Data Visualization with R
Ggplot2 tutorial (part 2)

Ariane Ducellier

University of Washington - Fall 2025

Themes

Theme is used to change the non-data elements of the plot:

Theme	Type	Arguments
axis.title.x	element_text	size, color, family, angle
axis.title.y	element_text	size, color, family, angle
plot.background	element_rect	fill, color, linewidth
panel.background	element_rect	fill, fill, color, line width
panel.grid.major	element_line	color, linetype, linewidth

Type `?theme` to show all possible types of themes, their types and their arguments.

Themes

You can add themes to the plot to customize the non-data elements:

```
p1 <- ggplot(dfn, aes(Genre, WorldGross))
p2 <- p1+ geom_bar(aes(fill=LeadStudio),
                   stat="Identity",
                   position="dodge")
p3 <- p2 + theme(axis.title.x=element_text(size=15),
                 axis.title.y=element_text(size=15),
plot.background=element_rect(fill="gray87"),
panel.background=element_rect(fill="beige"),
panel.grid.major=element_line(color="Gray",
                              linetype=1))
```

Themes

You can use predefined themes:

```
p2 + theme_bw() + ggtitle("theme_bw()")
p2 + theme_classic() + ggtitle("theme_classic()")
p2 + theme_classic() + ggtitle("theme_gray()")
p2 + theme_minimal() + ggtitle("theme_minimal()")
```

Themes

You can also use define your own theme:

```
mytheme <- theme(legend.title=element_blank(),
           legend.position="bottom",
           text=element_text(color="Blue"),
           axis.text=element_text(size=12,
                                  color="Red"),
           axis.title=element_text(size=rel(1.5)))
```

and use it for a single plot:

```
p2 + mytheme + ggtitle("Changed Plot with my theme")
```

or for all the plots by placing it at the beginning of your code:

```
theme_set(my_theme)
```

Themes

You can change the color palette.

Type `?scale_fill_brewer` to see all the color palettes available.

```
p4 <- p2 + theme_bw() + ggtitle("theme_bw()")
p4 + scale_fill_brewer(palette="Spectral")
```

Bubble charts

You can make the size of each marker proportional to a variable:

```
ggplot(dfs, aes(x=Year,
                y=Electricity_consumption_per_capita))
+ geom_point(aes(size=population, color=Country))
+ scale_size(breaks=c(0, 1e+8, 0.3e+9, 0.5e+9,
                      1e+9, 1.5e+9), range=c(1, 5))
```

Density plots

Instead of plotting an histogram, ggplot2 will estimate the density before plotting it:

```
ggplot(df, aes(x=loan_amnt)) +
geom_density() +
facet_wrap(~grade)
```

It is also possible to superimpose the density plots:

```
ggplot(df, aes(x=loan_amnt)) +
geom_density(aes(fill=grade), alpha=1/2) +
scale_fill_brewer(palette="Dark2")
```

Time series plots

Use`geom_line` to make time series plots:

```
df_fb$Date <- as.Date(df_fb$Date)
ggplot(df_fb, aes(x=Date, y=Close, group=1)) +
geom_line(color="black", na.rm=TRUE) +
scale_x_date(date_breaks='3 month')
```

We use `group=1` one there is only one line to be show.

Use `group=MyCategory` to plot a line per value of a categorical variable.

Statistical summaries

You can add statistical summaries of the data (e.g. mean, median, quantiles) to the plot:

```
ggplot(df_fb, aes(Month, Close)) +
geom_point(color="red", alpha=1/2,
           position=position_jitter(h=0.0, w=0.0)) +
stat_summary(geom="line", fun="quantile",
             fun.args=list(probs=.1), linetype=2,
             color="green", size=1) +
stat_summary(geom="line", fun="quantile",
             fun.args=list(probs=.9), linetype=2,
             color="green", size=1)
```

Linear regression

Ggplot can also fit a linear regression to the data and add it to the plot. It may be done for all the data:

```
ggplot(dfs, aes(gdp_per_capita,
                Electricity_consumption_per_capita)) +
geom_point(aes(color=Country)) +
stat_smooth(method=lm)
```

Linear regression

Or you can fit a linear regression to each category:

```
ggplot(dfs, aes(gdp_per_capita,
                Electricity_consumption_per_capita,
                color=Country)) +
geom_point() +
stat_smooth(method=lm)
```

Correlations

You can also plot correlations between different variables:

```
M <- cor(df)
corrplot(M, method="number")
corrplot(M, method="pie")
corrplot(M, method="ellipse")
```

Maps

You can use maps from the `maps` and `map_data` packages to plot maps:

```
world_map <- map_data("world")
states_map <- map_data("state")
```

You can select only some polygons from the map using:

```
europe <- map_data("world",
region=c("Germany", "Spain", "Italy",
         "France", "UK", "Ireland"))
```

Maps

To plot the data, we may use geom_polygom or geom_path.

The projection is defined with coord_map.

```
ggplot(states_map, aes(x=long, y=lat, group=group)) +
geom_polygon(fill="white", color="black")
```

```
ggplot(states_map, aes(x=long, y=lat, group=group)) +
geom_path() +
coord_map("mercator")
```