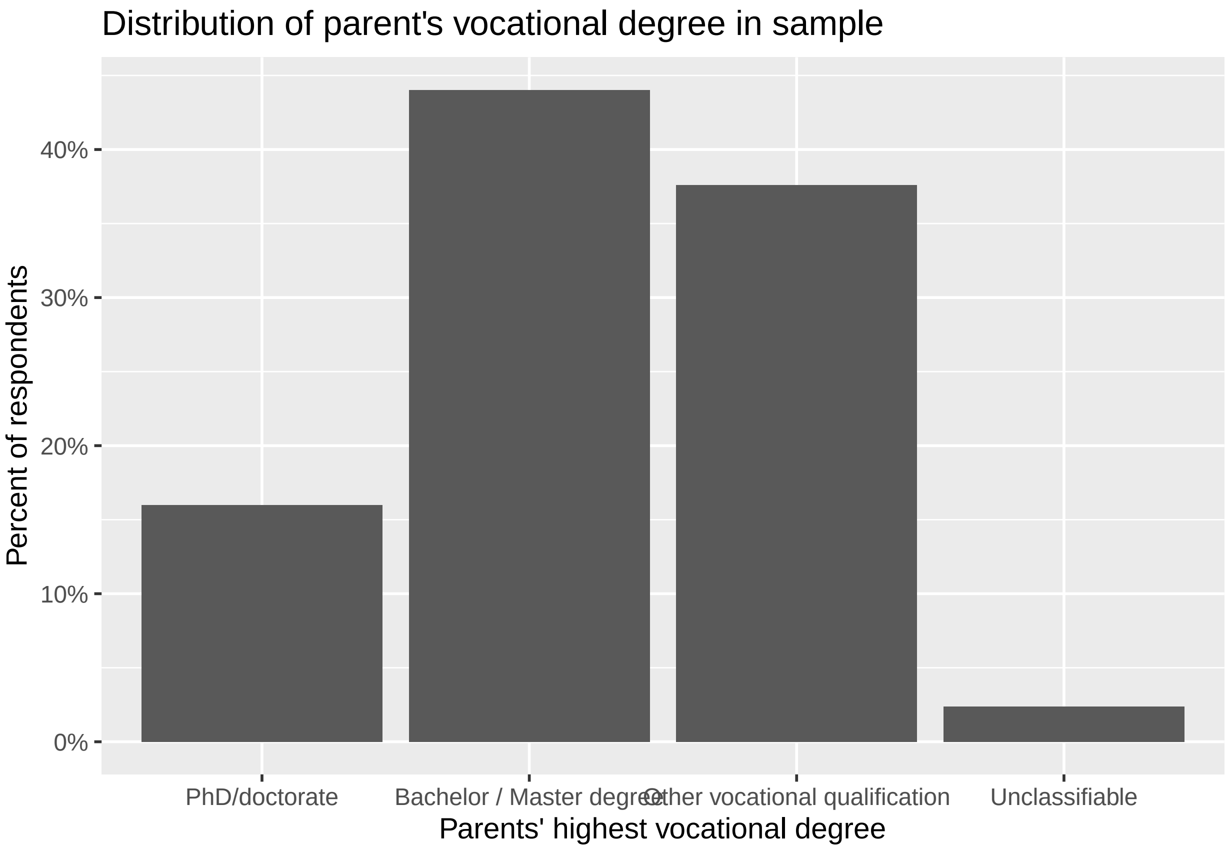
In other words: move down the cumulative percent column until you reach or pass the percentile you are interested in. The category at that point is the percentile category.

Note: For ordinal variables, percentiles are categories, not exact numeric values in between categories. As you can see, `frq()` also calculates a mean and standard deviation. However, these values do not really make sense for categorical variables.

3. Create a bar chart showing the *distribution* of `apar10`, without missings and percentages plotted on the y-axis.

Solution:

```r
mydata2 %>%
  filter(!is.na(apar10)) %>%              # remove missing values of apar10
  count(apar10) %>%                       # count how many cases in each category
  mutate(prop = n / sum(n)) %>%           # compute proportion (relative frequency)
  ggplot(aes(x = apar10, y = prop)) +     # map category to x-axis, proportion to y-axis
  geom_col() +                            # draw bars with given heights (prop)
  scale_y_continuous(labels = percent) +  # show y-axis as percentages
  labs(
    x = "Parents' highest vocational degree",  # x-axis label
    y = "Percent of respondents",         # y-axis label
    title = "Distribution of parent's vocational degree in sample"
  )
```



2.4. Metric variable: Overall life-satisfaction

Our outcome variable `bpsy01` measures the **overall life satisfaction** rated on a scale from 0 to 10. Technically, it is **an ordinal variable**, but in research practices, scales like this are often **treated as metric**. And this is what we are going to do.

1. Look at the distribution of `bpsy01`. Calculate the *absolute, relative and cumulative frequencies* and look at the *mean and standard distribution* using the `frq()` function.

Solution:

```
# frequency table for bpsy01
frq(mydata2$bpsy01)
```

```
## Overall life satisfaction (x) <numeric>
## # total N=2354 valid N=2197 mean=7.04 sd=1.90
##
## Value |                   Label |   N | Raw % | Valid % | Cum. %
## ----------------------------------------------------------------
##     0 | 0 not at all satisfied |  10 |  0.42 |    0.46 |   0.46
##     1 |                       1 |  13 |  0.55 |    0.59 |   1.05
##     2 |                       2 |  43 |  1.83 |    1.96 |   3.00
##     3 |                       3 |  92 |  3.91 |    4.19 |   7.19
##     4 |                       4 |  87 |  3.70 |    3.96 |  11.15
##     5 |                       5 | 137 |  5.82 |    6.24 |  17.39
##     6 |                       6 | 243 | 10.32 |   11.06 |  28.45
##      7 |                      7 | 531 | 22.56 |   24.17 |  52.62
##      8 |                      8 | 596 | 25.32 |   27.13 |  79.75
##      9 |                      9 | 342 | 14.53 |   15.57 |  95.31
##     10 |      10 fully satisfied | 103 |  4.38 |    4.69 | 100.00
##   <NA> |                   <NA> | 157 |  6.67 |    <NA> |   <NA>
```

2. Examine `bpsy01` using the summary and the `skim()` function from the *skimr* package. What can you see here? How do they differ?

Solution:

```
# distribution using the summary function
summary(mydata2$bpsy01)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
##   0.000   6.000   7.000   7.036   8.000  10.000     157
```

```
# distribution using the skim function
skim(mydata2$bpsy01)
```

Table 1: Data summary

Name	mydata2$bpsy01
Number of rows	2354
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	157	0.93	7.04	1.9	0	6	7	8	10	

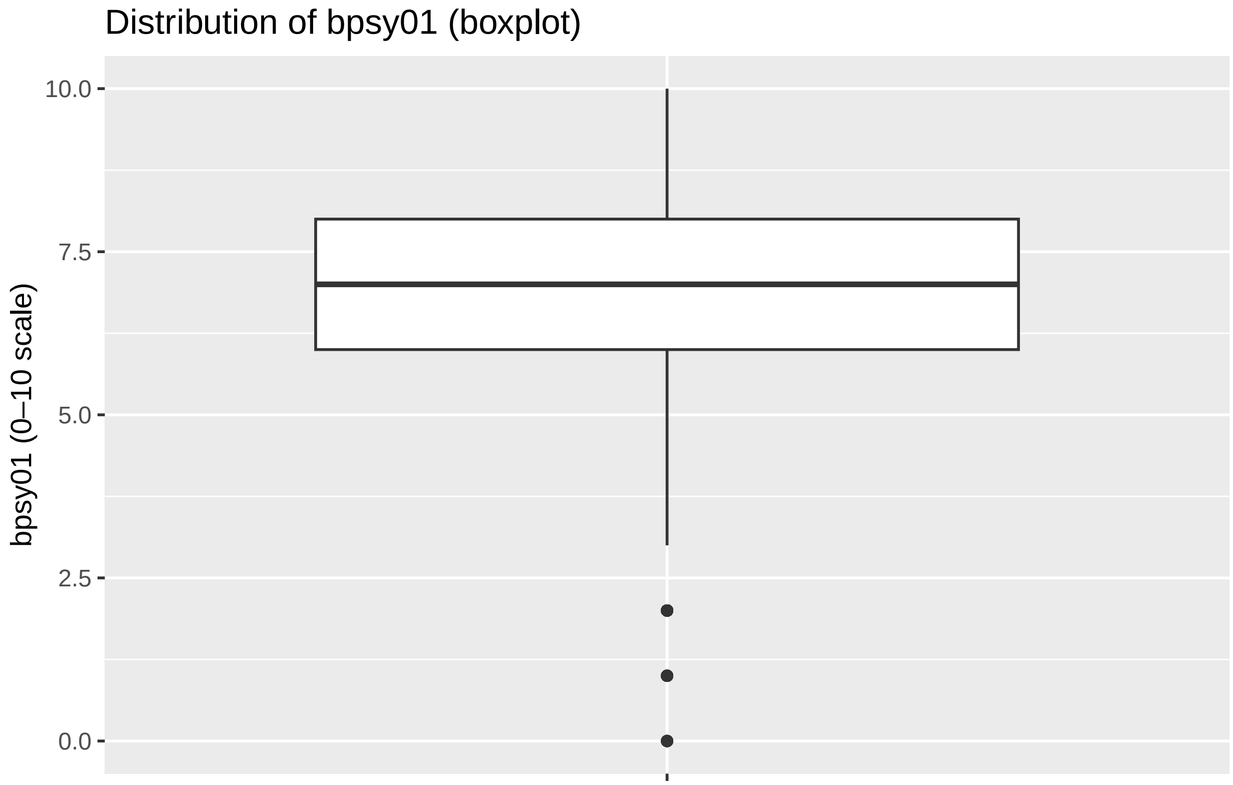
`summary()` is a base R function that gives a quick overview of each variable in a data frame (min, quartiles, median, mean, max for numeric variables; most frequent categories for factors; plus NAs).

`skim()` provides a more detailed, tidy overview of the data. It **detects variable types** and reports **type-specific statistics** (e.g. mean, SD, percentiles and a mini histogram for numeric variables; number of levels and top categories for factors).

3. To visualize the distribution of `bpsy01`, draw a boxplot with `ggplot`.

The code goes like this:

```
mydata2 %>%
  filter(!is.na(bpsy01)) %>%                 # remove missings
  ggplot(aes(x = "", y = bpsy01)) +          # bpsy01 on y-axis
  geom_boxplot() +                           # draw boxplot
  labs(
    x = "",
    y = "bpsy01 (0-10 scale)",
    title = "Distribution of bpsy01 (boxplot)"
  )
```



What does the boxplot show you?

Solution:

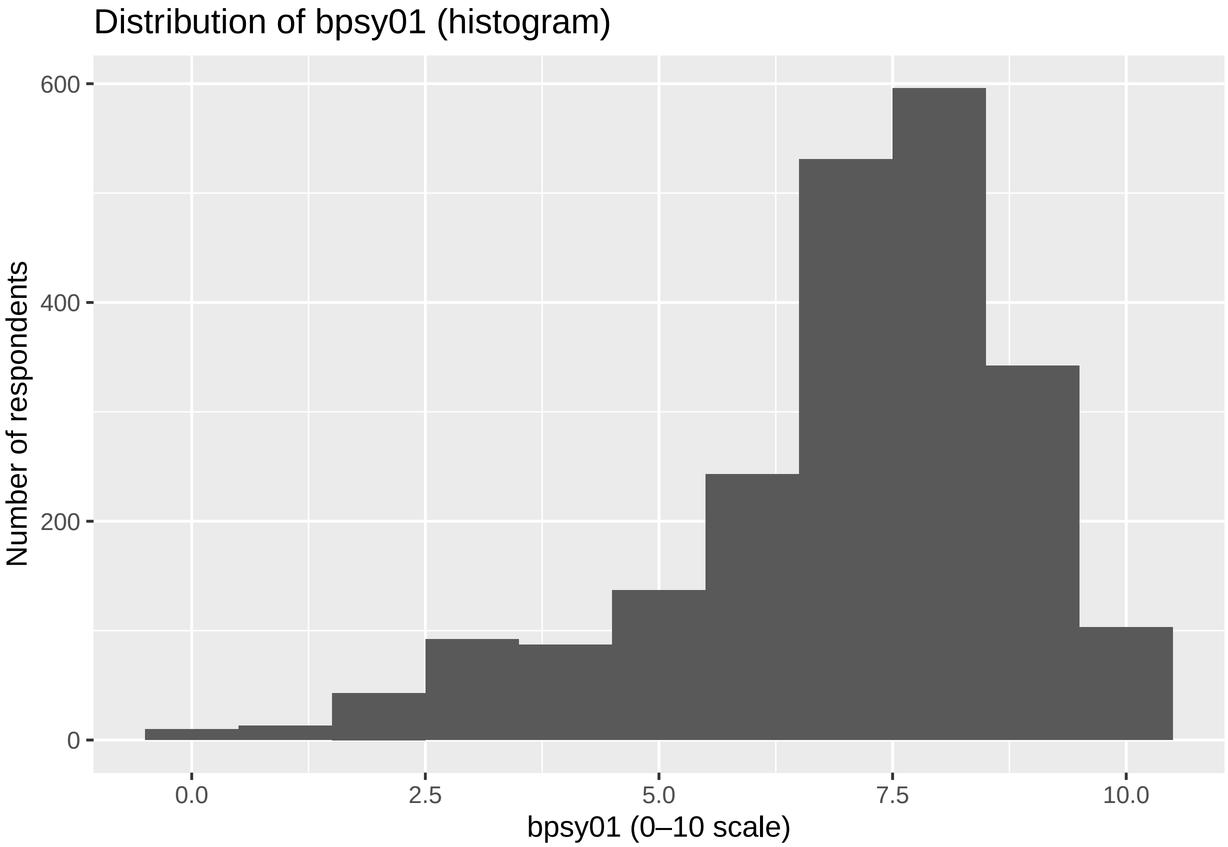
In a boxplot, the box shows the middle 50% of the values (between the 25th and 75th percentile), the horizontal line inside the box is the median, the whiskers show the typical range of the data, and the dots are outliers.

Here we see that most respondents have high values on `bpsy01`: the middle 50% lie roughly between 6 and 8, with a median around 7. A few respondents report much lower values (below 3), which appear as outliers at the bottom of the plot.

4. To see the overall shape of the distribution of `bpsy01`, create a *histogram* using `ggplot()`. Use `bpsy01` on the x-axis and choose a bin width of 1 to show the 0–10 scale clearly.

The code goes like this:

```
# histogram for bpsy01
mydata2 %>%
  filter(!is.na(bpsy01)) %>%                # remove missings
  ggplot(aes(x = bpsy01)) +                 # bpsy01 on x-axis
  geom_histogram(binwidth = 1) +            # bins of width 1 (0,1,2,...,10)
  labs(
    x = "bpsy01 (0-10 scale)",
    y = "Number of respondents",
    title = "Distribution of bpsy01 (histogram)"
  )
```



What do you see regarding the skewness of `bpsy01`?

Solution:

The histogram shows that bpsy01 is clearly concentrated at the higher end of the 0–10 scale: most respondents have values between about 6 and 9, with a peak around 7–8. Very low values (below 3) are rare, so the distribution is slightly **left-skewed (longer tail towards the lower scores)**.

Note: If you want to compute skewness and kurtosis, you can do it using the following functions from the *moments* package:

```
skewness(mydata2$bpsy01, na.rm = TRUE)
```

```
## [1] -1.051064
```

```
kurtosis(mydata2$bpsy01, na.rm = TRUE)
```

```
## [1] 4.064778
```

5. Let's look at inference statistics for `bpsy01`. Pick the functions from the table to calculate *standard errors* and both *95 % confidence intervals*.

Overview: Descriptive statistics, e.g. for variable `bpsy01`

What we compute	R command (inside `summarise()`)	Explanation
Number of valid cases (n)	`n = sum(!is.na(bpsy01))`	Counts all non-missing values of `bpsy01`.
Mean	`mean_val = mean(bpsy01, na.rm = TRUE)`	Average value of `bpsy01`.
Standard deviation (SD)	`sd_val = sd(bpsy01, na.rm = TRUE)`	How much values vary around the mean.
Variance	`var_val = var(bpsy01, na.rm = TRUE)`	SD squared.
Standard error of the mean (SE)	`se_val = sd_val / sqrt(n)`	Uncertainty of the sample mean.
95% CI – lower bound	`ci_lower = mean_val - 1.96 * se_val`	Lower limit of the 95% confidence interval.
95% CI – upper bound	`ci_upper = mean_val + 1.96 * se_val`	Upper limit of the 95% confidence interval.

Solution:

```r
mydata2 %>%
  summarise(
    n        = sum(!is.na(bpsy01)),             # number of valid cases
    mean_val = mean(bpsy01, na.rm = TRUE),      # mean
    sd_val   = sd(bpsy01, na.rm = TRUE),        # standard deviation
    se_val   = sd_val / sqrt(n),                # standard error of the mean
    ci_lower = mean_val - 1.96 * se_val,        # 95% CI lower bound
    ci_upper = mean_val + 1.96 * se_val         # 95% CI upper bound
  )
```

```
## # A tibble: 1 x 6
##       n mean_val sd_val se_val ci_lower ci_upper
##   <int>    <dbl>  <dbl>  <dbl>    <dbl>    <dbl>
## 1  2197     7.04   1.90 0.0406     6.96     7.12
```

Take Home checklist: Univariate analysis in R

Step	Question / task	Useful functions / tools
1	Do your categorical survey variables have readable categories (factors)?	Convert labelled variables with `as_factor()`, then use `tabyl()` or `ggplot()` on the factors.
2	Are missing values coded correctly?	Recode special codes to `NA` (e.g. `mutate(across(..., ~ ifelse(.x < 0, NA, .x)))`)
3	For nominal variables: how often does each category occur?	`tabyl()`, `count()`, bar charts with `ggplot(aes(x = var)) + geom_bar()` / `geom_col()`
5	For ordinal variables: how are responses distributed along the scale?	`frq()`, cumulative % and percentiles from the table
6	For metric variables: what are central tendency and spread and shape?	`mean()`, `sd()`, `var()`, `quantile()` `skimr()`, SE and CI via `summarise()`, box plot with `geom_boxplot()`, histogram with `geom_histogram()`
7	Are there obvious problems in the distributions?	Check tables, boxplots and histograms for impossible values, extreme outliers, very strong skewness etc.