Data Visualisation with R
Workshop Part 1

Grammar of Graphics

Presented by Emi Tanaka
Department of Econometrics and Business Statistics

MONASH University

6th Dec 2021 @ Statistical Society of Australia NSW Branch | Zoom
Why grammar of graphics for data visualisation?
Constructing plots with R: base version

df

```r
## # A tibble: 3 × 2
##   duty      perc
##   <chr>    <dbl>
## 1 Teaching    40
## 2 Research    40
## 3 Admin       20
```

Stacked barplot

```r
barplot(as.matrix(df$perc),
        legend = df$duty)
```

Pie chart

```r
pie(df$perc, labels = df$duty)
```

- Single purpose functions to generate "named plots"
- Input varies, here it is vector or matrix
How are **barplots** different to a **pie chart**? 🤔

Don't they all **depict the same information**? 😌
Grammar of graphics

Basic structure of ggplot

```
ggplot(data = <data>, mapping = aes(<mappings>)) + <layer>()
```

1. **data** as `data.frame` (or tibble),
2. a set of **aesthetic** mappings between variables in the data and visual properties, and
3. at least one **layer** which describes how to render each observation.
penguins data is from the palmerpenguins 📦

```r
library(palmerpenguins)
glimpse(penguins)
```

```
## Rows: 344
## Columns: 8
## $ species           <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Adel…
## $ island            <fct> Torgersen, Torgersen, Torgersen, Torgersen, Torgerse…
## $ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.9, 39.2, 34.1, …
## $ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.8, 19.6, 18.1, …
## $ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195, 193, 190, 186…
## $ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 3625, 4675, 3475, …
## $ sex               <fct> male, female, female, NA, female, male, female, male…
```


### Aesthetic mappings

**aesthetic = column in data**

```r
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm, y = bill_length_mm, color = species)) +
geom_point()
```

- `bill_depth_mm` is mapped to the x coordinate
- `bill_length_mm` is mapped to the y coordinate
- `species` is mapped to the color
Hidden argument names in ggplot

```
ggplot(data = <data>, mapping = aes(x = <x>, y = <y>, <other mappings>))
```

- No need to write explicitly write out `data =`, `mapping =`, `x =`, and `y =` each time in `ggplot`.
- `ggplot` code in the wild often omit these argument names.
- But position needs to be correct if argument names are not specified!
- If no layer is specified, then the plot is `geom_blank()`.
Each layer has a

- **geom** - the geometric object to use display the data,
- **stat** - statistical transformations to use on the data,
- **data** and **mapping** which is usually inherited from ggplot object,

Further specifications are provided by **position adjustment**, **show.legend** and so on.
The `<layer>` is usually created by a function preceded by `geom_` in its name.

```
ggplot(penguins, aes(bill_depth_mm, bill_length_mm)) + geom_point()
```

is a shorthand for

```
ggplot(penguins, aes(bill_depth_mm, bill_length_mm)) +
layer(geom = "point",
     stat = "identity", position = "identity",
     params = list(na.rm = FALSE))
```
Different geometric objects

```r
p <- ggplot(penguins, aes(species, bill_length_mm))
p + geom_violin()
p + geom_boxplot()
p + geom_point()
```
<table>
<thead>
<tr>
<th>geom</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>geom_abline, geom_hline, geom_vline</td>
<td>Reference lines: horizontal, vertical, and diagonal</td>
</tr>
<tr>
<td>geom_bar, geom_col</td>
<td>Bar charts</td>
</tr>
<tr>
<td>geom_bin_2d, geom_bin2d</td>
<td>Heatmap of 2d bin counts</td>
</tr>
<tr>
<td>geom_blank</td>
<td>Draw nothing</td>
</tr>
<tr>
<td>geom_boxplot</td>
<td>A box and whiskers plot (in the style of Tukey)</td>
</tr>
<tr>
<td>geom_contour, geom_contour_filled</td>
<td>2D contours of a 3D surface</td>
</tr>
<tr>
<td>geom_count</td>
<td>Count overlapping points</td>
</tr>
<tr>
<td>geom_density</td>
<td>Smoothed density estimates</td>
</tr>
<tr>
<td>geom_density_2d, geom_density2d, geom_density_2d_filled, geom_density2d_filled</td>
<td>Contours of a 2D density estimate</td>
</tr>
<tr>
<td>geom_dotplot</td>
<td>Dot plot</td>
</tr>
</tbody>
</table>
Statistical transformation

```r
g <- ggplot(penguins, aes(species, bill_length_mm)) + geom_boxplot()
```

- The y-axis is not the raw data!
- It is plotting a statistical transformation of the y-values.
- Under the hood, data is transformed (including x factor input to numerical values).

```r
layer_data(g, 1)
```

```
##   ymin lower middle  upper ymax outliers notchupper notchlower x
## 1 32.1 36.75  38.80 40.750 46.039.31431 38.28569 1
## 2 40.9 46.35  49.55 51.075 58.050.45532 48.64468 2
## 3 40.9 45.30  47.30 49.550 55.959.647.90547 46.69453 3
```

```r
1 4 / 2 2
```
Statistical transformation: stat_bin

- For geom_histogram, default is stat = "bin".
- For stat_bin, default is geom = "bar".
- Every geom has a stat and vice versa.

```r
p <- ggplot(penguins, aes(bill_length_mm))
p + geom_histogram()
p + stat_bin(geom = "bar")
p + stat_bin(geom = "line")
```
Using statistical transformations

To map an aesthetic to computed statistical variable (say called \texttt{var}), you can refer to it by either \texttt{stat(var)} or \texttt{..var..}.

\begin{verbatim}
stat = "bin"

# x  count  density
## 1 32.24138  1 0.003083466
## 2 33.18966  2 0.006166932
## 3 34.13793  6 0.018500797
## 4 35.08621  7 0.021584264
## 5 36.03448 18 0.055502392
## 6 36.98276 13 0.040085061
## 7 37.93103 22 0.067836257
## 8 38.87931 18 0.055502392
## 9 39.82759 19 0.058585859
##10 40.77586 21 0.064752791
##11 41.72414 13 0.040085061
\end{verbatim}

\begin{verbatim}
p + geom_histogram(aes(y = stat(density)))
p + geom_histogram(aes(y = ..density..))
\end{verbatim}
<table>
<thead>
<tr>
<th>stat</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_count</td>
<td>Bar charts</td>
</tr>
<tr>
<td>stat_bin_2d, stat_bin2d</td>
<td>Heatmap of 2d bin counts</td>
</tr>
<tr>
<td>stat_boxplot</td>
<td>A box and whiskers plot (in the style of Tukey)</td>
</tr>
<tr>
<td>stat_contour, stat_contour_filled</td>
<td>2D contours of a 3D surface</td>
</tr>
<tr>
<td>stat_sum</td>
<td>Count overlapping points</td>
</tr>
<tr>
<td>stat_density</td>
<td>Smoothed density estimates</td>
</tr>
<tr>
<td>stat_density_2d, stat_density2d, stat_density_2d_filled</td>
<td>Contours of a 2D density estimate</td>
</tr>
<tr>
<td>stat_function</td>
<td>Draw a function as a continuous curve</td>
</tr>
<tr>
<td>stat_bin_hex, stat_binhex</td>
<td>Hexagonal heatmap of 2d bin counts</td>
</tr>
<tr>
<td>stat_bin</td>
<td>Histograms and frequency polygons</td>
</tr>
</tbody>
</table>
The difference between a **stacked barplot** and a **pie chart** is that the coordinate system have been transformed from **Cartesian coordinate** to **polar coordinate**.

How do we create this barplot in ggplot?

```r
df
## # A tibble: 3 × 2
##   duty   perc
##   <chr> <dbl>
## 1 Teaching    40
## 2 Research    40
## 3 Admin       20
barplot(df$perc, names.arg = df$duty)
```

```r
ggplot(data = df, aes(x = duty, y = perc)) + geom_col()
```
What graph will this yield?

df2

## A tibble: 6 × 3
## duty   perc  type  
##<chr>   <dbl> <chr>  
## Teaching 40 standard
## Research 40 standard
## Admin  20 standard
## Teaching 80 teaching
## Research 0 teaching
## Admin  20 teaching

g <- ggplot(df2, 
aes(x = type, 
y = perc, 
fill = duty)) + 
geom_col()

g + coord_polar("y")

g + coord_polar("x")
Open part1-exercise-01.Rmd
devtools::session_info()

## ─ Session info

##  hash: leg: medium-dark skin tone, person in bed: light skin tone, thumbs down: medium skin tone

##  setting value

## version  R version 4.1.2 (2021-11-01)
## os      macOS Big Sur 10.16
## system  x86_64, darwin17.0
## ui      X11
## language (EN)
## collate en_AU.UTF-8
## ctype   en_AU.UTF-8
## tz      Australia/Melbourne
## date    2021-11-30
## pandoc 2.11.4.2 (Applications/RStudio.app/Contents/MacOS/pandoc/ (via rmarkdown))