Intro and descriptive statistics

The ASTA team

Contents

1 Software

1.1 Rstudio

- Make a folder on your computer where you want to keep files to use in **Rstudio**. **Do NOT use Danish characters æ, ø, å** in the folder name (or anywhere in the path to the folder).
- Set the working directory to this folder: `Session -> Set Working Directory -> Choose Directory` (shortcut: Ctrl+Shift+H).
- Make the change permanent by setting the default directory in: `Tools -> Global Options -> Choose Directory`.

1.2 R basics

- Ordinary calculations:

```
4.6 * (2 + 3)^4
```

```
## [1] 2875
```

- Make a (scalar) object and print it:

```
a <- 4
a
```

```
## [1] 4
```

- Make a (vector) object and print it:

```
b <- c(2, 5, 7)
b
```

```
## [1] 2 5 7
```

- Make a sequence of numbers and print it:

```
s <- 1:4
s
```

```
## [1] 1 2 3 4
```

- Note: A more flexible command for sequences:

```
s <- seq(1, 4, by = 1)
```

- **R** does elementwise calculations:

```
a * b
```

```
## [1]  8 20 28
```

```
a + b
```

```
## [1]  6  9 11
```

```
b ^ 2
```

```
## [1]  4 25 49
```

- Sum and product of elements:

```
sum(b)
```

```
## [1] 14
```

```
prod(b)
```

```
## [1] 70
```

1.3   R extensions

- The functionality of **R** can be extended through libraries or packages (much like plugins in browsers etc.). Some are installed by default in **R** and you just need to load them.
- To install a new package in **Rstudio** use the menu: `Tools -> Install Packages`
- You need to know the name of the package you want to install. You can also do it through a command:

```
install.packages("mosaic")
```

- When it is installed you can load it through the `library` command:

```
library(mosaic)
```

- This loads the `mosaic` package which has a lot of convenient functions for this course (we will get back to that later). It also prints a lot of info about functions that have been changed by the `mosaic` package, but you can safely ignore that.

1.4   R help

- You get help via `?<command>`:

```
?sum
```

- Use `tab` to make **Rstudio** guess what you have started typing.
- Search for help:

```
help.search("plot")
```

- You can find a cheat sheet with the **R** functions we use for this course here.
- Save your commands in a file for later usage:
    - Select history tab in top right pane in **Rstudio** .
    - Mark the commands you want to save.
    - Press `To Source` button.

2   Data

2.1   Data example

Data: Magazine Ads Readability

- Thirty magazines were ranked by educational level of their readers.
- Three magazines were **randomly** selected from each of the following groups:
    - Group 1: highest educational level
    - Group 2: medium educational level
    - Group 3: lowest educational level.
- Six advertisements were **randomly** selected from each of the following nine selected magazines:
    - Group 1: [1] Scientific American, [2] Fortune, [3] The New Yorker
    - Group 2: [4] Sports Illustrated, [5] Newsweek, [6] People
    - Group 3: [7] National Enquirer, [8] Grit, [9] True Confessions
- So, the data contains information about a total of 54 advertisements.

2.2   Data example (continued) - variables and format

- For each advertisement (54 cases), the data below were observed.
- **Variable names**:
    - WDS = number of words in advertisement

- – SEN = number of sentences in advertisement
- – 3SYL = number of 3+ syllable words in advertisement
- – MAG = magazine (1 through 9 as above)
- – GROUP = educational level (1 through 3 as above)
- Take a look at the data from within **Rstudio**:

```
magAds <- read.delim("https://asta.math.aau.dk/datasets?file=magazineAds.txt")
head(magAds)
```

```
##   WDS SEN X3SYL MAG GROUP
## 1 205   9    34   1     1
## 2 203  20    21   1     1
## 3 229  18    37   1     1
## 4 208  16    31   1     1
## 5 146   9    10   1     1
## 6 230  16    24   1     1
```

- Variable names are in the top row. They are not allowed to start with a digit, so an `X` has been prefixed in `X3SYL`.

2.3   Data types

2.3.1   Quantitative variables

- The measurements have numerical values.
- Quantative data often comes about in one of the following ways:
  - **Continuous variables**: measurements of e.g. waiting times in a queue, revenue, share prices, etc.
  - **Discrete variables**: counts of e.g. words in a text, hits on a webpage, number of arrivals to a queue in one hour, etc.
- Measurements like this have a well-defined scale and in **R** they are stored as the type **numeric**.
- It is important to be able to distinguish between discrete count variables and continuous variables, since this often determines how we describe the uncertainty of a measurement.

2.3.2   Categorical/qualitative variables

- The measurement is one of a set of given categories, e.g. sex (male/female), social status, satisfaction score (low/medium/high), etc.
- The measurement is usually stored (which is also recommended) as a **factor** in **R**. The possible categories are called **levels**. Example: the levels of the factor "sex" is male/female.
- Factors have two so-called scales:
  - **Nominal scale**: There is no natural ordering of the factor levels, e.g. sex and hair color.
  - **Ordinal scale**: There is a natural ordering of the factor levels, e.g. social status and satisfaction score. A factor in **R** can have a so-called **attribute** assigned, which tells if it is ordinal.

3   Population and sample

3.1   Aim of statistics

- Statistics is all about "saying something" about a population.
- Typically, this is done by taking a random sample from the population.
- The sample is then analysed and a statement about the population can be made.
- The process of making conclusions about a population from analysing a sample is called **statistical inference**.

3.2 Selecting randomly

- For the magazine data:
  - First we select **randomly** 3 magazines from each group.
  - Then we select **randomly** 6 ads from each magazine.
  - An important detail is that the selection is done completely at **random**, i.e.
    * each magazine within a group have an equal chance of being chosen and
    * each ad within a magazine have an equal chance of being chosen.
- In the following it is a fundamental requirement that the data collection respects this principle of randomness and in this case we use the term **sample**.
- More generally:
  - We have a **population** of objects.
  - We choose completely at random n of these objects, and from the jth object we get the measurement y_j, $j = 1, 2, \ldots, n$.
  - The measurements $y_1, y_2, \ldots, y_n$ are then called a **sample**.
- If we e.g. are measuring the water quality 4 times in a year then it is a bad idea to only collect data in fair weather. The chosen sampling time is not allowed to be influenced by something that might influence the measurement itself.

4 Variable grouping and frequency tables

4.1 Binning

- The function `cut` will divide the range of a numeric variable in a number of equally sized intervals, and record which interval each observation belongs to. E.g. for the variable `X3SYL` (the number of words with more than three syllables) in the magazine data:

```
# Before 'cutting':
magAds$X3SYL[1:5]
```

```
## [1] 34 21 37 31 10
```

```
# After 'cutting' into 4 intervals:
syll <- cut(magAds$X3SYL, 4)
syll[1:5]
```

```
## [1] (32.2,43]    (10.8,21.5]  (32.2,43]    (21.5,32.2]  (-0.043,10.8]
## Levels: (-0.043,10.8] (10.8,21.5] (21.5,32.2] (32.2,43]
```

- The result is a `factor` and the labels are the interval end points by default. Custom ones can be assigned through the `labels` argument:

```
labs <- c("few", "some", "many", "lots")
syll <- cut(magAds$X3SYL, 4, labels = labs) # NB: this overwrites the 'syll' defined above
syll[1:5]
```

```
## [1] lots some lots many few
## Levels: few some many lots
```

```
magAds$syll <- syll # Adding a new column to the dataset
```

4.2 Tables

- To summarize the results we can use the function `tally` from the `mosaic` package (remember the package **must be loaded** via `library(mosaic)` if you did not do so yet):

```
tally( ~ syll, data = magAds)
```

```
## syll
##  few some many lots
##   26   14   10    4
```

- In percent:

```
tally( ~ syll, data = magAds, format = "percent")
```

```
## syll
##  few some many lots
## 48.1 25.9 18.5  7.4
```

- Here we use an **R formula** (characterized by the "tilde" sign ~) to indicate that we want this variable from the dataset `magAds` (without the tilde it would look for a global variable called `syll` and use that rather than the one in the dataset).

4.3 2 factors: Cross tabulation

- To make a table of all combinations of two factors we use `tally` again:

```
tally( ~ syll + GROUP, data = magAds)
```

```
##        GROUP
## syll    1  2  3
##   few   8 11  7
##   some  4  2  8
##   many  3  5  2
##   lots  3  0  1
```

- Relative frequencies (in percent) columnwise:

```
tally( ~ syll | GROUP, data = magAds, format = "percent")
```

```
##        GROUP
## syll      1    2    3
##   few  44.4 61.1 38.9
##   some 22.2 11.1 44.4
##   many 16.7 27.8 11.1
##   lots 16.7  0.0  5.6
```

- So, the above table shows e.g. how many percentage of the advertisements in group 1 that have 'few', 'some', 'many' or 'lots' words with more than 3 syllables.
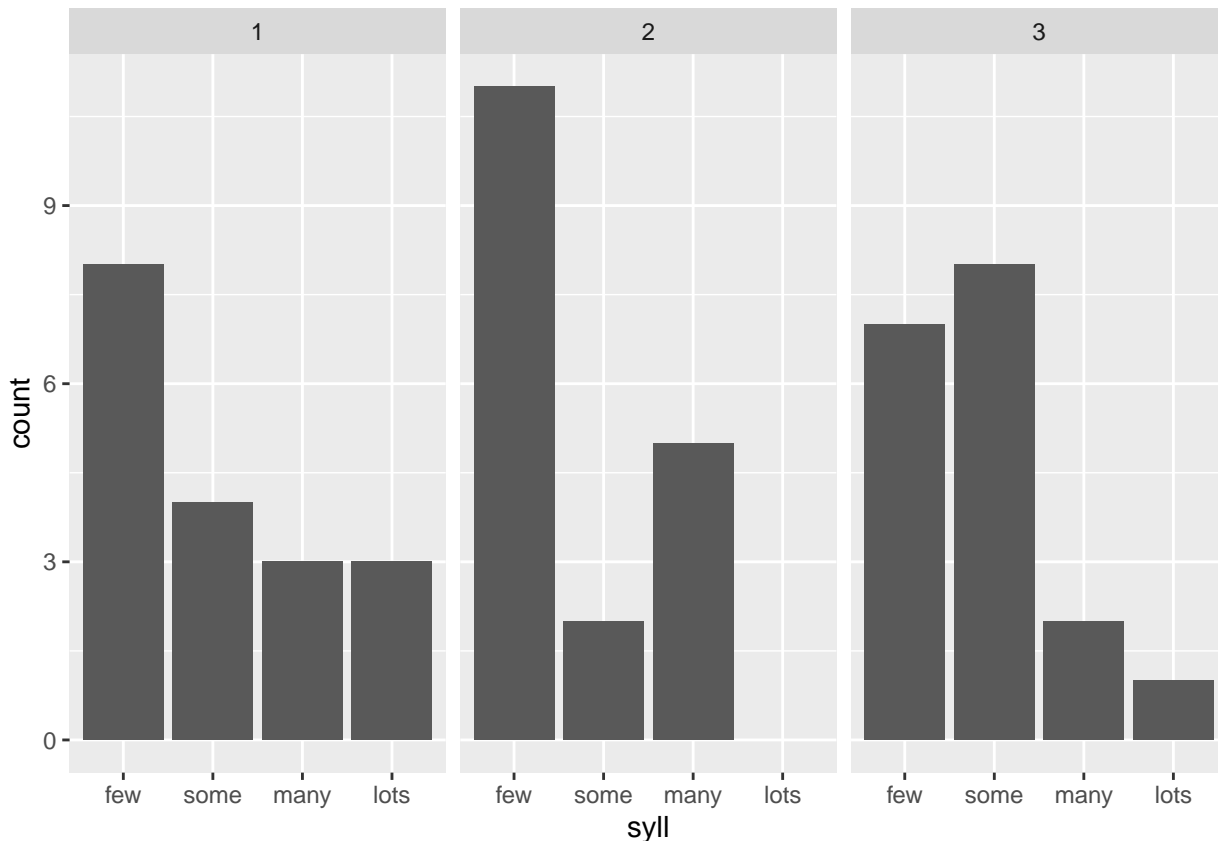
5 Graphics

5.1 Bar graph

- To create a bar graph plot of table data we use the function `gf_bar` from `mosaic`. For each level of the factor a box is drawn with the height proportional to the frequency (count) of the level.

```
gf_bar( ~ syll, data = magAds)
```
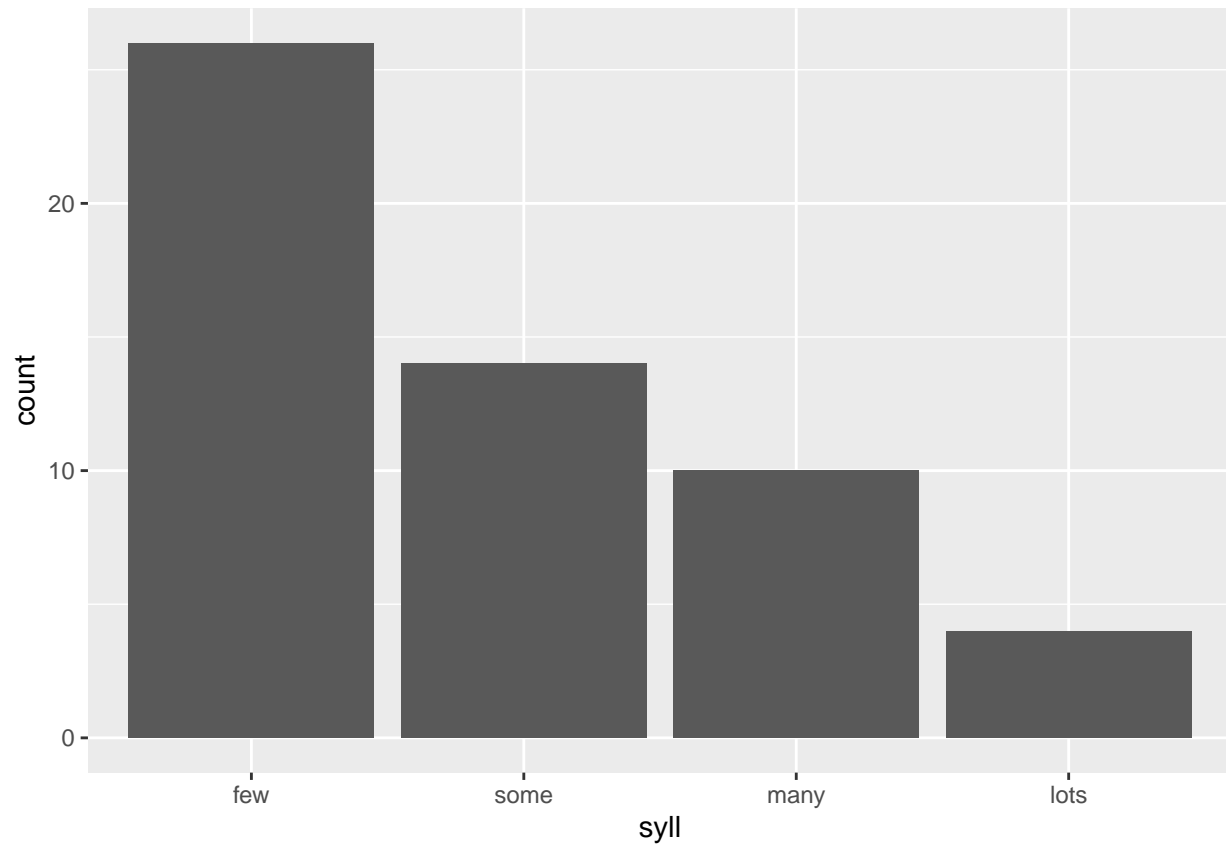
- The bar graph can also be split by group:

```
gf_bar( ~ syll | GROUP, data = magAds)
```

5.2 The Ericksen data

- Description of data: Ericksen 1980 U.S. Census Undercount.
- This data contains the following variables:
  - `minority`: Percentage black or Hispanic.
  - `crime`: Rate of serious crimes per 1000 individuals in the population.
  - `poverty`: Percentage poor.
  - `language`: Percentage having difficulty speaking or writing English.
  - `highschool`: Percentage aged 25 or older who had not finished highschool.
  - `housing`: Percentage of housing in small, multiunit buildings.
  - `city`: A factor with levels: `city` (major city) and `state` (state or state-remainder).
  - `conventional`: Percentage of households counted by conventional personal enumeration.
  - `undercount`: Preliminary estimate of percentage undercount.
- The Ericksen data has 66 rows/observations and 9 columns/variables.
- The observations are measured in 16 large cities, the remaining parts of the states in which these cities are located, and the other U.S. states.

```
Ericksen <- read.delim("https://asta.math.aau.dk/datasets?file=Ericksen.txt")
head(Ericksen)
```

```
##          name minority crime poverty language highschool housing  city
## 1      Alabama     26.1    49      19      0.2         44     7.6 state
## 2       Alaska      5.7    62      11      1.7         18    23.6 state
## 3      Arizona     18.9    81      13      3.2         28     8.1 state
## 4     Arkansas     16.9    38      19      0.2         44     7.0 state
## 5 California.R     24.3    73      10      5.0         26    11.8 state
## 6     Colorado     15.2    73      10      1.2         21     9.2 state
```

8

```
##   conventional undercount
## 1            0     -0.04
## 2          100      3.35
## 3           18      2.48
## 4            0     -0.74
## 5            4      3.60
## 6           19      1.34
```

- Want to make a histogram for crime rate - how?

5.3 Histogram (quantitative variables)

- How to make a histogram for some variable `x`:
    - Divide the interval from the minimum value of `x` to the maximum value of `x` in an appropriate number of equal sized sub-intervals.
    - Draw a box over each sub-interval with the height being proportional to the number of observations in the sub-interval.
- Histogram of crime rates for the Ericksen data

```
gf_histogram( ~ crime, data = Ericksen)
```
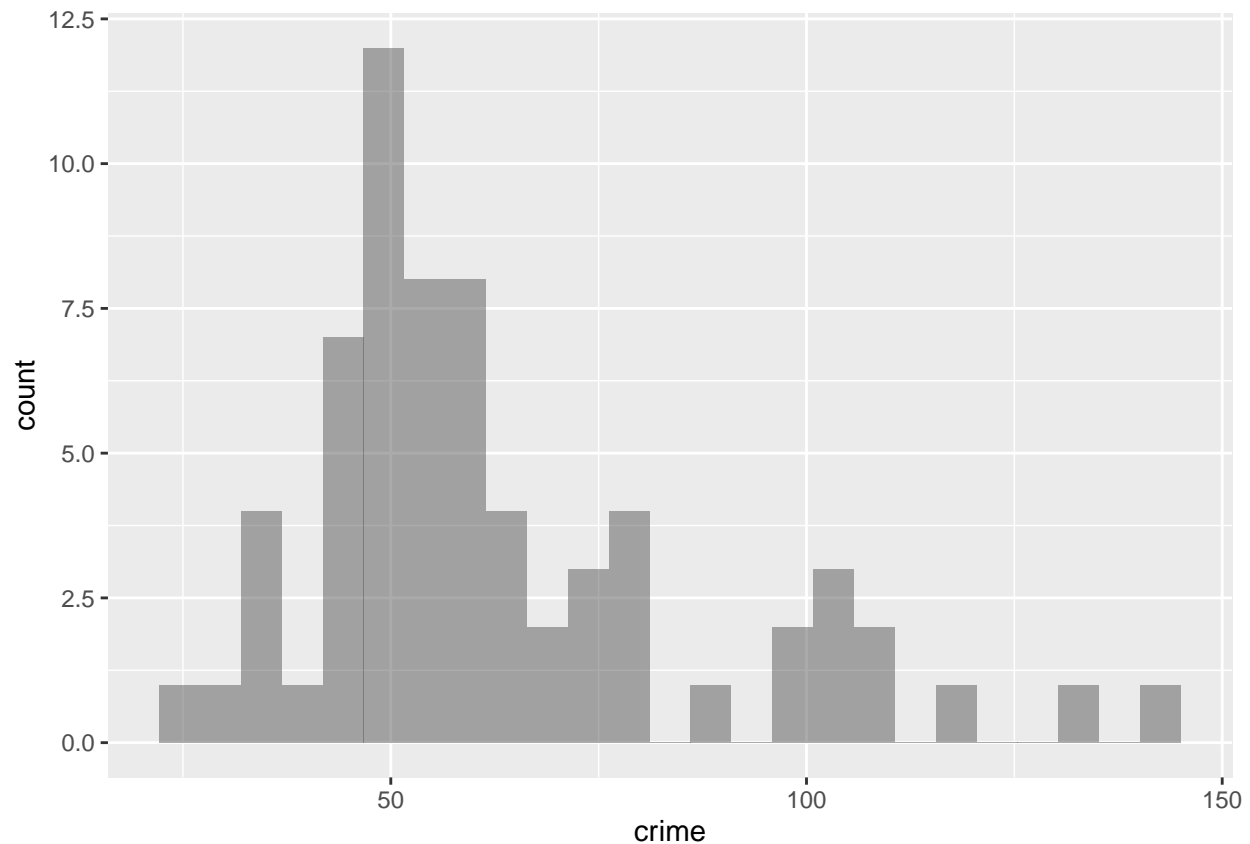

6 Summary of quantitative variables

6.1 Measures of center of data: Mean and median

- We return to the magazine ads example (`WDS` = number of words in advertisement). A number of numerical summaries for `WDS` can be retrieved using the `favstats` function:

```r
favstats( ~ WDS, data = magAds)
```

```
##  min Q1 median  Q3 max mean sd  n missing
##   31 69     96 202 230  123 66 54       0
```

- The observed values of the variable `WDS` are $y_1 = 205$, $y_2 = 203, \ldots, y_n = 208$, where there are a total of $n = 54$ values. As previously defined this constitutes a **sample**.
- **mean** $= 123$ is the **average** of the sample, which is calculated by

$$\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$$

We may also call $\bar{y}$ the **(empirical) mean** or the **sample mean**.
- **median** $= 96$ is the 50-percentile, i.e. the value that splits the sample in 2 groups of equal size.
- An important property of the **mean** and the **median** is that they have the same unit as the observations (e.g. meter).

6.2 Measures of variability of data: range, standard deviation and variance

- The **range** is the difference of the largest and smallest observation.
- The **(empirical) variance** is the average of the squared deviations from the mean:

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (y_i - \bar{y})^2.$$

- **sd** $=$ **standard deviation** $= s = \sqrt{s^2}$.
- Note: If the observations are measured in meter, the **variance** has unit meter^2 which is hard to interpret. The **standard deviation** on the other hand has the same unit as the observations.
- The standard deviation describes how much data varies around the (empirical) mean.

6.3 Calculation of mean, median and standard deviation using R

The mean, median and standard deviation are just some of the summaries that can be read of the `favstats` output (shown on previous page). They may also be calculated separately in the following way:

- Mean of `WDS`:

```r
mean( ~ WDS, data = magAds)
```

```
## [1] 123
```

- Median of `WDS`:

```r
median( ~ WDS, data = magAds)
```

```
## [1] 96
```

- Standard deviation for `WDS`:

```r
sd( ~ WDS, data = magAds)
```

```
## [1] 66
```

We may also calculate the summaries for each group (variable `GROUP`), e.g. for the mean:
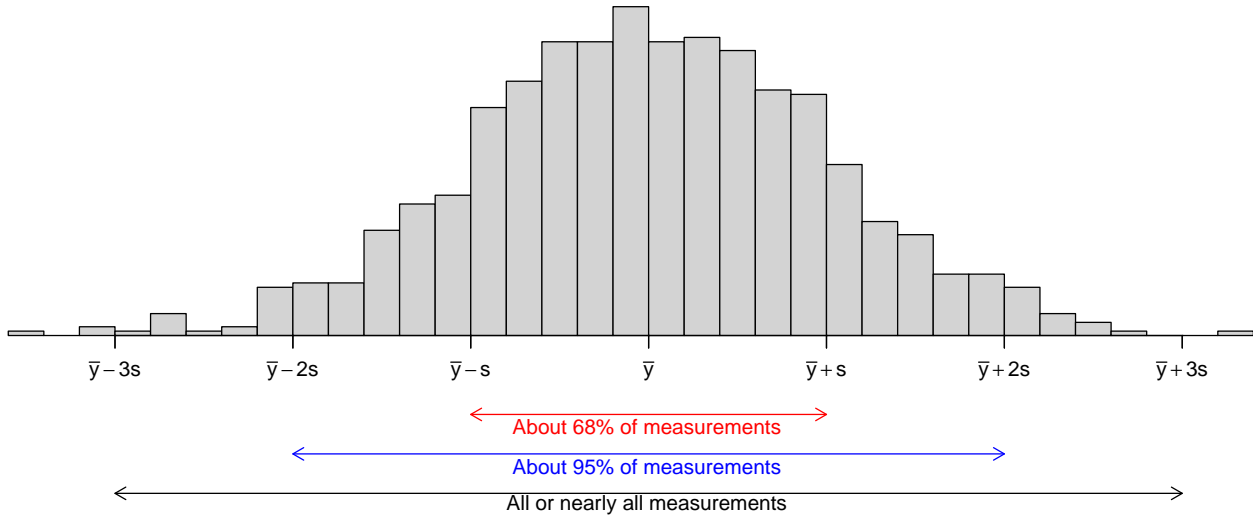
```r
mean( ~ WDS | GROUP, data = magAds)
```

```
##   1   2   3
## 140 121 106
```

6.4 A word about terminology

- **Standard deviation**: a measure of variability of a population or a sample.
- **Standard error**: a measure of variability of an estimate. For example, a measure of variability of the sample mean.

6.5 The empirical rule

If the histogram of the sample looks like a bell shaped curve, then

- about 68% of the observations lie between $\bar{y} - s$ and $\bar{y} + s$.
- about 95% of the observations lie between $\bar{y} - 2s$ and $\bar{y} + 2s$.
- All or almost all (99.7%) of the observations lie between $\bar{y} - 3s$ and $\bar{y} + 3s$.

6.6 Percentiles

- **The pth percentile** is a value such that about $p\%$ of the population (or sample) lies below or at this value and about $(100 - p)\%$ of the population (or sample) lies above it.

6.6.1 Percentile calculation for a sample:

- First, sort data in increasing order. For the `WDS` variable in the magazine data:

$$y_{(1)} = 31, y_{(2)} = 32, y_{(3)} = 34, \ldots, y_{(n)} = 230.$$

  Here the number of observations is $n = 54$.

- Find the 5th percentile (i. e. $p = 5$):
  - The observation number corresponding to the 5-percentile is $N = \frac{54 \cdot 5}{100} = 2.7$.
  - So the 5-percentile lies between the observations with observation number $k = 2$ and $k + 1 = 3$. That is, its value lies somewhere in the interval between $y_2 = 32$ and $y_3 = 34$
  - One of several methods for estimating the 5-percentile from the value of N is defined as:

$$y_{(k)} + (N - k)(y_{(k+1)} - y_{(k)})$$

  which in this case states

$$y_2 + (2.7 - 2)(y_3 - y_2) = 32 + 0.7 \cdot (34 - 32) = 33.4$$

6.7 Median, quartiles and interquartile range

Recall

```
favstats( ~ WDS, data = magAds)
```

```
##  min Q1 median  Q3 max mean sd  n missing
##   31 69     96 202 230  123 66 54       0
```

- 50-percentile = 96 is the **median** and it is a measure of the center of data.
- 0-percentile = 31 is the **minimum** value.
- 25-percentile = 69 is called the **lower quartile** (Q1). Median of lower 50% of data.
- 75-percentile = 202 is called the **upper quartile** (Q3). Median of upper 50% of data.
- 100-percentile = 230 is the **maximum** value.
- **Interquartile Range (IQR)**: a measure of variability given by the difference of the upper and lower quartiles: 202 - 69 = 133.

7 More graphics

7.1 Box-and-whiskers plots (or simply box plots)

How to draw a box-and-whiskers plot:

- Box:
  - Calculate the median, lower and upper quartiles.
  - Plot a line by the median and draw a box between the upper and lower quartiles.
- Whiskers:
  - Calculate interquartile range and call it IQR.
  - Calculate the following values:
    * L = lower quartile - 1.5*IQR
    * U = upper quartile + 1.5*IQR
  - Draw a line from lower quartile to the smallest measurement, which is larger than L.
  - Similarly, draw a line from upper quartile to the largest measurement which is smaller than U.
- Outliers: Measurements smaller than L or larger than U are drawn as circles.

*Note: Whiskers are minimum and maximum of the observations that are not deemed to be outliers.*
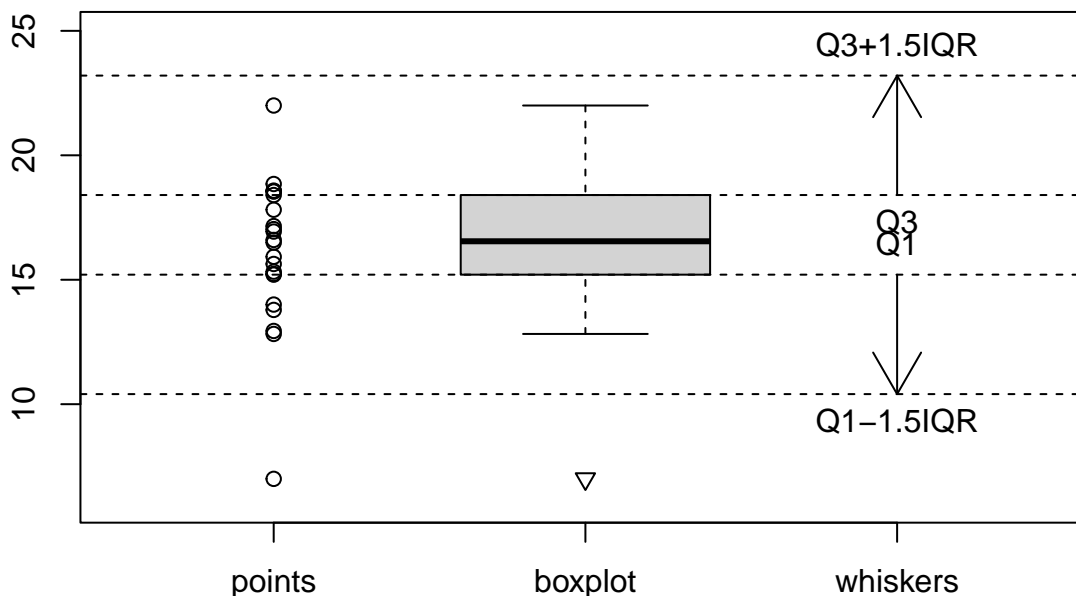
7.1.1 Boxplot for Ericksen data

Boxplot of the poverty rates separately for cities and states (variable `city`):

```
favstats(poverty ~ city, data = Ericksen)
```

```
##     city   min   Q1 median Q3 max mean   sd   n missing
## 1   city 11.5 13.8     19 21  23   18 4.0  16       0
## 2  state  6.8  9.5     11 14  24   12 3.7  50       0
```

```
gf_boxplot(poverty ~ city, data = Ericksen)
```
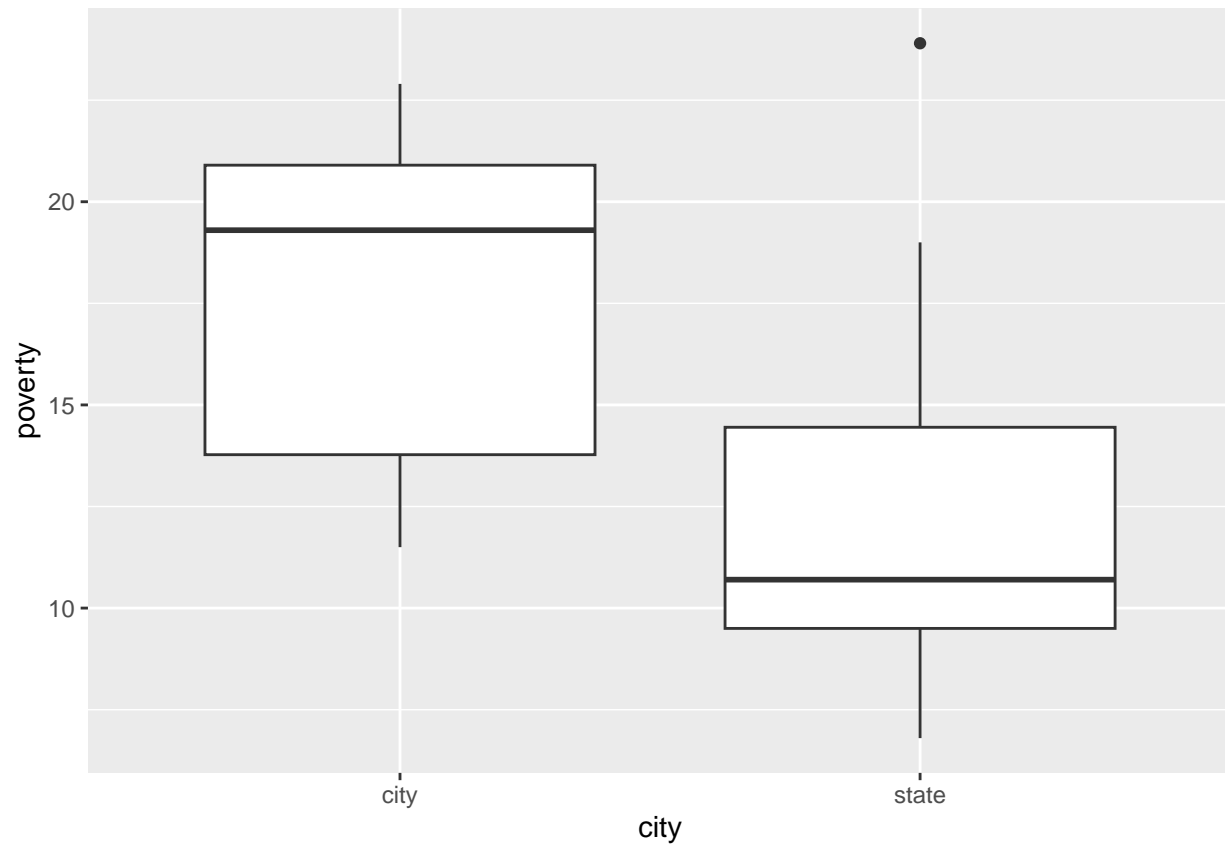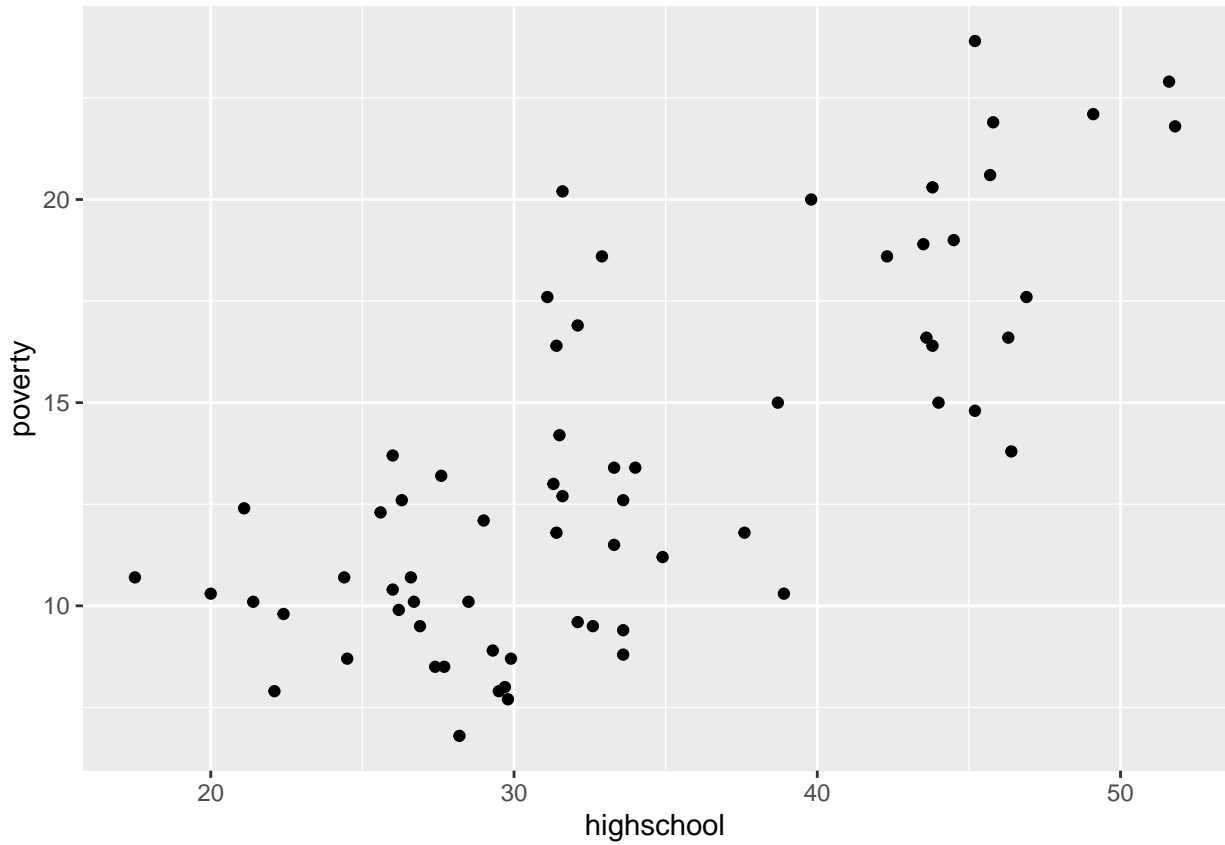

- There seems to be more poverty in the cities.
- A single state differs noticeably from the others with a high poverty rate.
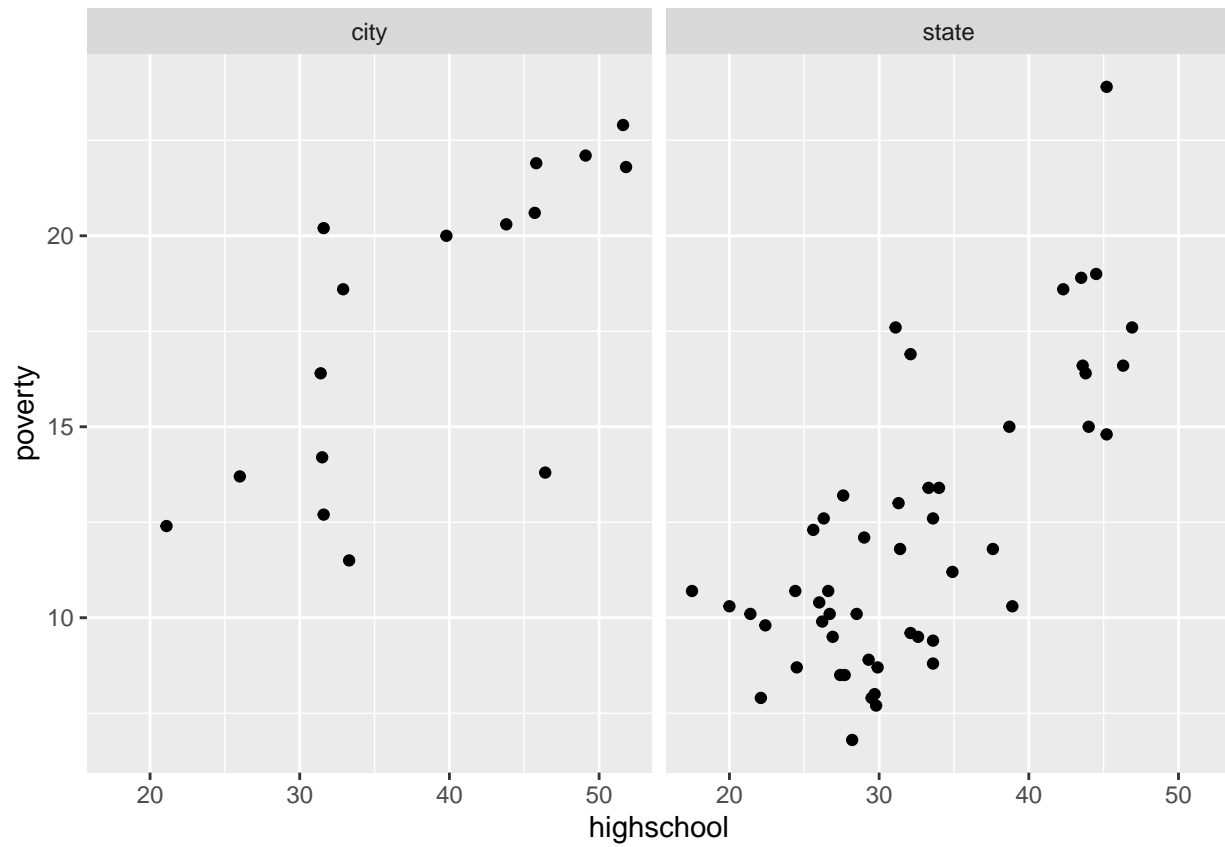
7.2 2 quantitative variables: Scatter plot

For two quantitative variables the usual graphic is a scatter plot:

```
gf_point(poverty ~ highschool, data = Ericksen)
```
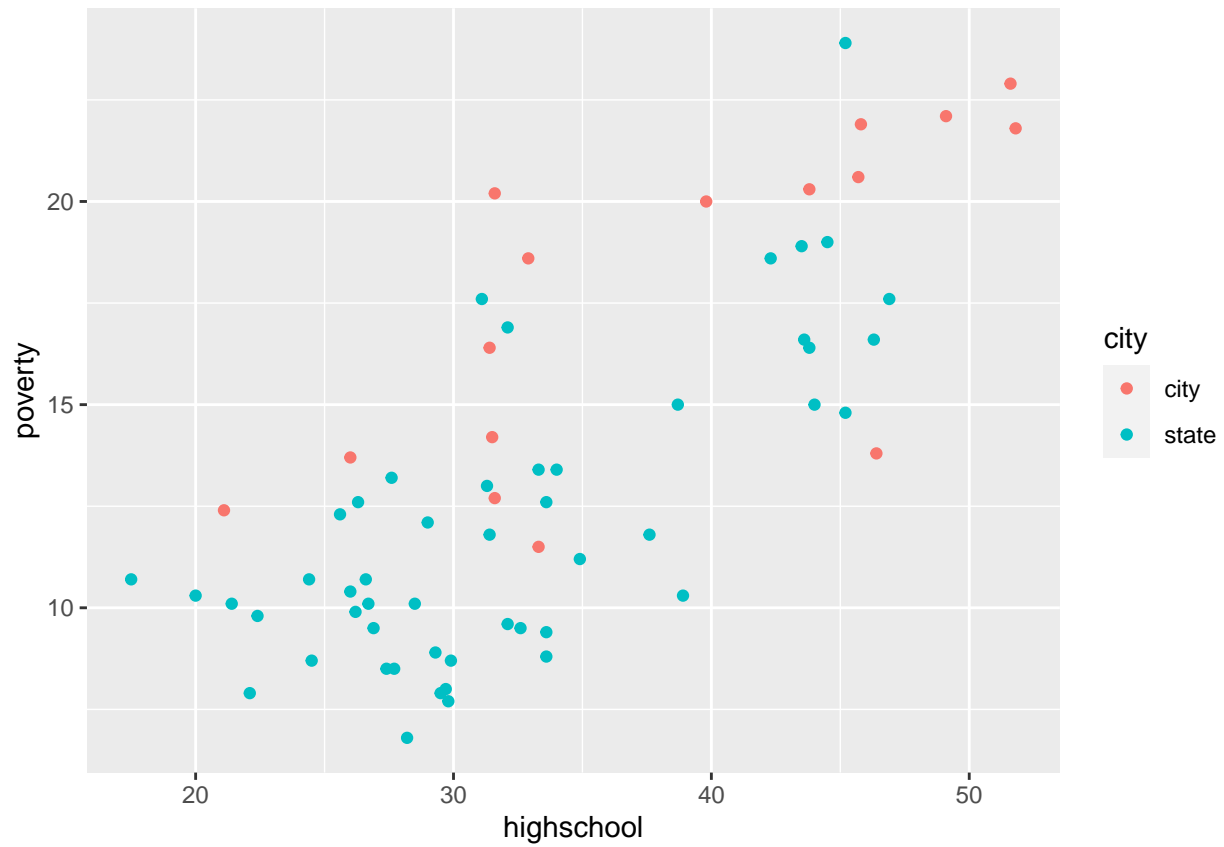
This can be either split or coloured according to the value of `city`:

```
gf_point(poverty ~ highschool | city, data = Ericksen)
```
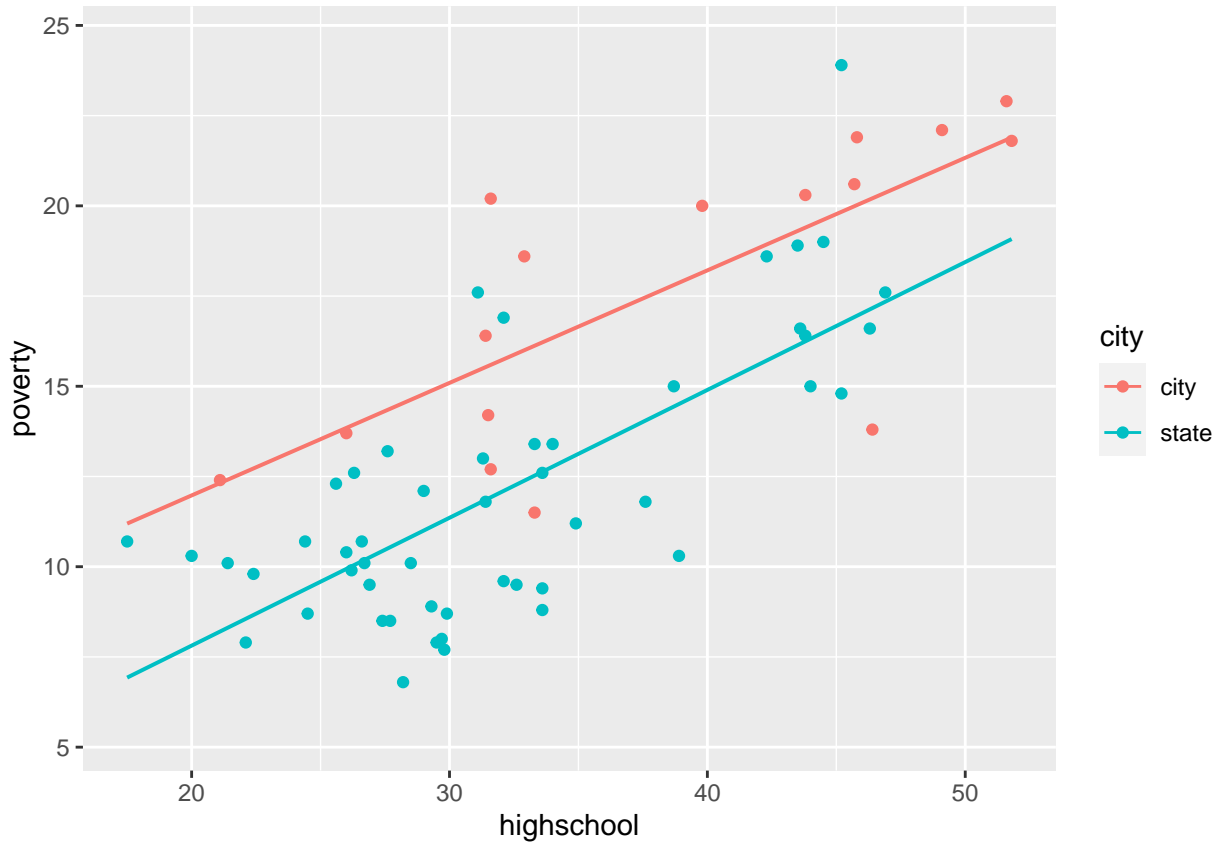
```
gf_point(poverty ~ highschool, col = ~city, data = Ericksen)
```

If we want a regression line along with the points we can do:

```
gf_point(poverty ~ highschool, col = ~city, data = Ericksen) %>% gf_lm()
```

8 Appendix

8.1 Recoding variables

- The function `factor` will directly convert a vector to be of type `factor`. E.g.:

```
head(magAds$GROUP)
```

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

```
f <- factor(magAds$GROUP)
magAds$GROUP <- f
head(magAds$GROUP)
```

```
## [1] 1 1 1 1 1 1
## Levels: 1 2 3
```

- Custom labels for the levels can also be used:

```
f <- factor(magAds$GROUP,
            levels = c("1", "2", "3"),
            labels = c("high", "medium", "low"))
magAds$GROUP <- f
head(magAds$GROUP)
```

```
## [1] high high high high high high
## Levels: high medium low
```

- In this way the numbers are replaced by more informative labels describing the educational level.