

Advanced Graphics

Erasmus Q-Intelligence B.V.

Data Science and Business Analytics
Programming



Content

- 1 Software requirements
- 2 Finetuning graphics
- 3 Adding to plots
- 4 Conclusions



References to Online Book

- Chapter 1*
- Chapter 11

* for this lecture (Advanced Graphics) as well as the previous lecture (Graphics)



Software requirements



Software requirements

- Data from packages `dplyr` and `ggplot2` are used
- Functions from packages `dplyr` and `ggplot2` are applied (automatically installed with the `tidyverse` package)

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



Data sets

Atlantic hurricane database track data, 1975-2015

```
R> data(storms, package = "dplyr")  
R> storms_2015 <- filter(storms, year == 2015)
```

US economic time series

```
R> data(economics, package = "ggplot2")
```

Eredivisie points of Ajax and Feyenoord

```
R> eredivisie <- readRDS('../../data/eredivisie.Rds')
```



Finetuning graphics



Arguments for finetuning

Finetuning through **arguments**:

`color` Color for points/lines

`fill` Fill color for areas

`alpha` Transparency of colors

`shape` Symbol for points

`linetype` Type of line

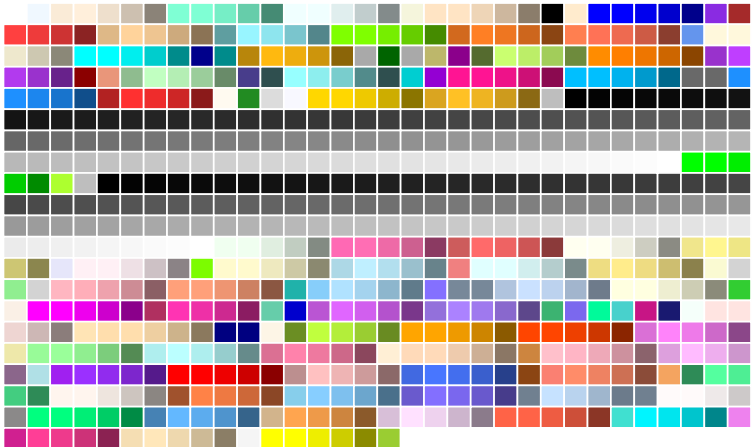
`size` Size of points/lines

→ If the **same setting** should be used for all points/lines/areas, it is best to **supply those arguments to `geom_xxx()`**



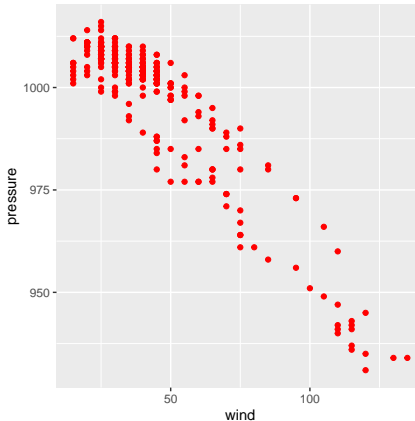
Named colors

```
R> colors()
```



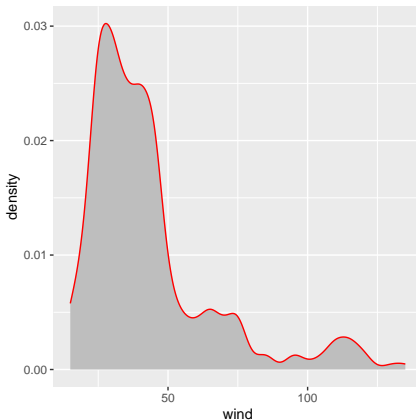
Named colors: scatterplot

```
R> ggplot(storms_2015, aes(x = wind, y = pressure)) +  
+   geom_point(color = "red")
```



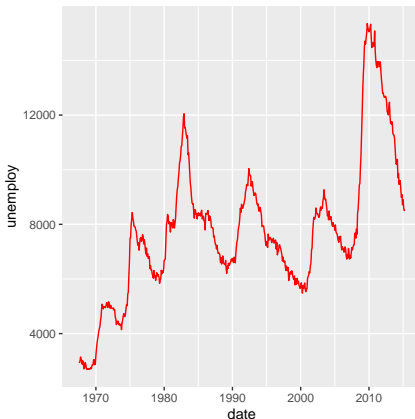
Named colors: density plot

```
R> ggplot(storms_2015, aes(x = wind)) +  
+   geom_density(color = "red", fill = "grey")
```














Named colors: time series plot

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



Plot symbols

0  1  2  3  4  5 

6  7  8  9  10  11 

12  13  14  15  16  17 

18  19  20  21  22  23 

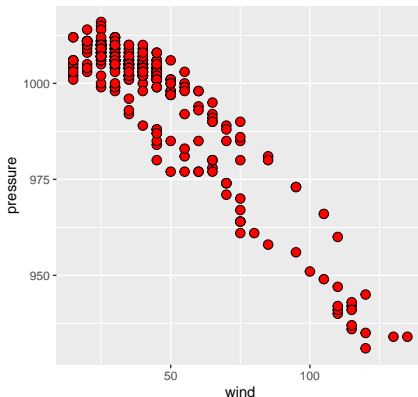
24  25  "A" A "b" b "." · "#" #

→ Specify fill color for symbols 21–25 with parameter `fill`



Plot symbols: scatterplot

```
R> ggplot(storms_2015, aes(x = wind, y = pressure)) +  
+   geom_point(color = "black", fill = "red",  
+             shape = 21, size = 3)
```



Line types: density plot

0 "blank"

1 "solid"



2 "dashed"



3 "dotted"



4 "dotdash"



5 "longdash"



6 "twodash"



→ Specify line type by index or character string



Line types: density plot

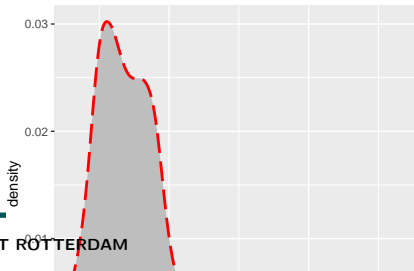
```
R> ggplot(storms_2015, aes(x = wind)) +  
+   geom_density(color = "red", fill = "grey",  
+               linetype = 5, size = 1)
```

Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.

i Please use 'linewidth' instead.

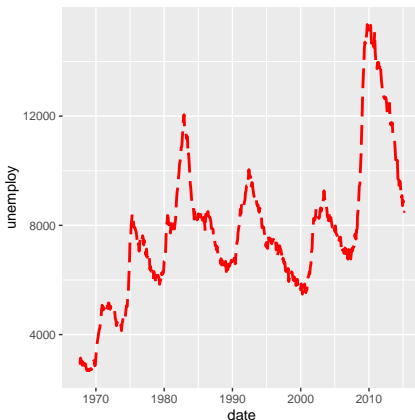
This warning is displayed once every 8 hours.

Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was generated.



Line types: time series plot

```
R> ggplot(economics, aes(x = date, y = unemploy)) +  
+   geom_line(color = "red", linetype = "longdash", size = 1)
```



Arguments for finetuning revisited

Finetuning through **arguments**:

`color` Color for points/lines

`fill` Fill color for areas

`alpha` Transparency of colors

`shape` Symbol for points

`linetype` Type of line

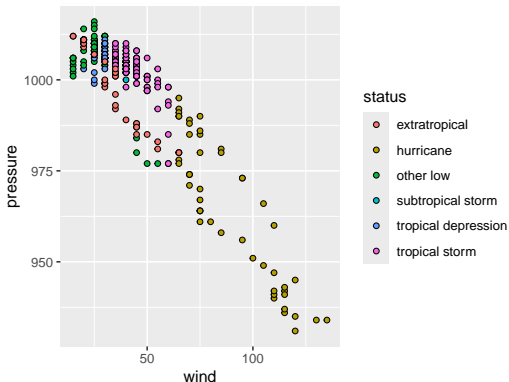
`size` Size of points/lines

- If the **values should depend on some variable**, those arguments should be **supplied to `aes()`**
(information from those variables is mapped to the visual representation)



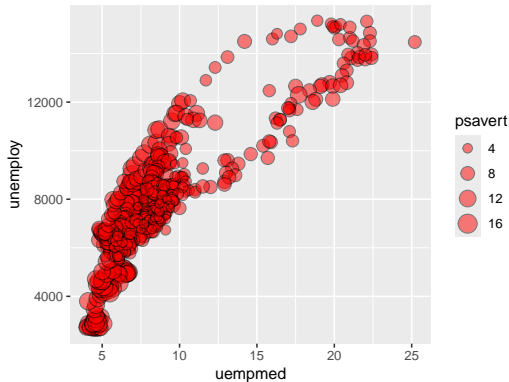
Colors: scatterplot

```
R> ggplot(storms_2015,  
+         aes(x = wind, y = pressure, fill = status)) +  
+         geom_point(shape = 21, color = "black")
```



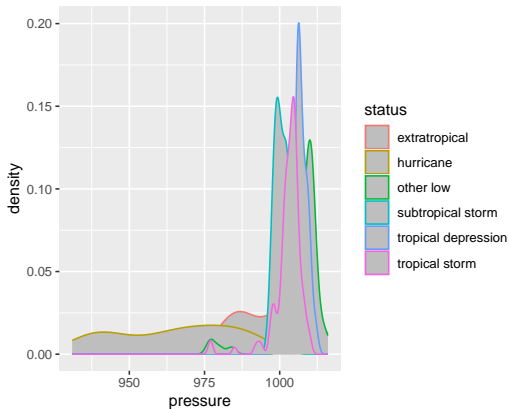
Size: scatterplot

```
R> ggplot(economics,  
+         aes(x = uempmed, y = unemploy, size = psavert)) +  
+         geom_point(shape = 21, color = "black",  
+                   fill = "red", alpha = 0.5)
```



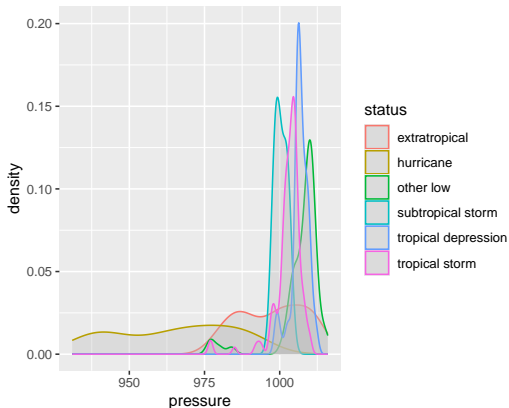
Colors: density plot

```
R> ggplot(storms_2015, aes(x = pressure, color = status)) +  
+   geom_density(fill = "grey")
```



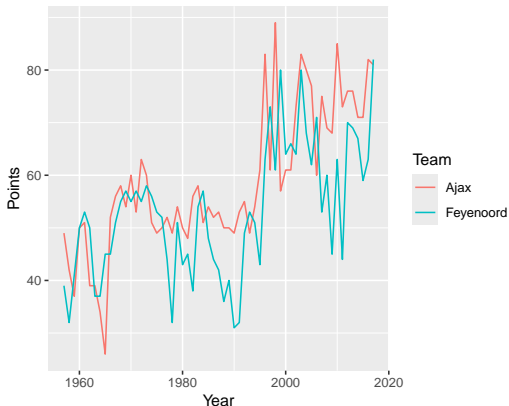
Colors: density plot

```
R> ggplot(storms_2015, aes(x = pressure, color = status)) +  
+   geom_density(fill = "grey", alpha = 0.35)
```



Colors: time series plot

```
R> ggplot(eredivisie, aes(x = Year, y = Points, color = Team)) +  
+   geom_line()
```



Exercises

- Download file *AdvancedGraphics-Exercises.pdf* from Canvas and open it
- Do Exercise 1



Manual scales

→ Built-in scales are used to obtain default values

Manual scales through function family `scale_xxx_manual()`:

<code>scale_color_manual()</code>	Color for points/lines
<code>scale_fill_manual()</code>	Fill color for areas
<code>scale_alpha_manual()</code>	Transparency of colors
<code>scale_shape_manual()</code>	Symbol for points
<code>scale_linetype_manual()</code>	Type of line
<code>scale_size_manual()</code>	Size of points/lines

→ Useful if points/lines/areas depend on a categorical variable

→ Custom values are supplied via argument values

→ Scales are added to a plot with the + operator



Plot symbols: scatterplot

```
R> ggplot(storms_2015,  
+       aes(x = wind, y = pressure, shape = status)) +  
+       geom_point() + scale_shape_manual(values = 1:3)
```

```
Error in 'palette()':  
! Insufficient values in manual scale. 6 needed but only 3  
provided.
```



Plot symbols: scatterplot

```
R> ggplot(storms_2015, aes(x = wind, y = pressure,  
+                           shape = status, color = status)) +  
+   geom_point(size = 3) +  
+   scale_shape_manual(values = c("h", "d", "s")) +  
+   scale_color_manual(values = c("blue", "goldenrod1", "red"),  
+                       na.translate = FALSE)
```

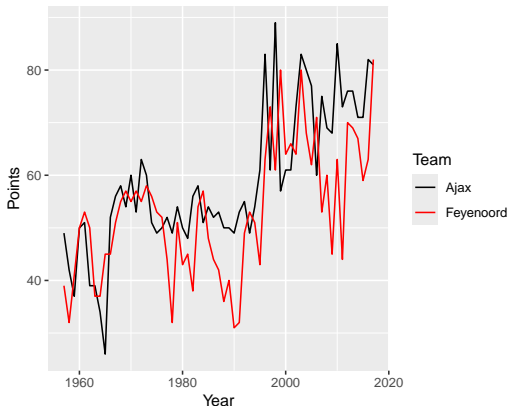
Error in 'palette()':

! Insufficient values in manual scale. 6 needed but only 3 provided.



Colors: time series plot

```
R> ggplot(eredivisie, aes(x = Year, y = Points, color = Team)) +  
  + geom_line() + scale_color_manual(values = c("black", "red"))
```



Continuous scales

Continuous scales through function family

`scale_xxx_continuous()`:

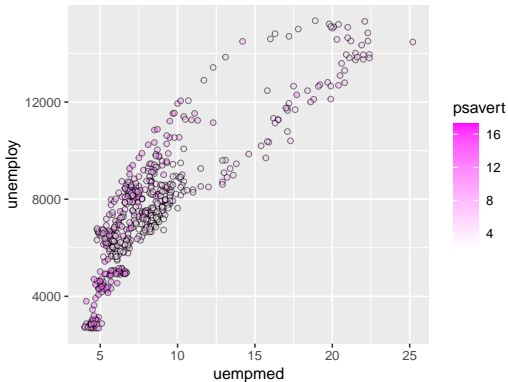
<code>scale_color_continuous()</code>	Color for points/lines
<code>scale_fill_continuous()</code>	Fill color for areas
<code>scale_alpha_continuous()</code>	Transparency of colors
<code>scale_size_continuous()</code>	Size of points/lines

- Useful if points/lines/areas depend on a continuous variable
- Scales are added to a plot with the + operator



Colors: scatterplot

```
R> ggplot(economics, aes(x = uempmed, y = unemploy,  
+                          fill = psavert)) +  
+   geom_point(shape = 21, color = "black", alpha = 0.5) +  
+   scale_fill_continuous(low = "white", high = "magenta")
```



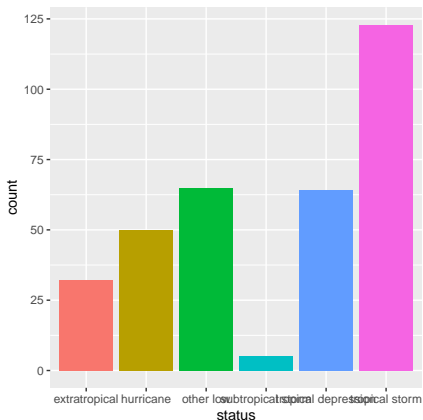
Removing legends

- For some plots, the legend might not provide new information
- Legend can be removed by setting the argument `show.legend` to `FALSE`



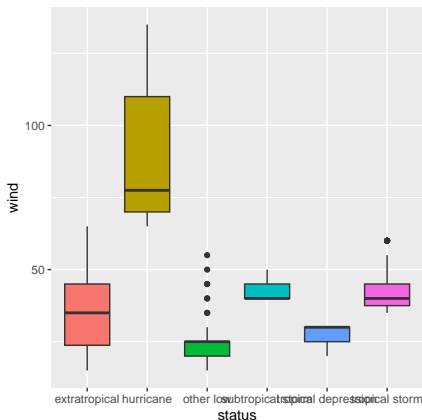
Removing legends: barplot

```
R> ggplot(storms_2015, aes(x = status, fill = status)) +  
+   geom_bar(show.legend = FALSE)
```



Removing legends: conditional boxplots

```
R> ggplot(storms_2015, aes(x = status, y = wind, fill = status)) +  
+   geom_boxplot(show.legend = FALSE)
```



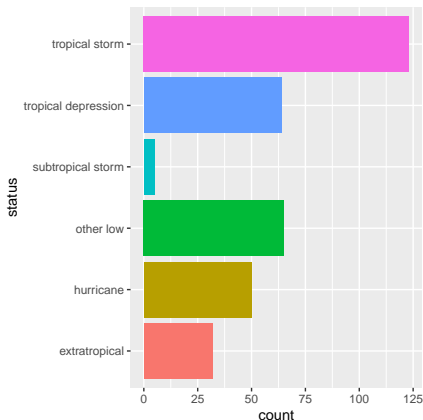
Flipping coordinates

- Some plots (e.g., barplots or conditional boxplots), are better displayed horizontally than vertically
 - Lengths are easier to judge for humans than heights
 - Easier to fit group labels
- Add function `coord_flip()` to the plot with the `+` operator



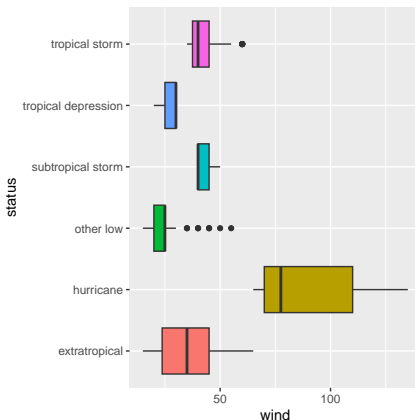
Flipping coordinates: barplot

```
R> ggplot(storms_2015, aes(x = status, fill = status)) +  
+   geom_bar(show.legend = FALSE) + coord_flip()
```



Flipping coordinates: conditional boxplots

```
R> ggplot(storms_2015, aes(x = status, y = wind, fill = status)) +  
+   geom_boxplot(show.legend = FALSE) + coord_flip()
```



Printer-friendly graphics

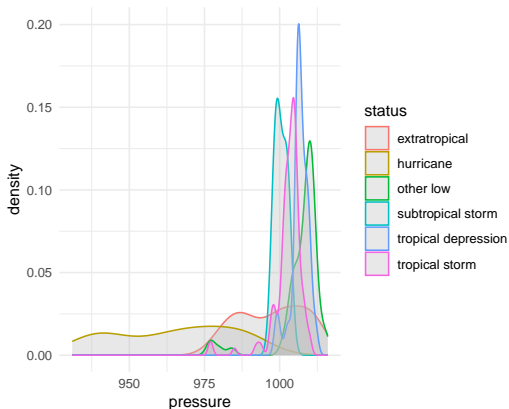
- Default grey background is less tiring for the eyes on screen
- But for printing, a white background may be preferred

- Add function `theme_minimal()` to the plot with the `+` operator
- Custom theme using function `theme()` instead



Printer-friendly graphics: density plot

```
R> ggplot(storms_2015, aes(x = pressure, color = status)) +  
+   geom_density(fill = "grey", alpha = 0.35) + theme_minimal()
```



Axis limits and annotation

Axis limits:

`xlim()` x-Axis limits

`ylim()` y-Axis limits

→ Two values for lower and upper limit

Annotation: function `labs()` with the following arguments

`title` Plot title

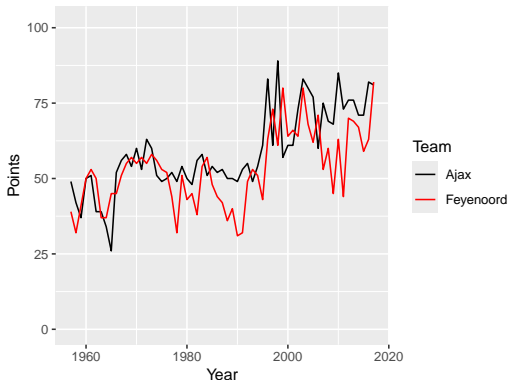
`x` x-Axis label

`y` y-Axis label



Axis limits: time series plot

```
R> ggplot(eredivisie, aes(x = Year, y = Points, color = Team)) +  
+   geom_line() + scale_color_manual(values = c("black", "red")) +  
+   ylim(0, 102)
```



Exercises

→ Open file *AdvancedGraphics-Exercises.pdf*

→ Do Exercise 2



Adding to plots



Layers

- ggplot2 works in layers, each `geom_xxx()` defines one layer
- Data and aesthetic mapping can be specified separately for each layer in the corresponding call to `geom_xxx()`
- Allows to add additional information to a plot



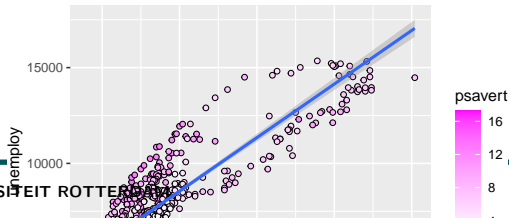
Adding a scatterplot smoother

```
R> ggplot(economics,  
+       aes(x = uempmed, y = unemploy, fill = psavert)) +  
+   geom_point(shape = 21, color = "black") +  
+   scale_fill_continuous(low = "white", high = "magenta") +  
+   geom_smooth(method = "lm", show.legend = FALSE)
```

Warning: The following aesthetics were dropped during statistical transformation: fill.

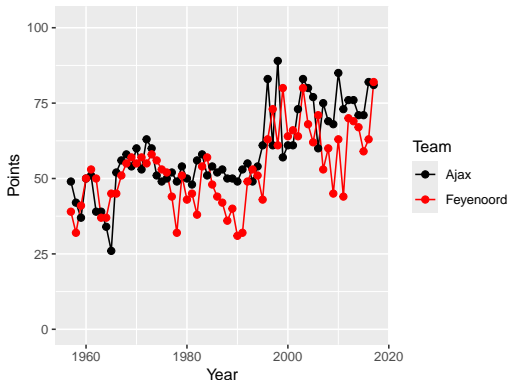
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a 'group' aesthetic or to convert a numerical variable into a factor?



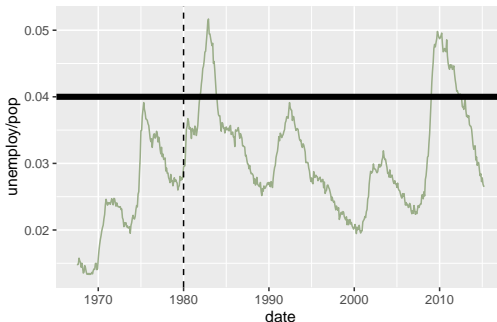
Plotting both points and lines

```
R> ggplot(eredivisie, aes(x = Year, y = Points, color = Team)) +  
+   ylim(0, 102) + geom_line() + geom_point(size = 2) +  
+   scale_color_manual(values = c("black", "red"))
```



Adding vertical lines

```
R> ggplot(economics, aes(x = date, y = unemploy/pop)) +  
+   geom_line(color = '#9aad88') +  
+   geom_vline(aes(xintercept = as.Date('1980-01-01')),  
+               linetype = "dashed") +  
+   geom_hline(aes(yintercept = 0.04), size = 2)
```



Exercises

→ Open file *AdvancedGraphics-Exercises.pdf*

→ Do Exercise 3 and 4



Conclusions



Conclusions

- Package ggplot2 offers a coherent yet flexible framework to produce **high-quality graphics** for publications

