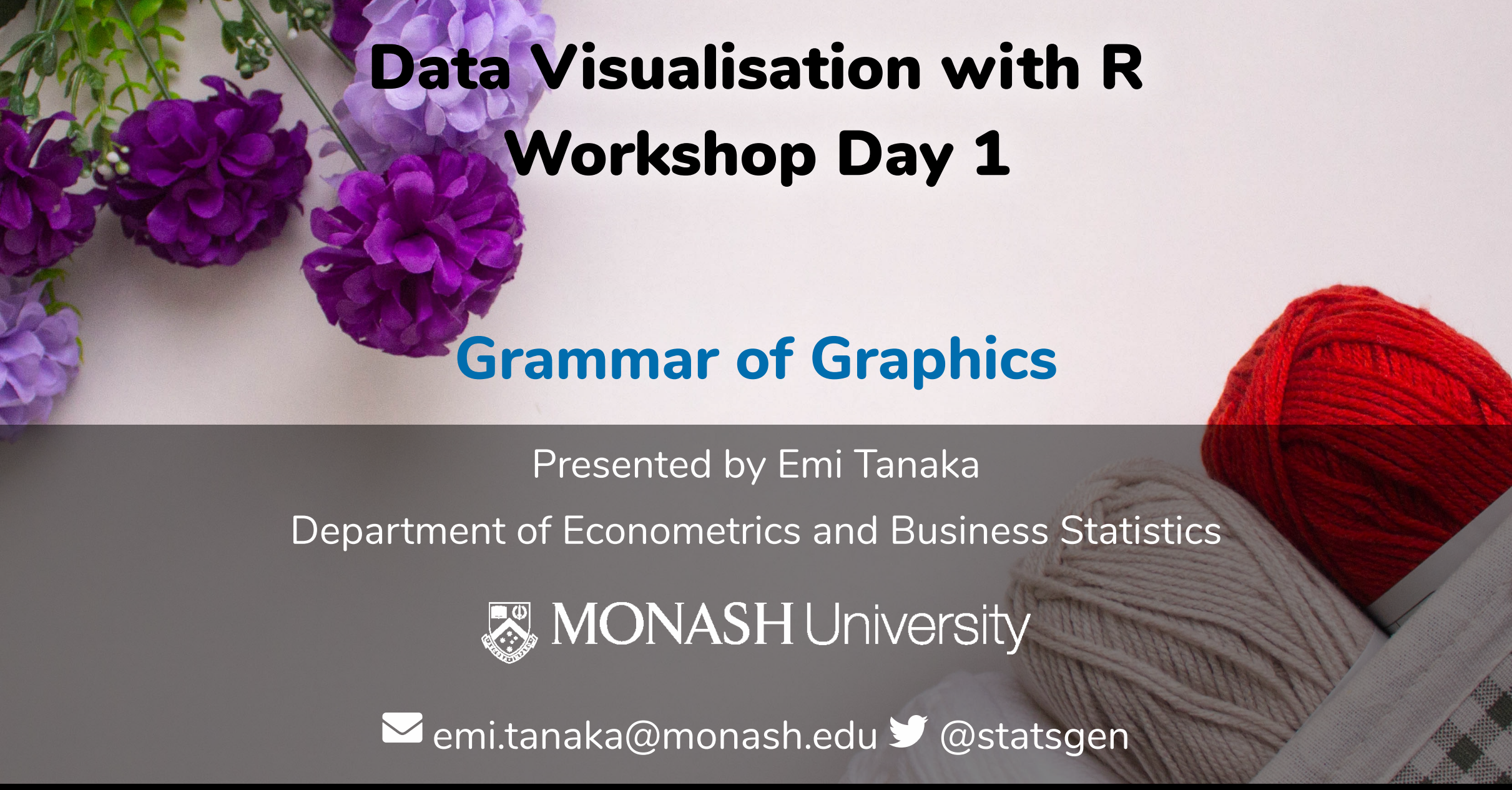


Data Visualisation with R
Workshop Day 1

Grammar of Graphics

Presented by Emi Tanaka

Department of Econometrics and Business Statistics

MONASH University

emi.tanaka@monash.edu @statsgen

28th July 2020 @ Statistical Society of Australia | Online

🔑 Why grammar of graphics for data visualisation?

Constructing plots with R: base version

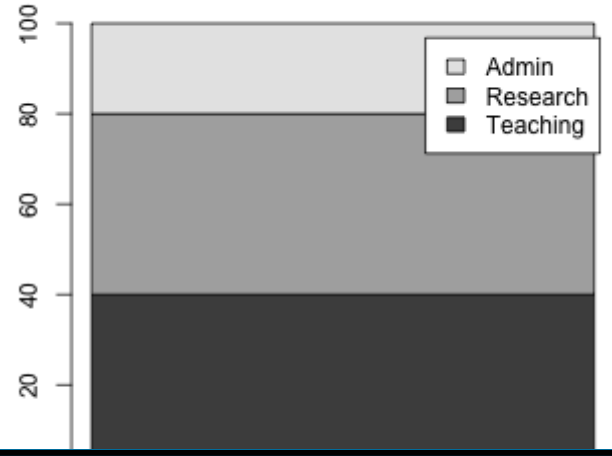
```
df
```

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

▸ **ggplot2** version

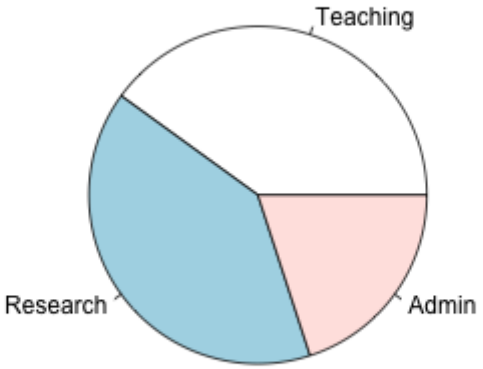
Stacked barplot

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



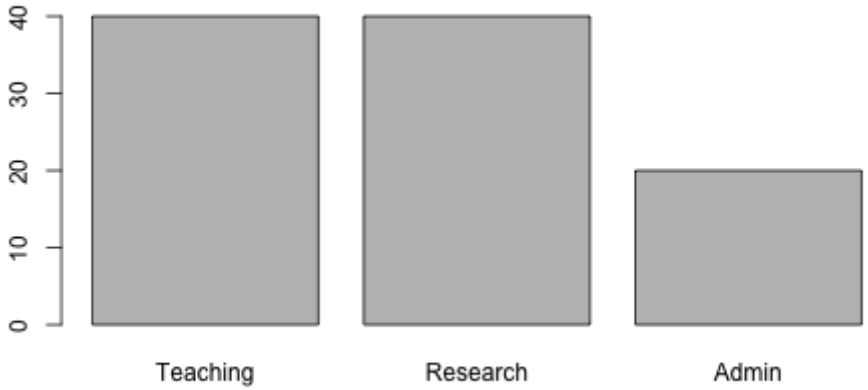
Pie chart

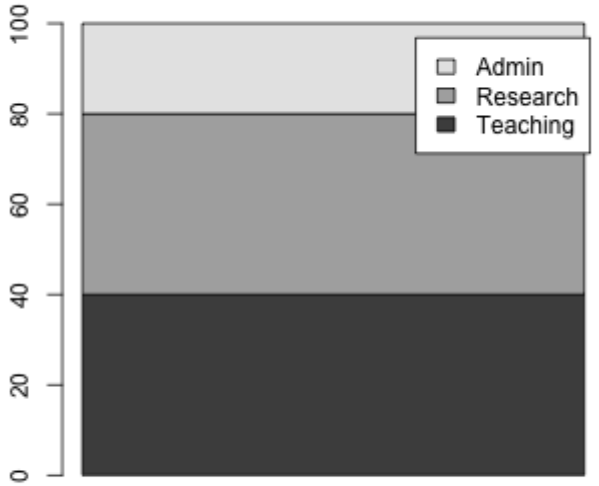
```
pie(df$perc, labels = df$duty)
```



- **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**? 😒

1

Grammar of graphics

Wickham (2016) ggplot2: Elegant Graphics for Data Analysis. *Springer-Verlag New York*

Basic structure of ggplot

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

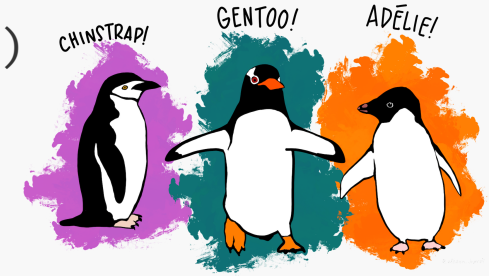
1. **data** as `data.frame` (or `tibble`),
2. a set of **aesthetic** mappings between variables in the data and visual properties, and
3. at least one **layer** which describes how to render each observation.

demo data for today

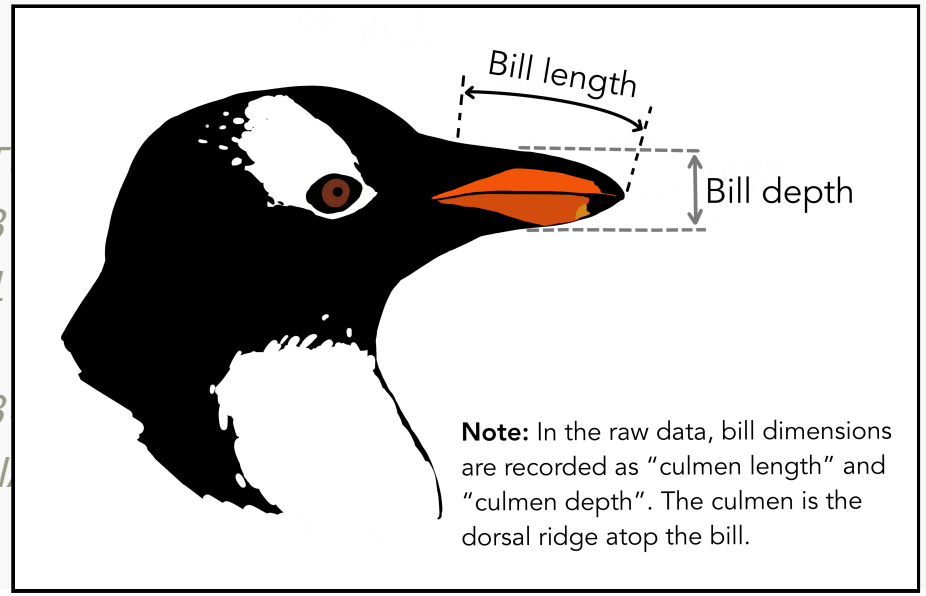
Palmer penguins

penguins data is from the palmerpenguins 📦

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




```
## Rows: 344
## Columns: 8
## $ species           <fct> Adelie, Adelie, Adelie,
## $ island            <fct> Torgersen, Torgersen, T
## $ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 3
## $ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 1
## $ flipper_length_mm <int> 181, 186, 195, NA, 193,
## $ body_mass_g       <int> 3750, 3800, 3250, NA, 3
## $ sex               <fct> male, female, female, N
## $ year              <int> 2007, 2007, 2007, 2007,
```



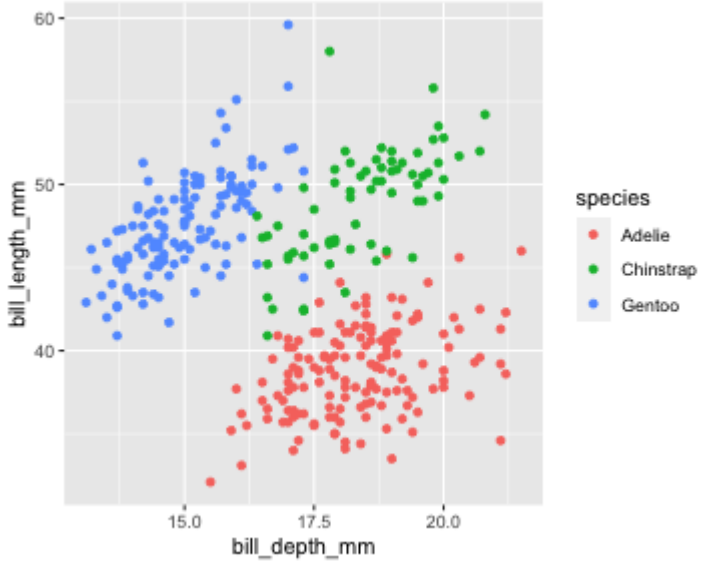
Horst AM, Hill AP, Gorman KB (2020). palmerpenguins: Palmer Archipelago (Antarctica) penguin data. R package version 0.1.0. https://allisonhorst.github.io/palmerpenguins/

Gorman KB, Williams TD, Fraser WR (2014). Ecological sexual dimorphism and environmental variability within a community of Antarctic penguins (genus Pygoscelis). PLoS ONE 9(3):e90081.

Aesthetic mappings

aesthestic = column in data

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




```
aes(x = bill_depth_mm, y = bill_length_mm, color = species)
```

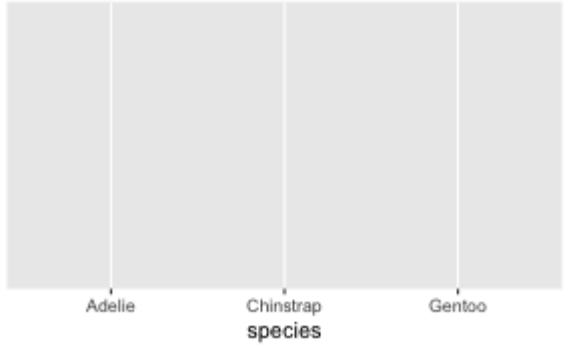
year	sex	body_mass_g	flipper_length_mm	bill_depth_mm	bill_length_mm	island	species

- `bill_depth_mm` is mapped to the x coordinate
- `bill_length_mm` is mapped to the y coordinate
- `species` is mapped to the `color`

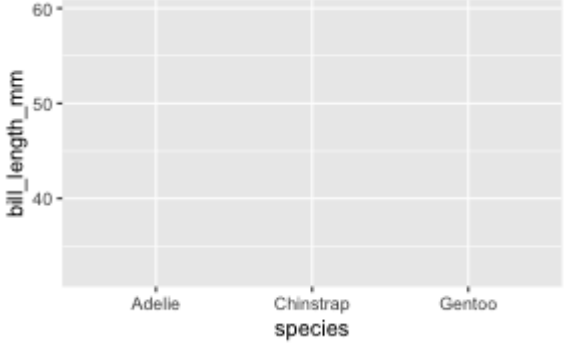
Hidden argument names in ggplot

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

```
ggplot(penguins, aes(species))
```




```
ggplot(penguins, aes(species, bill_length_mm))
```



- 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()`.

Layer

Each layer has a

- `geom` - the geometric object to use display the data,
- `stat` - statisitcal 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.

Example layer: geom_point()

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()
```

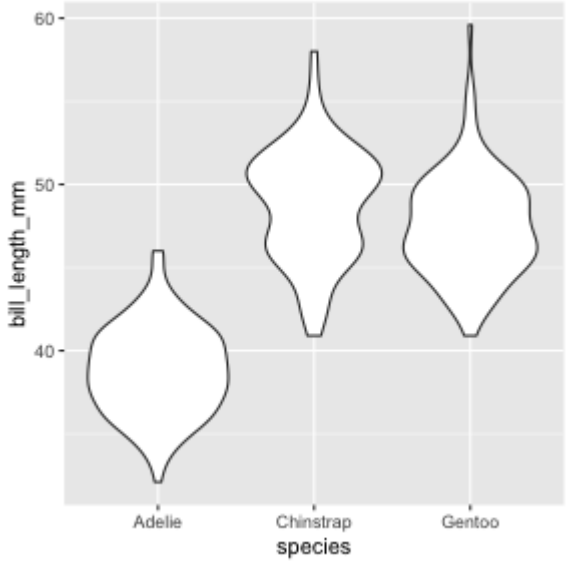
is a shorthand for

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

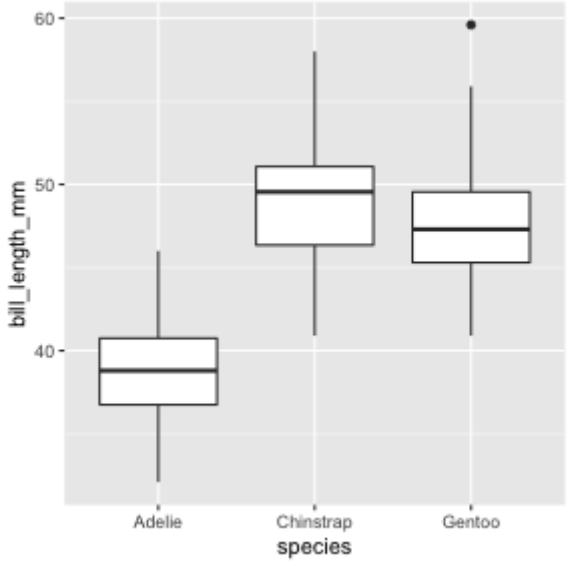
Different geometric objects

```
p <- ggplot(penguins, aes(species, bill_length_mm))
```

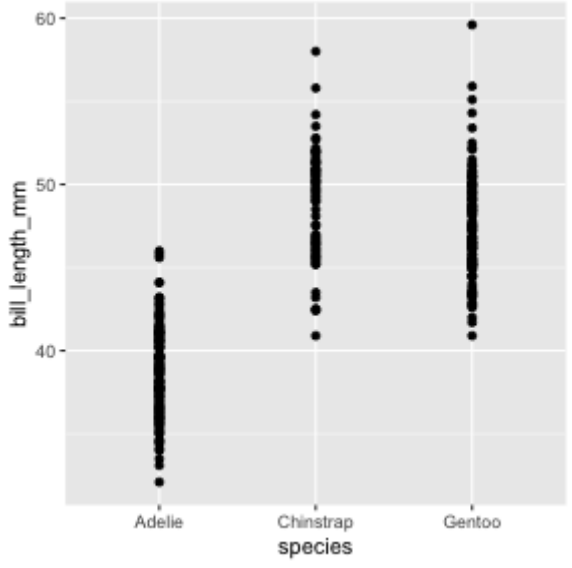
```
p + geom_violin()
```




```
p + geom_boxplot()
```




```
p + geom_point()
```

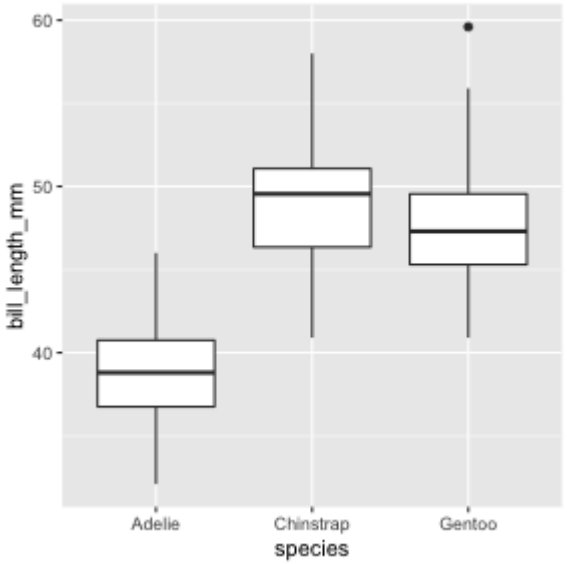


geom

geom	Description
geom_abline, geom_hline, geom_vline	Reference lines: horizontal, vertical, and diagonal
geom_bar, geom_col	Bar charts
geom_bin2d	Heatmap of 2d bin counts
geom_blank	Draw nothing
geom_boxplot	A box and whiskers plot (in the style of Tukey)
geom_contour, geom_contour_filled	2D contours of a 3D surface
geom_count	Count overlapping points
geom_density	Smoothed density estimates
geom_density_2d, geom_density2d, geom_density_2d_filled, geom_density2d_filled	Contours of a 2D density estimate
geom_dotplot	Dot plot

Statistical transformation

```
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).

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

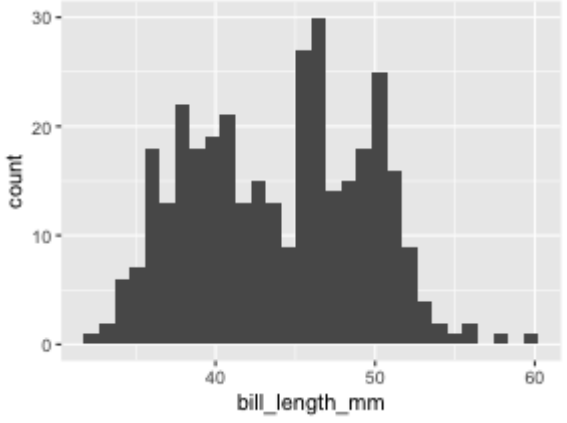
```
##   ymin lower middle  upper ymax outliers notchupper notchlower
## 1 32.1 36.75  38.80 40.750 46.0           39.31431   38.28569
## 2 40.9 46.35  49.55 51.075 58.0           50.45532   48.64468
## 3 40.9 45.30  47.30 49.550 55.9     59.6  47.90547   46.69453
##   PANEL group ymin_final ymax_final  xmin  xmax xid newx new_wi
## 1     1     1       32.1       46.0 0.625 1.375   1    1      0
## 2     1     2       40.9       58.0 1.625 2.375   2    2      0
## 3     1     3       40.9       59.6 2.625 3.375   3    3      0
```

Statistical transformation: stat_bin

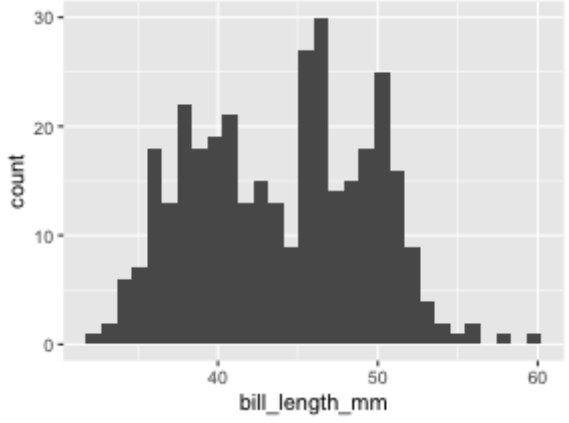
- For `geom_histogram`, default is `stat = "bin"`.
- For `stat_bin`, default is `geom = "bar"`.
- Every `geom` has a `stat` and vice versa.

```
p <- ggplot(penguins, aes(bill_length_mm))
```

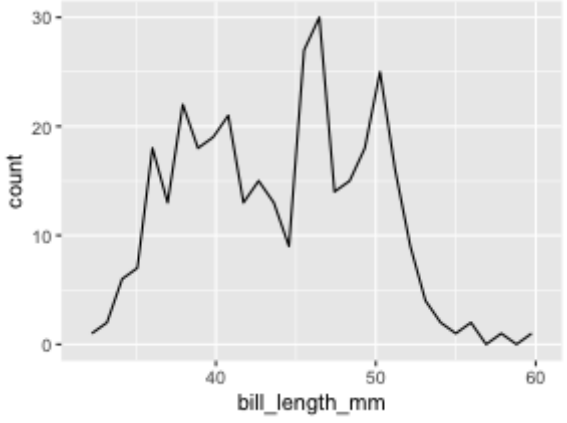
```
p + geom_histogram()
```




```
p + stat_bin(geom = "bar")
```




```
p + stat_bin(geom = "line")
```



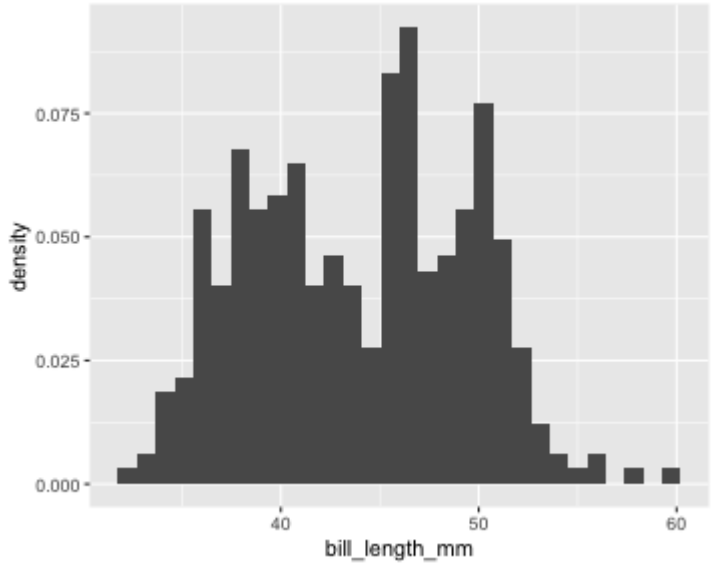
Using statistical transformations

To map an aesthestic to computed statistical variable (say called `var`), you can refer to it by either `stat(var)` or `..var..`.

`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
```

```
p + geom_histogram(aes(y = stat(density) ))
```




```
p + geom_histogram(aes(y = ..density.. ))
```

stat

stat	Description
stat_count	Bar charts
stat_bin_2d, stat_bin2d	Heatmap of 2d bin counts
stat_boxplot	A box and whiskers plot (in the style of Tukey)
stat_contour, stat_contour_filled	2D contours of a 3D surface
stat_sum	Count overlapping points
stat_density	Smoothed density estimates
stat_density_2d, stat_density2d, stat_density_2d_filled	Contours of a 2D density estimate
stat_function	Draw a function as a continuous curve
stat_bin_hex, stat_binhex	Hexagonal heatmap of 2d bin counts
stat_bin	Histograms and frequency polygons

Constructing plots with R: ggplot2 version

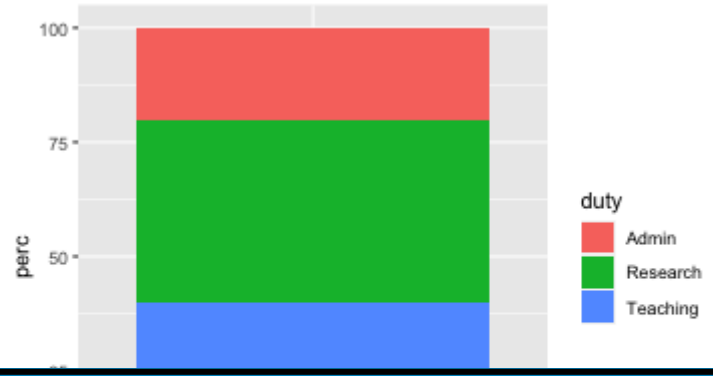
```
df
```

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

▸ **base** version

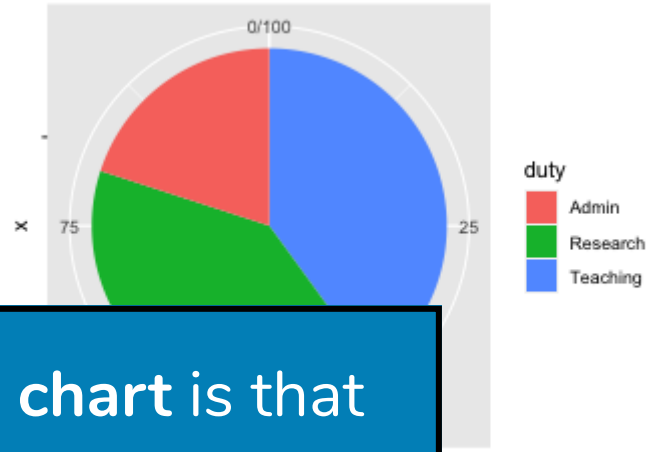
Stacked barplot

```
ggplot(df, aes(x = "", # dummy
               y = perc,
               fill = duty)) +
  geom_col()
```



Pie chart

```
ggplot(df, aes(x = "", # dummy
               y = perc,
               fill = duty)) +
  geom_col() +
  coord_polar(theta = "y")
```



The difference between a **stacked barplot** and a **pie chart** is that the coordinate system have been transformed from **Cartesian coordinate** to **polar coordinate**.

Wickham (2010) A Layered Grammar of Graphics, *Journal of Computational and Graphical Statistics, 19:1, 3-28*

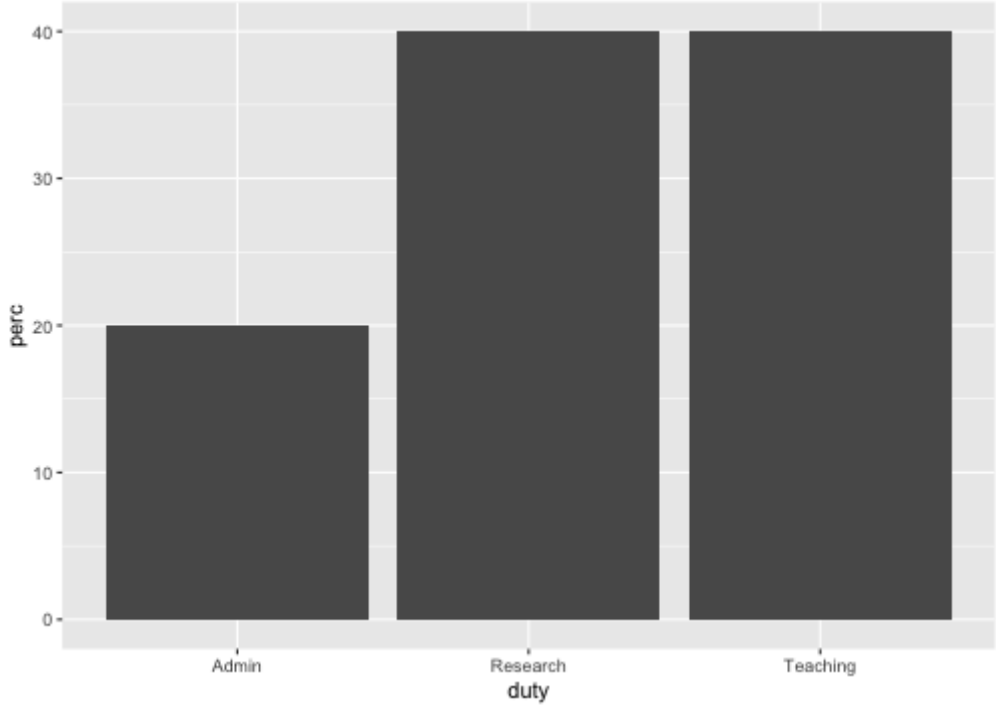
How do we create this barplot in ggplot?

```
df
```

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

```
barplot(df$perc, names.arg = df$duty)
```

```
ggplot(data = df,
       aes(x = duty, y = perc)) +
  geom_col()
```

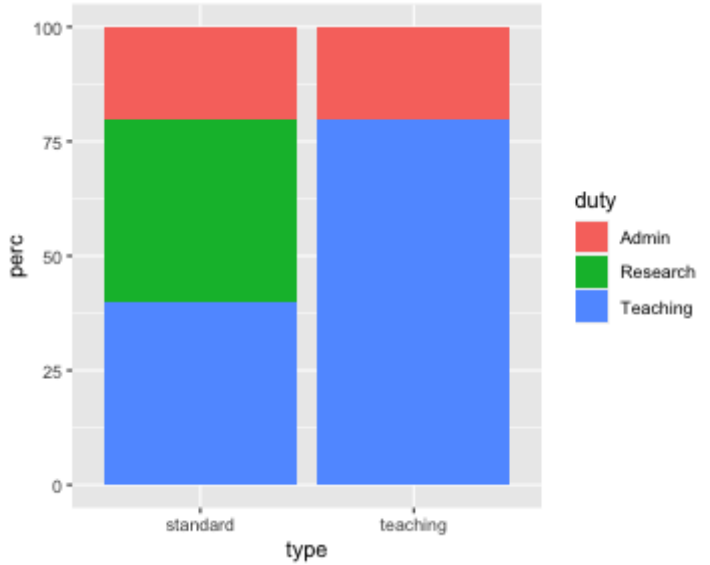


What graph will this yield?

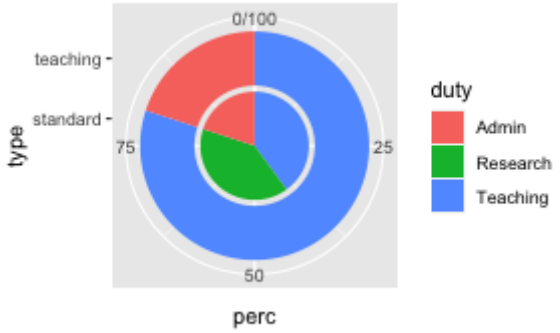
```
df2
```

```
## # A tibble: 6 x 3
##   duty       perc type
##   <chr>     <dbl> <chr>
## 1 Teaching     40 standard
## 2 Research     40 standard
## 3 Admin        20 standard
## 4 Teaching     80 teaching
## 5 Research      0 teaching
## 6 Admin        20 teaching
```

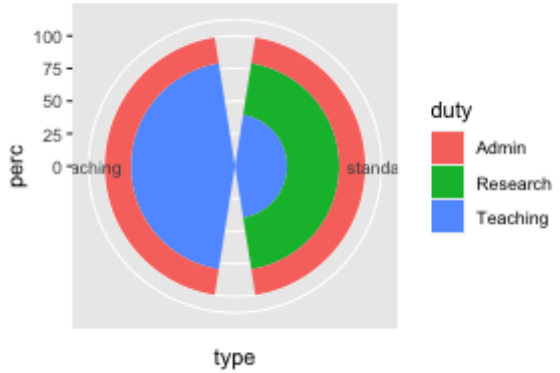
```
g <- ggplot(df2,
            aes(x = type,
                y = perc,
                fill = duty)) +
            geom_col()
g
```




```
g + coord_polar("y")
```




```
g + coord_polar("x")
```



</> Open `exercise-01.Rmd`

15:00

Session Information

```
devtools::session_info()
```

```
## ─ Session info ─────────────────────────────────────────────────────────────
##  setting  value
##  version  R version 4.0.1 (2020-06-06)
##  os       macOS Catalina 10.15.6
##  system   x86_64, darwin17.0
##  ui       X11
##  language (EN)
##  collate  en_AU.UTF-8
##  ctype    en_AU.UTF-8
##  tz       Australia/Melbourne
##  date     2020-07-25
##
## ─ Packages ─────────────────────────────────────────────────────────────────
##  package     * version    date       lib
```