Getting started with graphics

Erasmus Q-Intelligence B.V.

Data Science and Business Analytics
Programming

Content

1. Built-in R graphics versus package `ggplot2`
2. Basic plots
3. Conclusions

References to Online Book

- Chapter 1*

* for this lecture (Graphics) as well as next lecture (Advanced Graphics)

Built-in R graphics versus package `ggplot2`

The usual suspect

Function `plot()`:

⟶ Scatterplot matrix for data frame

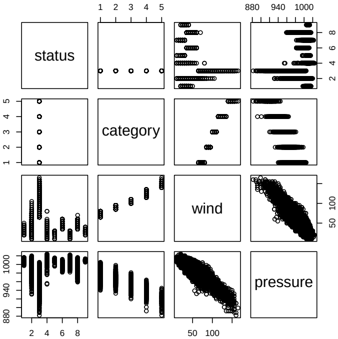
⟶ Works with many other objects, e.g., density estimates, linear models

⟶ Whatever analysis you do, always check if you can `plot()` the result

Scatterplot matrix

```
R> data(storms, package = 'dplyr')
R> ?storms
```

```
R> plot(select(storms, status, category, wind, pressure))
```



Built-in R graphics

- + Allow the user to create quick plots for exploring the data
- + Easy to add elements to an existing plot
- + Fine tuning to produce high-quality graphics for publications

- – Designed in the 1970s/80s
- – Sometimes inconsistencies in usage or behavior
- – Customization via cryptic graphical parameters (see `?par`)

⟶ Murrell (2011): R Graphics

The grammar of graphics

- Designed with recent research on data visualization and human perception in mind
- Focused on coherence between geometry of the data and geometry of the plot

⟶ The visual representation should fit the data

⟶ Always need to explicitly specify what variables to use and how to plot them

⟶ Implemented in package `ggplot2`

⟶ Wickham (2009): `ggplot2`: Elegant Graphics for Data Analysis

Package `ggplot2`

+ Coherent approach to graphics
+ Highly flexible and customizable via options and layers
+ Pretty plots (subjective)

– Steeper learning curve than built-in R graphics

– Often not straightforward to add elements to the plot

– Slow even for moderately sized data sets

Basic usage of `ggplot2`

Add together two basic elements:

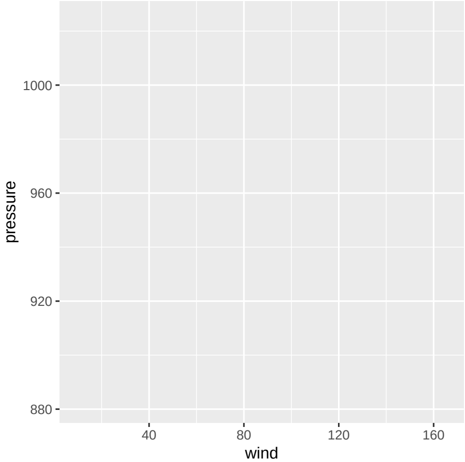
1. Scaffolding defined by `ggplot()`
    - Selects the data set
    - Defines the variables to be used (the aesthetic mapping): function `aes()`
2. Any number of visual representations of the data, known as geoms
    - Define the visual representation (the geometric objects): function family `geom_xxx()`
    - Different elements are added to the plot using the + operator

```
R> library("ggplot2")
```

Basic plots

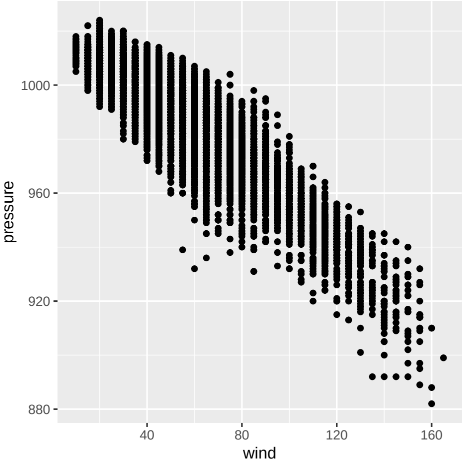
Scatterplot: Scaffolding

```
R> ggplot(storms, aes(x = wind, y = pressure))
```



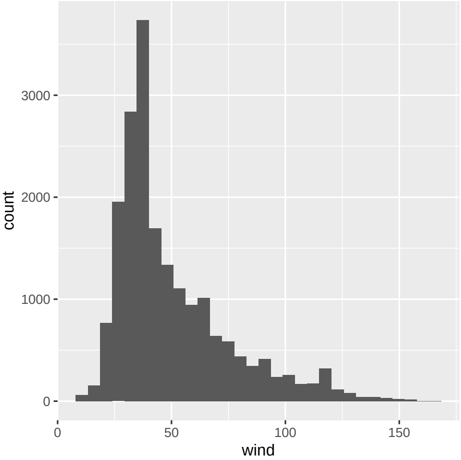
Scatterplot: Scaffolding + points

```
R> ggplot(storms, aes(x = wind, y = pressure)) + geom_point()
```



Histogram

```
R> ggplot(storms, aes(x = wind)) + geom_histogram()
```



Histogram: number of bins

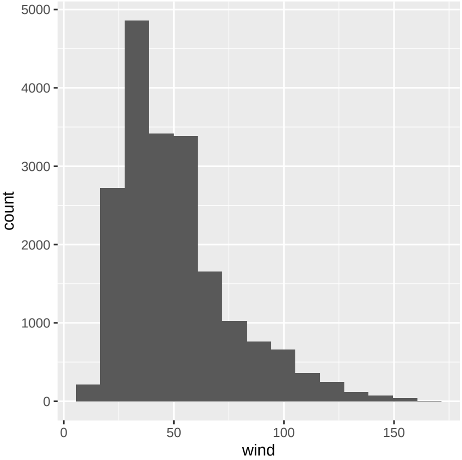
⟶ For histograms, it is always a good idea to play with the number of bins

⟶ Number of bins can be specified with argument `bins`

⟶ Bin width can be specified with argument `binwidth`

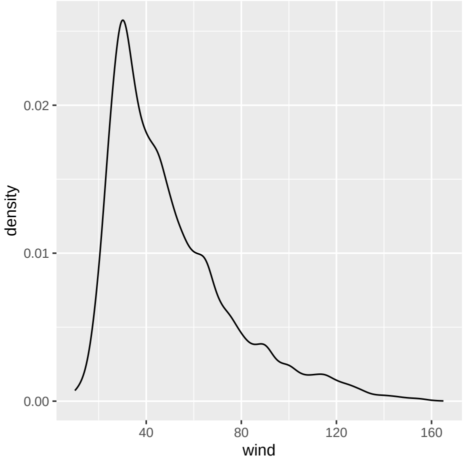
Histogram: number of bins

```
R> ggplot(storms, aes(x = wind)) + geom_histogram(bins = 15)
```



Density plot

```
R> ggplot(storms, aes(x = wind)) + geom_density()
```



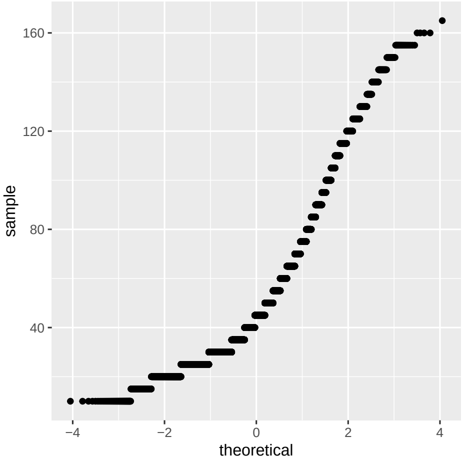
Density plot: kernel and bandwidth

- Density estimate depends on the kernel and smoothing bandwidth
- Default Gaussian kernel is symmetric and therefore not optimal for asymetric distributions

⟶ Still useful to get an insight on the shape of the distribution, but be aware of those issues

Quantile-quantile plot

```
R> ggplot(storms, aes(sample = wind)) + geom_qq()
```



Quantile-quantile plot: straight line?

⟶ Plot sample quantiles against theoretical quantiles

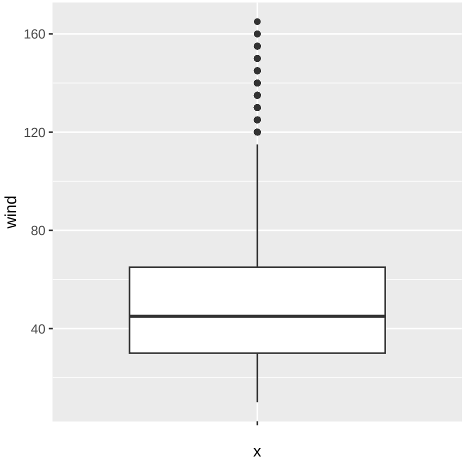
⟶ If the distributional assumption holds, the points form almost a straight line

⟶ By default the normal distribution is used

⟶ Distribution can be specified with argument `distribution`

Boxplot

```
R> ggplot(storms, aes(x = "", y = wind)) + geom_boxplot()
```



Boxplot statistics

Upper whisker Largest point still within $1.5 \cdot IQR$ of the upper quartile

Top of box Upper quartile (i.e., 75% quantile)

Middle line Median (i.e., 50% quantile)

Bottom of box Lower quartile (i.e., 25% quantile)

Lower whisker Smallest point still within $1.5 \cdot IQR$ of the lower quartile

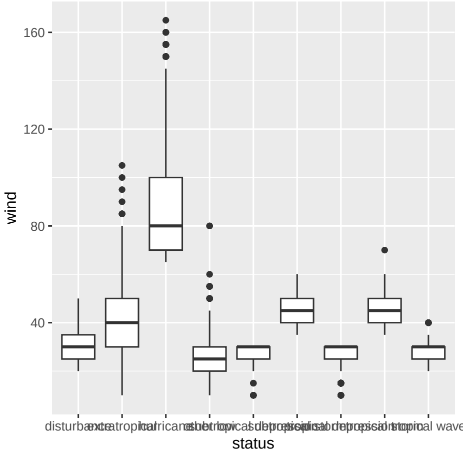
IQR Interquartile range (i.e., difference between upper and lower quartile)

$\longrightarrow$ No assumption about statistical distribution

$\longrightarrow$ But: definition of whiskers assumes some degree of symmetry

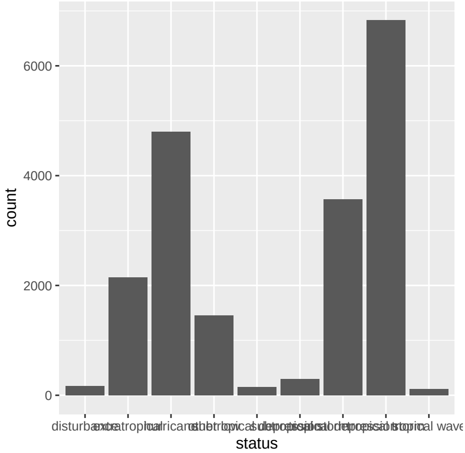
Conditional boxplot

```
R> ggplot(storms, aes(x = status, y = wind)) + geom_boxplot()
```



Barplot

```
R> ggplot(storms, aes(x = status)) + geom_bar()
```



Time series plot

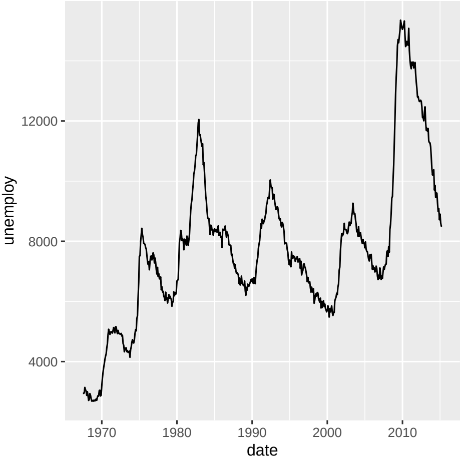
⟶ Simply use `geom_line()` instead of `geom_point()` to draw connected line instead of scattered points

⟶ Example: US economic time series

```
R> data(economics, package = 'ggplot2')
R> ?economics
```

Time series plot

```
R> ggplot(economics, aes(x = date, y = unemploy)) + geom_line()
```



Some geoms

For a complete list of geoms, click here. Important ones include:

`geom_point()`	Points
`geom_line()`	Lines / time series
`geom_hline()`	Horizontal lines
`geom_vline()`	Vertical lines
`geom_bar()`	Bars
`geom_boxplot()`	Box and whiskers plot
`geom_density()`	Density estimate
`geom_smooth()`	Fitted regression line
`geom_text()`	Text
`geom_label()`	Text within rectangle
`geom_tile()`	Rectangles for heat maps

⟶ Use appropriate geoms!

Exercises

Load the `patents` data from the `patents.Rds` file, and do Exercise 1.1.

Conclusions

Conclusions

- Basic function `ggplot()` to initialize the plot
- Function `aes()` to define the variables to be used
- Function family `geom_xxx()` to define the visual representation
- Use scripts for reproducibility of the plots

References

P. Murrell. **R Graphics**. Chapman & Hall/CRC, 2nd edition, 2011.

H. Wickham. `ggplot2`**: Elegant Graphics for Data Analysis**. Springer-Verlag, 2009.