Intro to ggplot2

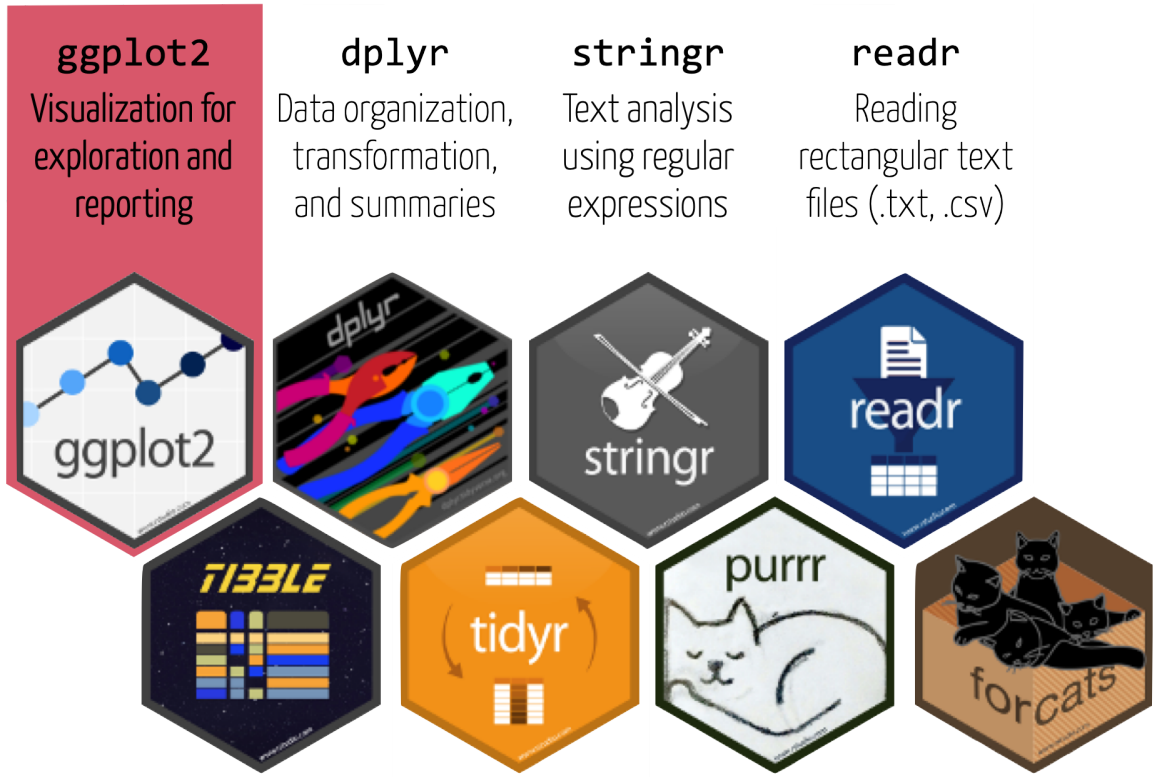
Data Visualization for Social Good

CorrelAid Switzerland

February 2021

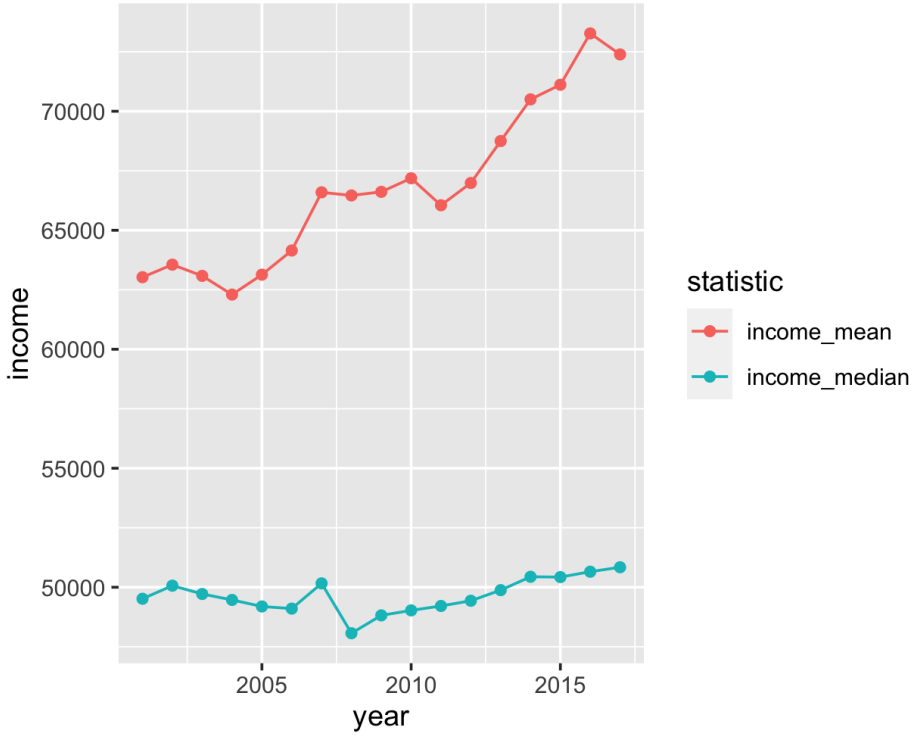
Tidyverse

1. The tidyverse is...
   - A collection of user-friendly **packages** for analyzing **tidy data**
   - An **ecosystem** for analytics and data science with common design principles
   - A **dialect** of the R language



Modular graphics in `ggplot2`

1. **`data`**: the data set
2. **`mapping`**: the plot's structure
   - What do the axes represent?
   - What do size, shapes, colors, etc. represent?
3. **`geoms`**: geometric shapes illustrating data
4. **`labs`**: Plot annotation
5. **`themes`**: Aesthetic details
6. **`facets`**: Stratify plot according to variable
7. **`scales`**: Scaling of dimensions



ggplot()

1. All plots start with `ggplot()`
2. Two arguments
   - `data` | The data set (`tibble`)
   - `mapping` | The plot structure. Defined using `aes()`

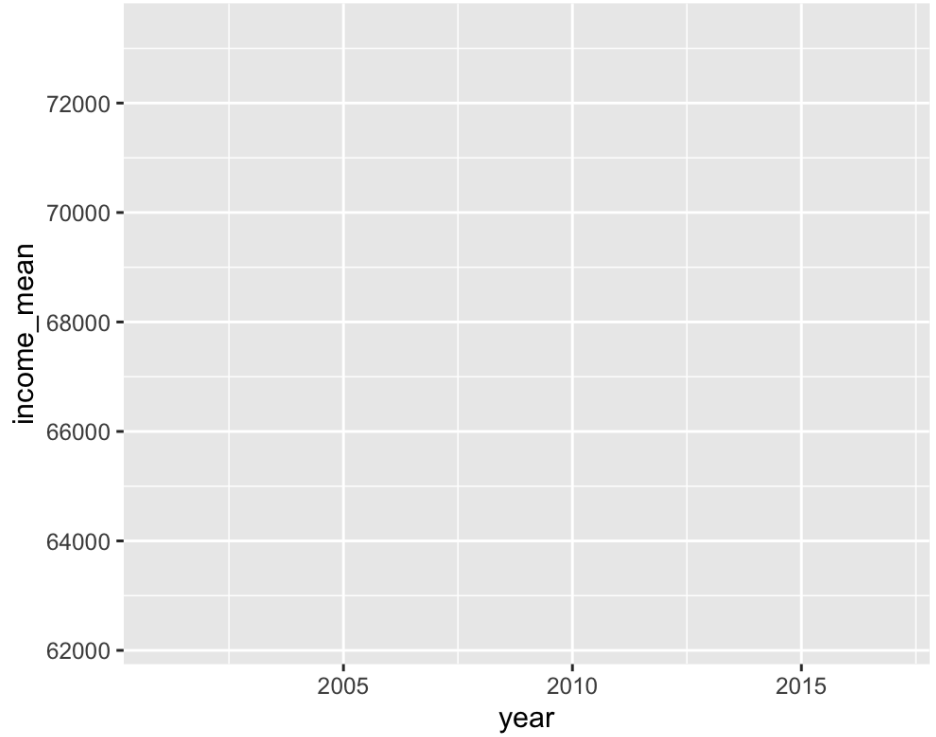
```
# averages per year
basel_avg <- basel %>%
  group_by(year) %>%
  summarize(
    income_mean = mean(income_mean),
    income_median = mean(income_median))
```

```
ggplot(data = basel_avg)
```

aes()

1. `aes()` helps define the structure of the **`mapping`** Argument.
2. Key arguments:
   - `x, y` | Defines axes
   - `color,fill` | Defines colors
   - `alpha` | Defines opacity
   - `size` | Defines sizes
   - `shape` | Defines shapes (e.g., circles or squares)

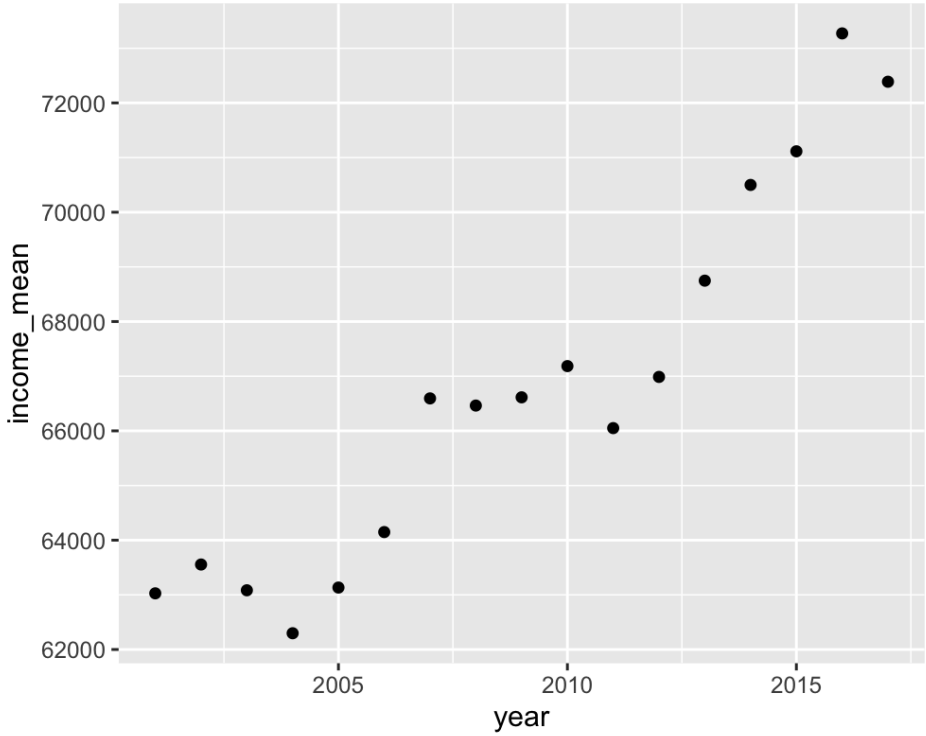
```
ggplot(data = basel_avg,
       mapping = aes(x = year,
                     y = income_mean))
```



+

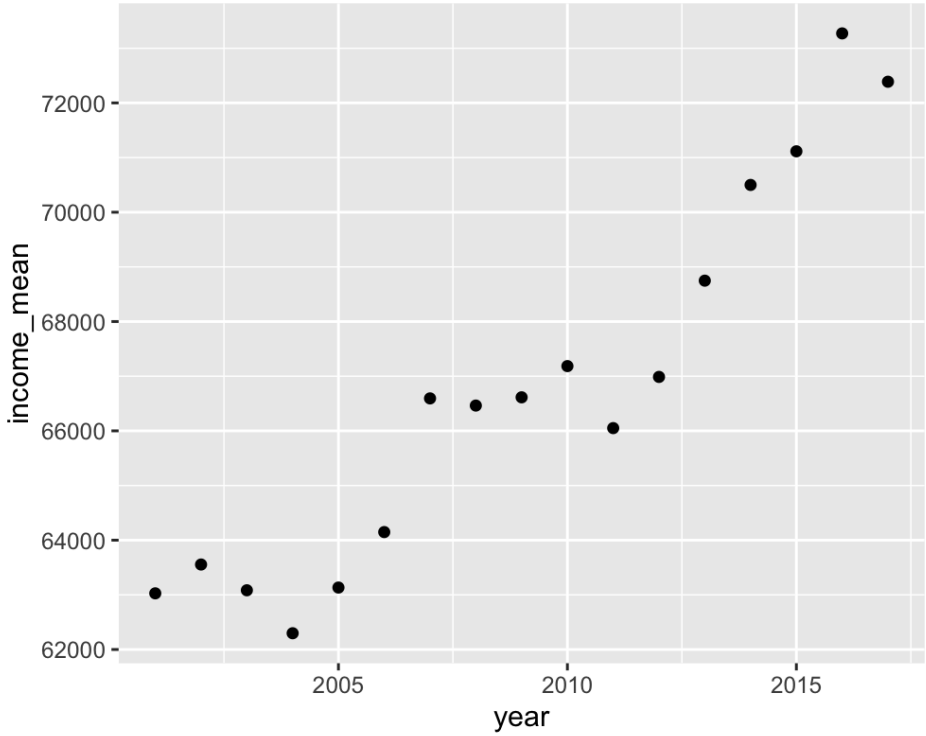
1. The + operator "adds" **additional elements** to the plot.
1. Not to be confused with the pipe `%>%`.

```
ggplot(data = basel_avg,
       mapping = aes(x = year,
                     y = income_mean)) +

  # Show as points
  geom_point()
```



geom_*()

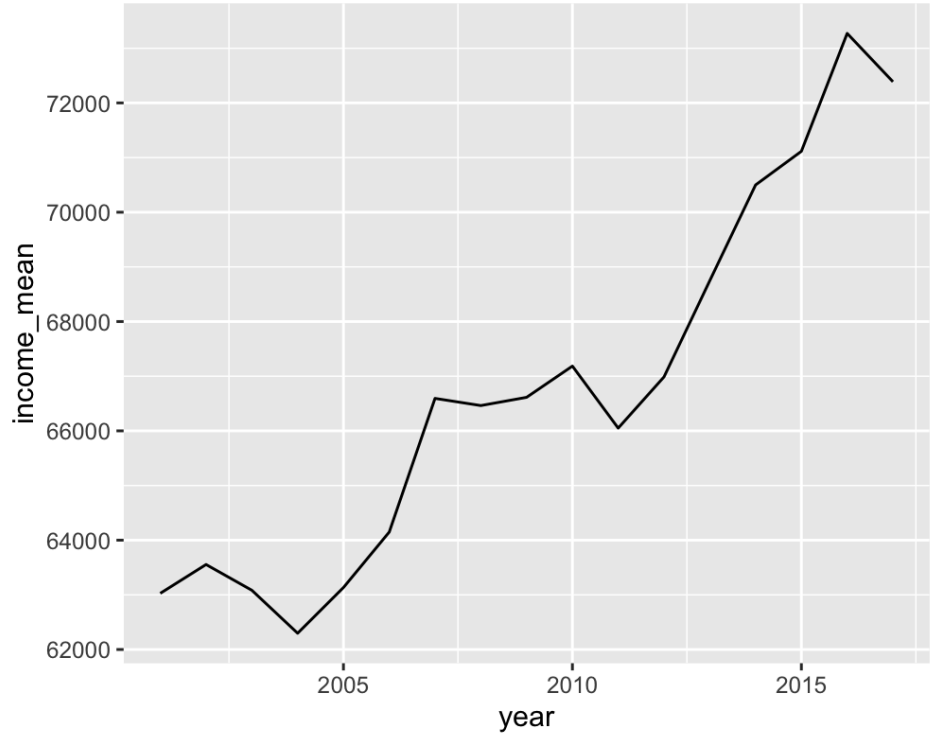
1. `geom_*()` functions define which geometric objects are used to illustrate the data.
2. A few examples `geoms`:
   - `geom_point()` | for points
   - `geom_line()` | for lines
   - `geom_smooth()` | for smooth curves
   - `geom_bar()` | for bars
   - `geom_boxplot()` | for box-plots
   - `geom_violin()` | for violin-plots



geom_*()

1. `geom_*()` functions define which geometric objects are used to illustrate the data.

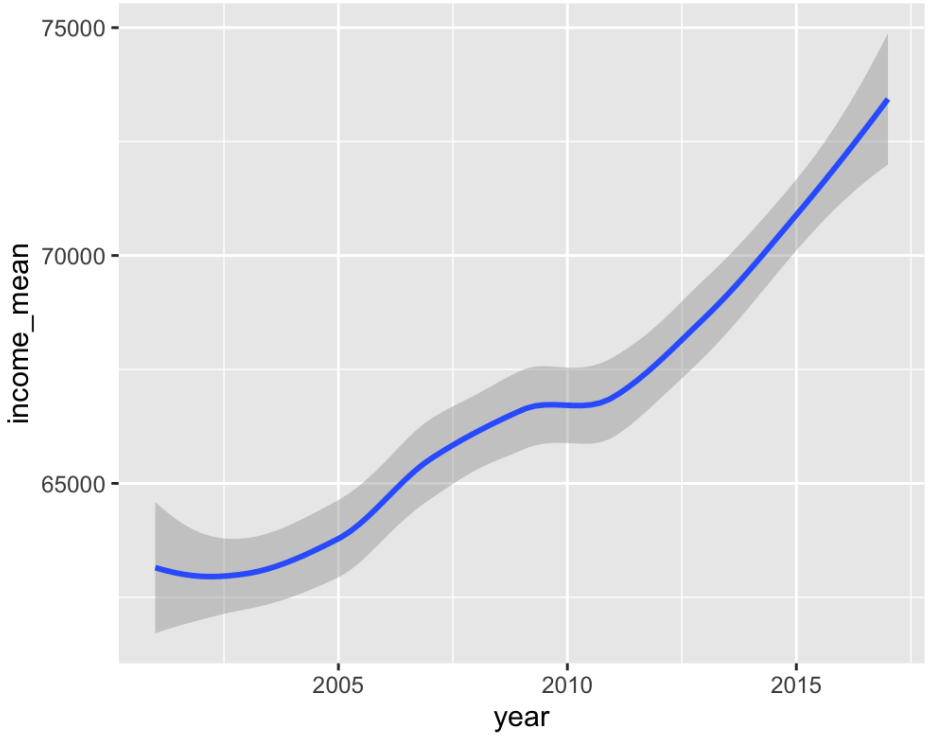
```
ggplot(data = basel_avg,
       mapping = aes(x = year,
                     y = income_mean)) +

  # Show as lines
  geom_line()
```



geom_*()

1. `geom_*()` functions define which geometric objects are used to illustrate the data.

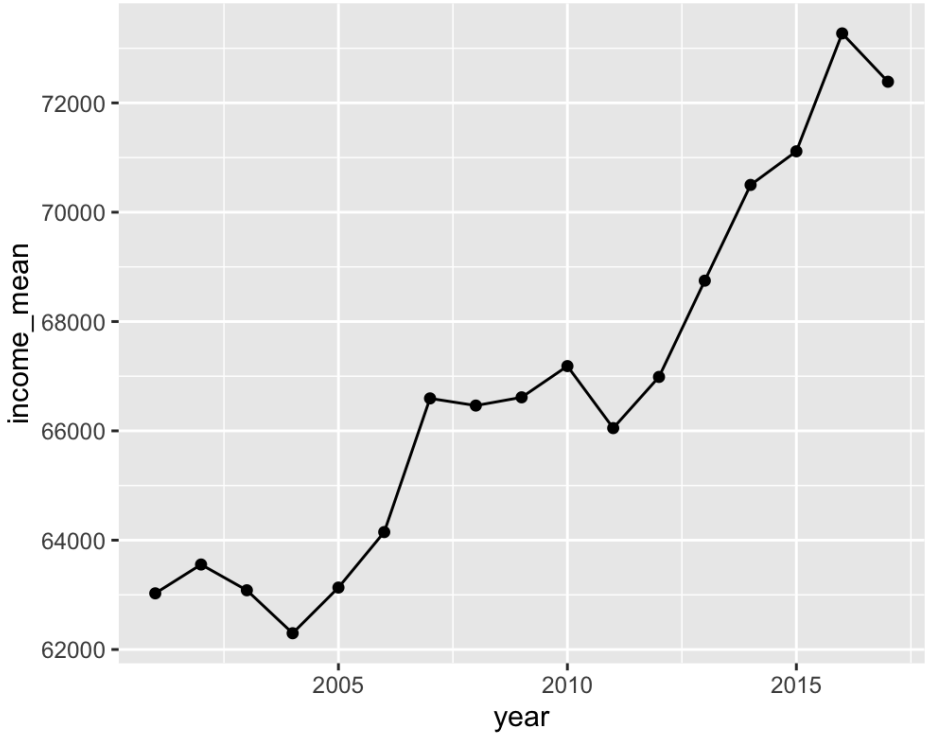
```
ggplot(data = basel_avg,
       mapping = aes(x = year,
                     y = income_mean)) +

  # Show as smoothed curve
  geom_smooth()
```



geom_*()

1. `geom_*()` functions define which geometric objects are used to illustrate the data.

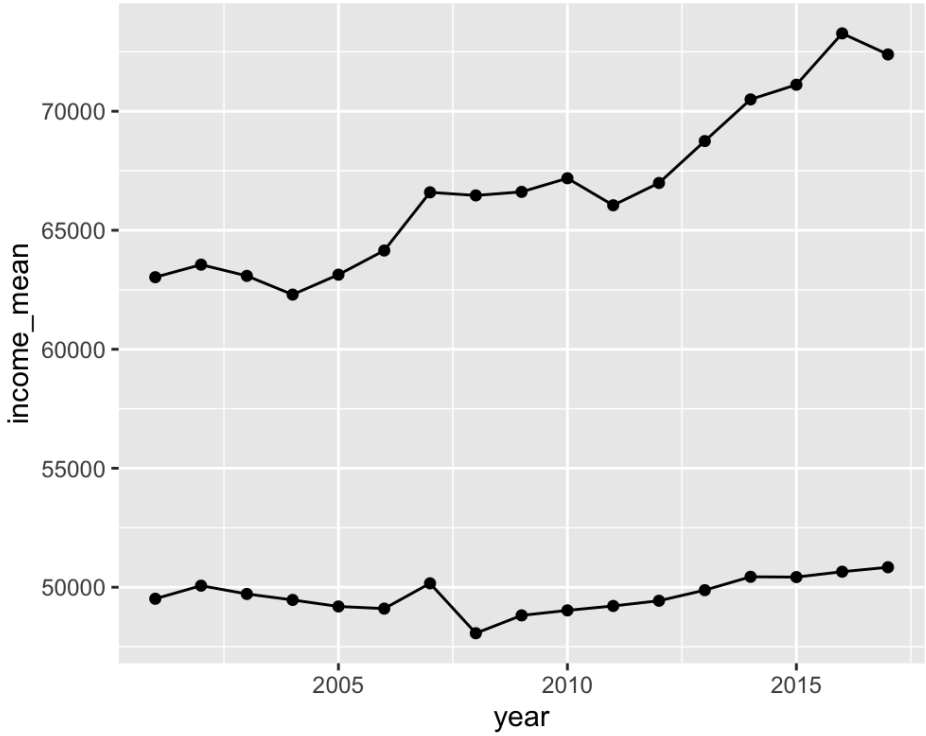
```
ggplot(data = basel_avg,
       mapping = aes(x = year,
                     y = income_mean)) +

  # Show as points and lines
  geom_point() +
  geom_line()
```



geom_*()

1. Most `geom_*()` functions allow specification of **`data`** and **`mapping`**.

```
ggplot(data = basel_avg,
       mapping = aes(x = year,
                     y = income_mean)) +
  geom_point() +
  geom_line() +

  # Add points and lines for median
  geom_point(aes(y = income_median)) +
  geom_line(aes(y = income_median))
```



Wrangling

1. Oftentimes, creating the desired plot requires appropriate data wrangling.
2. `ggplot` works best with **long data formats**.

```
# pivot to long format
basel_avg_long <- basel_avg %>%
  pivot_longer(-year,
               names_to = "statistic",
               values_to = "income")
```

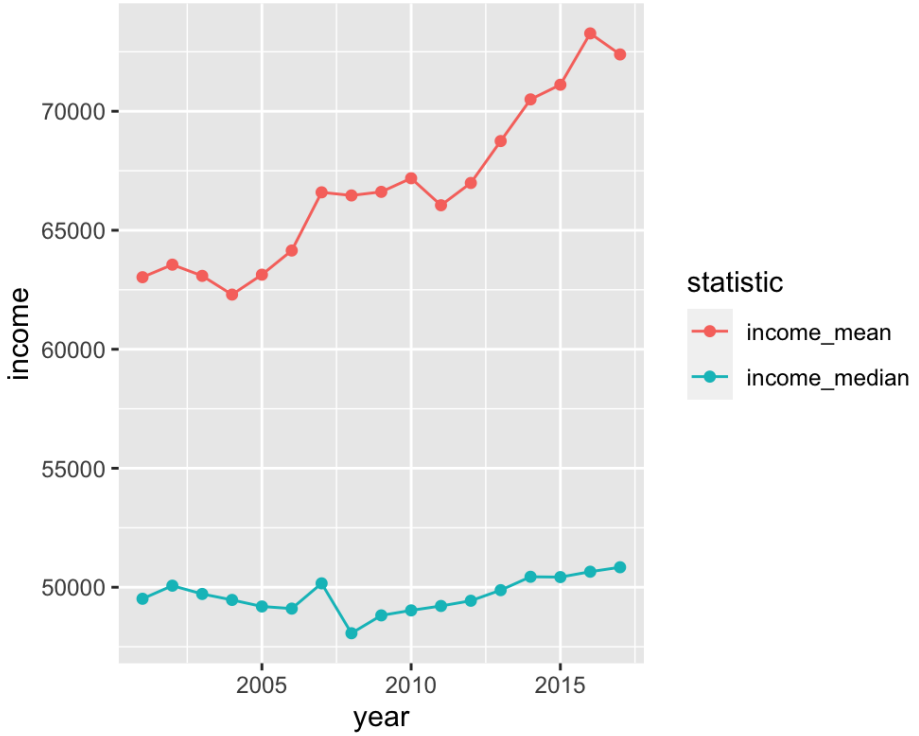
```
basel_avg_long
```

```
# A tibble: 34 x 3
    year statistic     income
   <dbl> <chr>          <dbl>
 1  2001 income_mean   63027.
 2  2001 income_median 49516.
 3  2002 income_mean   63555.
 4  2002 income_median 50066.
 5  2003 income_mean   63083.
 6  2003 income_median 49717.
 7  2004 income_mean   62298.
 8  2004 income_median 49467.
 9  2005 income_mean   63133.
10  2005 income_median 49192.
# … with 24 more rows
```

aes()

1. `aes()` helps define the structure of the **`mapping`** Argument.

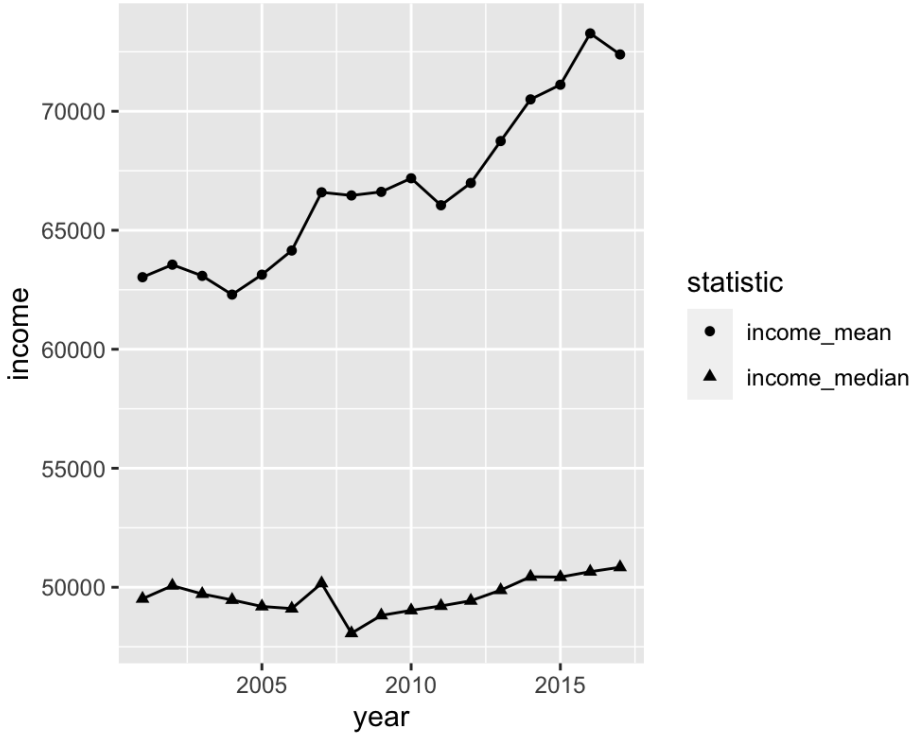
```
# use basel_avg_long
ggplot(data = basel_avg_long,
       mapping = aes(
         x = year,
         y = income,

         # add color dimension
         col = statistic)) +
  geom_point() +
  geom_line()
```



aes()

1. `aes()` helps define the structure of the **`mapping`** Argument.

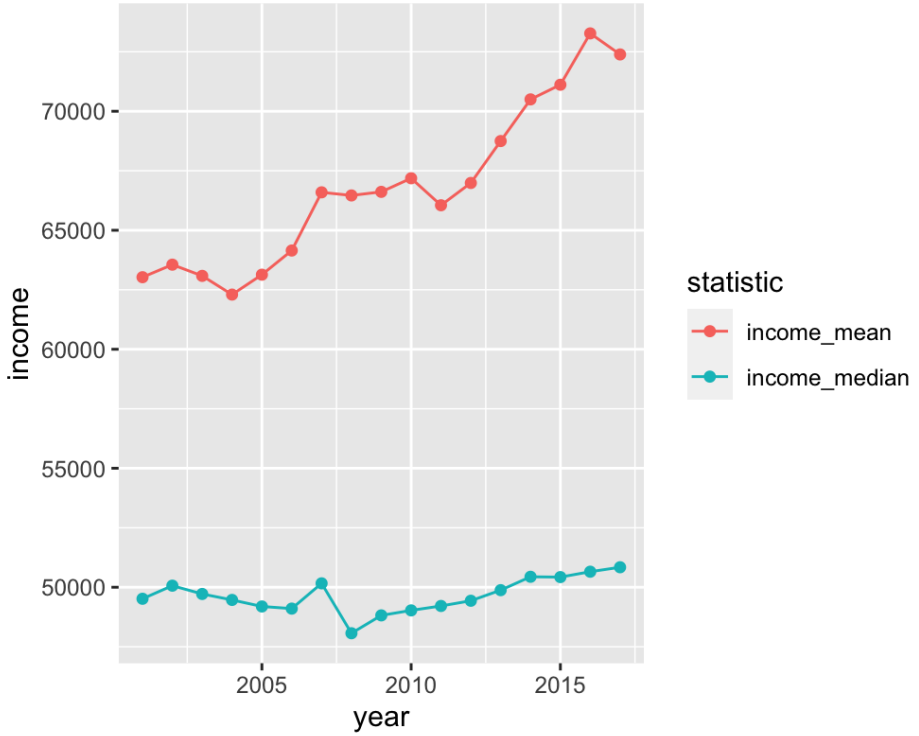
```
# use basel_avg_long
ggplot(data = basel_avg_long,
       mapping = aes(
         x = year,
         y = income,

         # add shape dimension
         shape = statistic)) +
  geom_point() +
  geom_line()
```



facet_*()

1. Facetting creates the **same plot for groups** defined by another variable.
2. Key functions:
   - `facet_wrap()`
   - `facet_grid()`

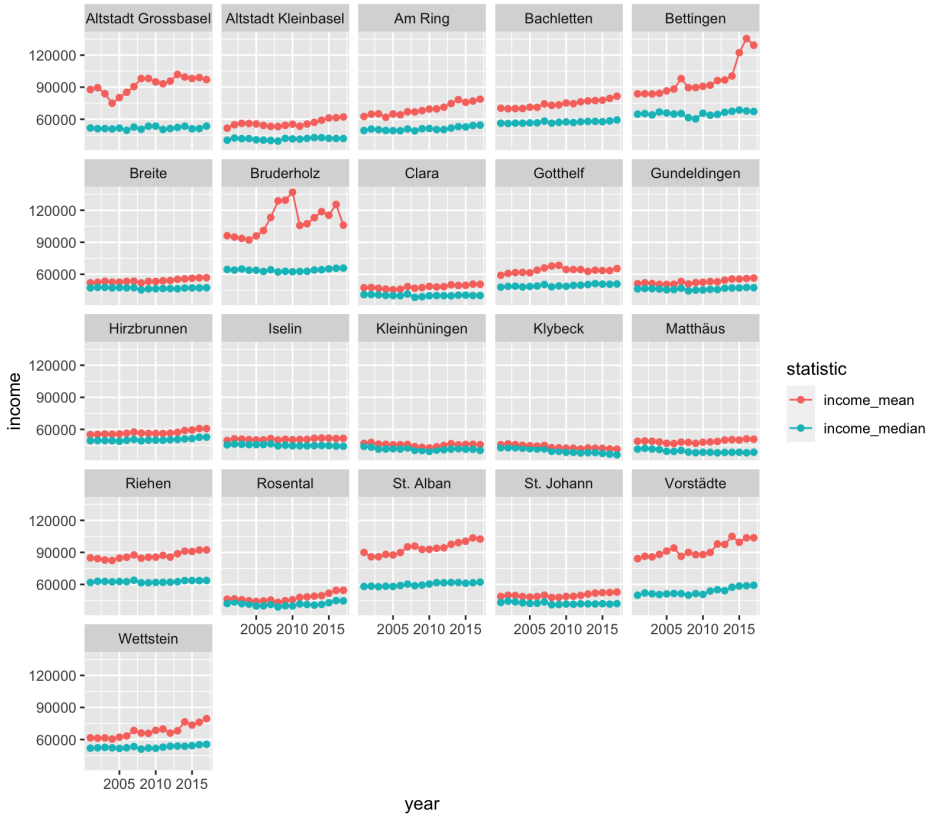
```
basel_long <- basel %>%
  pivot_longer(c(income_mean, income_median
               names_to = 'statistic',
               values_to = 'income')
```



facet_*()

1. Facetting creates the **same plot for groups** defined by another variable.

```
# use basel_long
ggplot(data = basel_long,
       mapping = aes(
         x = year,
         y = income,
         col = statistic)) +
  geom_point() +
  geom_line() +

  # facet by quarter
  facet_wrap(~quarter)
```



Schedule