Pleas for data visualization

Prof. Dr. Nicolas Meseth

The R code for the following sections is also available as plain `.R` scripts. If you downloaded the ZIP-file and you view this as a PDF-document, you find the `.R` files in the same folder as this document.

Numbers tell only a part of the story

To illustrate why data visualization is useful, let's look at two examples. Below we read some data from a CSV-file.

```
some_data <- read_csv("data/some_data.csv")

#> # A tibble: 142 x 2
#>       x     y
#>   <dbl> <dbl>
#> 1  55.4  97.2
#> 2  51.5  96.0
#> 3  46.2  94.5
#> ...
```

As you can see, the data contains two variables `x` and `y` with 142.

If we didn't have visualization as a tool in our data analytics toolkit, we could try to get some insight into the data with descriptive statistics. For example, we could calculate the mean for both variables:

```
some_data %>%
  summarise(across(everything(), mean, .names = "{.col}_mean"))
```

```
# A tibble: 1 x 2
  x_mean y_mean
   <dbl>  <dbl>
1   54.3   47.8
```

Similarly, we could calculate a measure of spread, such as the standard deviation:

```
some_data %>%
  summarise(across(everything(), sd, .names = "{.col}_sd"))
```

```
# A tibble: 1 x 2
   x_sd  y_sd
  <dbl> <dbl>
1  16.8  26.9
```

Or other measures:

```
some_data %>%
  summarise(
    across(everything(),
           list(mean = mean, sd = sd, median = median),
           .names = "{.col}_{.fn}"
           )
    )
```

```
# A tibble: 1 x 6
  x_mean  x_sd x_median y_mean  y_sd y_median
   <dbl> <dbl>    <dbl>  <dbl> <dbl>    <dbl>
1   54.3  16.8     53.3   47.8  26.9     46.0
```

We could also calculate Pearson's correlation coefficient:

```
tibble(
  pearson = cor(some_data$x, some_data$y)
  )
```

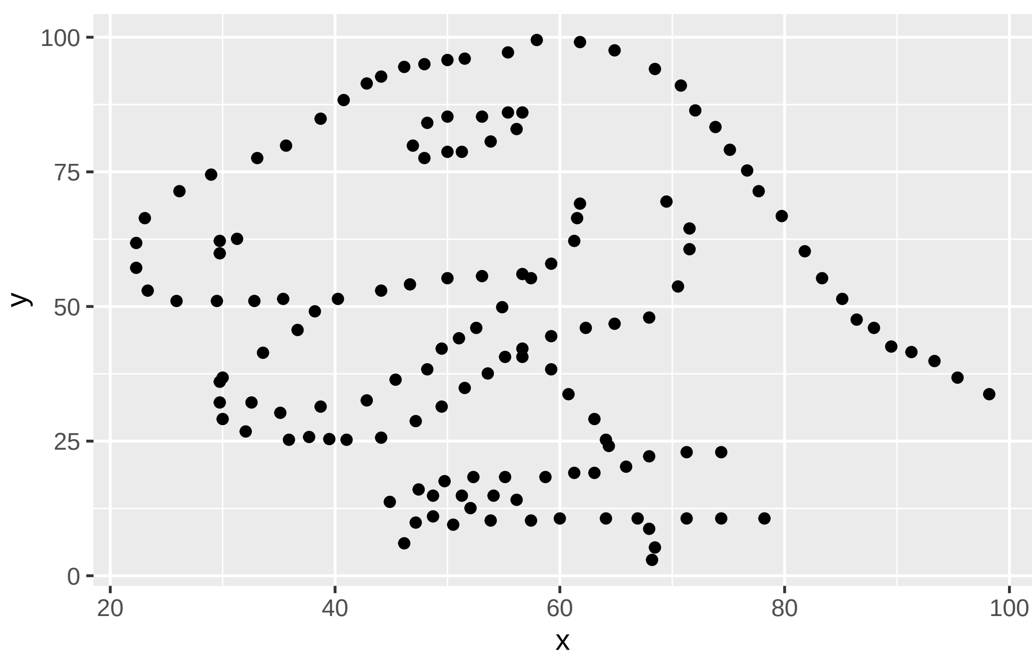
```
# A tibble: 1 x 1
  pearson
    <dbl>
1 -0.0645
```

From the rather small value, we could hypothesize that the variables are unrelated. But are they?

Visualization can reveal hidden patterns

Let's add visualization to our toolkit and find out:

```
some_data %>%
  ggplot() +
  aes(x, y) +
  geom_point()
```



The data certainly does not look unrelated to me. Of course, this an exaggerated example, but it makes the point: Only when we visualize data can we identify patterns that would otherwise stay hidden in the numbers. No statistical method could have told us there is dinosaur in the data. Well, actually it is called a datasaurus, and there is even a whole R-package with the name `{datasauRus}` dedicated to it. This packages contains the same data set, but adds more that share the same statistical measures. We could not distinguish between the data by just looking at measures such as mean, standard deviation or correlation coefficient. We would have to visualize the data:

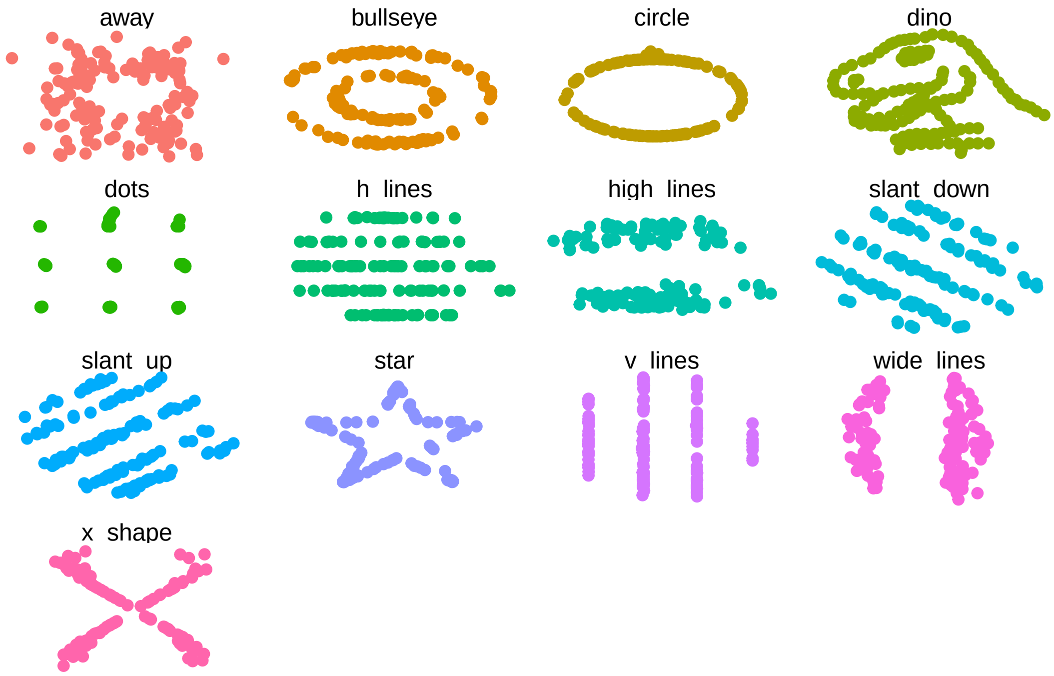
```r
#install.packages("datasauRus")
library(datasauRus)

datasaurus_dozen %>%
  group_by(dataset) %>%
  summarize(
    mean_x    = mean(x),
    mean_y    = mean(y),
    std_dev_x = sd(x),
    std_dev_y = sd(y),
    corr_x_y  = cor(x, y)
    )
```

```
# A tibble: 13 x 6
   dataset    mean_x mean_y std_dev_x std_dev_y corr_x_y
   <chr>       <dbl>  <dbl>     <dbl>     <dbl>    <dbl>
 1 away         54.3   47.8      16.8      26.9  -0.0641
 2 bullseye     54.3   47.8      16.8      26.9  -0.0686
 3 circle       54.3   47.8      16.8      26.9  -0.0683
 4 dino         54.3   47.8      16.8      26.9  -0.0645
 5 dots         54.3   47.8      16.8      26.9  -0.0603
 6 h_lines      54.3   47.8      16.8      26.9  -0.0617
 7 high_lines   54.3   47.8      16.8      26.9  -0.0685
 8 slant_down   54.3   47.8      16.8      26.9  -0.0690
 9 slant_up     54.3   47.8      16.8      26.9  -0.0686
10 star         54.3   47.8      16.8      26.9  -0.0630
11 v_lines      54.3   47.8      16.8      26.9  -0.0694
12 wide_lines   54.3   47.8      16.8      26.9  -0.0666
13 x_shape      54.3   47.8      16.8      26.9  -0.0656
```

The table shows the mean, standard deviation and correlation coefficient for all 13 data sets included in the `{datasauRus}` package. As you can see, the values are the same across all data sets. Only when we visualize do we see the different patterns in the data:

```r
datasaurus_dozen %>%
  ggplot() +
  aes(x = x, y = y, colour = dataset) +
  geom_point() +
  theme_void() +
  theme(legend.position = "none") +
  facet_wrap(~dataset, ncol = 4)
```



Anscombe's Quartet

Another and even older plea for the visualization of data can be found in Francis Anscombe's publication *Graphs in Statistical Analysis* from the year 1973. In this paper, Anscombe presented four data sets that looked very much the same when viewing the common descriptive statistical measures. Again, only by visualizing the data can we see the real patterns.

Let's load the data and see for ourselves:

```
anscombe1 <- read_csv("data/anscombe1.csv") %>%
  mutate(dataset = "1")

anscombe2 <- read_csv("data/anscombe2.csv") %>%
  mutate(dataset = "2")

anscombe3 <- read_csv("data/anscombe3.csv") %>%
  mutate(dataset = "3")

anscombe4 <- read_csv("data/anscombe4.csv") %>%
  mutate(dataset = "4")
```

We now want all four in one data frame. We can achieve this with the `union_all()` function:

```r
anscombe <-
  anscombe1 %>%
  union_all(anscombe2) %>%
  union_all(anscombe3) %>%
  union_all(anscombe4)

#> # A tibble: 44 x 3
#>       x     y dataset
#>   <dbl> <dbl> <chr>
#> 1    10  8.04 1
#> 2     8  6.95 1
#> 3    13  7.58 1
#> ...
```

We now have all four of Anscombe's Quartet in one data frame and we can distinguish the original data set by the column `dataset`. First, let's look at the descriptive statistics:

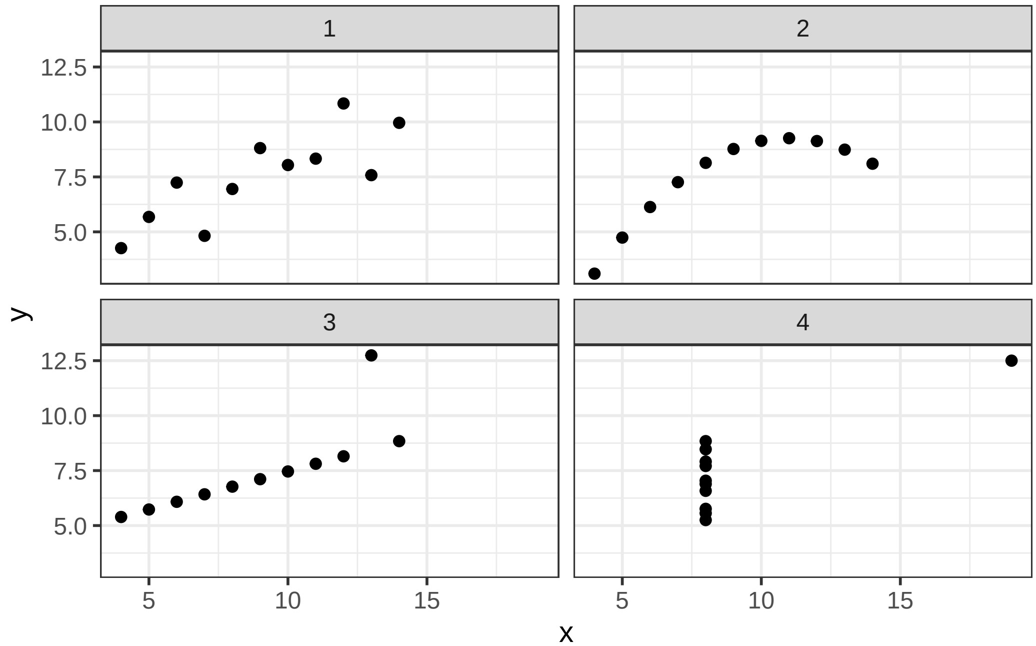
```r
anscombe %>%
  group_by(dataset) %>%
  summarize(
    mean_x    = mean(x),
    mean_y    = mean(y),
    std_dev_x = sd(x),
    std_dev_y = sd(y),
    corr_x_y  = cor(x, y)
    )
```

```
# A tibble: 4 x 6
  dataset mean_x mean_y std_dev_x std_dev_y corr_x_y
  <chr>    <dbl>  <dbl>     <dbl>     <dbl>    <dbl>
1 1            9   7.50      3.32      2.03    0.816
2 2            9   7.50      3.32      2.03    0.816
3 3            9   7.5       3.32      2.03    0.816
4 4            9   7.50      3.32      2.03    0.817
```

As expected, all measures look the same for all 4 data sets. But again, a plot reveals the truth:

```r
anscombe %>%
  ggplot() +
  aes(x, y) +
```

```
  geom_point() +
  theme_bw() +
  theme(legend.position = "none") +
  facet_wrap(~dataset, ncol = 2)
```



The first plot shows a linear trend with some noise, as we might already have suspected from a correlation coefficient of roughly 0.81. The second plot, although having the same correlation coefficient, displays a obviously non-linear trajectory. The third plot would have had a perfect correlation if it wasn't for the single outlier. In contrast, the last plot would have had no correlation between `x` and `y` if the point on the very top-right didn't exist. Again, we could not have gotten this insight from any statistical measure we can calculate.

I hope the examples convinced you of the importance of data visualization in data analytics. There are even more good reasons why we should visualize data, besides the fact that otherwise couldn't reveal hidden patterns. We know from psychological research about the way humans process information that the visualizations are a much faster way into our brains. We can not only grasp what we see in a good data visualization faster, but also comprehend it better and create a better memory of it. If that doesn't convince you, nothing will.

References

- The official website of the `{datasauRus}` package

- YouTube video on Anscombe's Quartet
- Original Paper *Graphs in Statistical Analysis* by Francis Anscombe