



# Data Handling: Import, Cleaning and Visualisation

Lecture 11:

Visualisation and Dynamic Documents

Prof. Dr. Ulrich Matter

10/12/2020

Updates

# Week 12

- Thursday, 17 December
  - Wrap up
  - Exam info
  - Feedback
  - Q&A (send questions until tomorrow! [ulrich.matter@unisg.ch](mailto:ulrich.matter@unisg.ch))
- Friday, 18 December
  - Decentral exam for **exchange students**!
  - See Canvas for details on place/time.

# Mock exam

- On Studynet/Canvas today
  - Mock exam
  - Solutions
  - Answersheet
  - Answersheet example

**Data Display**

# Data display

- Formatting data values for publication.
- Typical: String operations to make numbers and text look nicer.
  - Before creating a table or figure...

# Data display

```
# load packages and data
library(tidyverse)
data("swiss")
# compute summary statistics
swiss_summary <-
  summarise(swiss,
    avg_education = mean(Education, na.rm = TRUE),
    avg_fertility = mean(Fertility, na.rm = TRUE),
    N = n()
  )
swiss_summary

##   avg_education avg_fertility  N
## 1      10.97872      70.14255 47
```

Problems?

# Data display: round numeric values

```
swiss_summary_rounded <- round(swiss_summary, 2)  
swiss_summary_rounded
```

```
##      avg_education avg_fertility  N  
## 1             10.98           70.14 47
```



# Data display: detailed formatting of numbers

- Coerce to text.
- String operations.
- Decimal marks, units (e.g., currencies), other special characters for special formats (e.g. coordinates).
- **`format()`**-function

# Data display: `format()` example

```
swiss_form <- format(swiss_summary_rounded,  
                     decimal.mark=",")
```

```
swiss_form
```

```
##      avg_education avg_fertility  N  
## 1             10,98           70,14 47
```

# Data Visualisation with R (`ggplot2`)

# Data visualisation

- Final step of data pipeline/data science procedure!
  - Convincingly communicating insights from data.
- **R** is a very powerful tool to do this!
  - (Very powerful graphics engine)

# Data visualisation in R

Three main approaches:

1. The original `graphics` package ((R Core Team 2018); shipped with the base R installation).

# Data visualisation in R

Three main approaches:

1. The original `graphics` package ((R Core Team 2018); shipped with the base R installation).
2. The `lattice` package (Sarkar 2008), an implementation of the original Bell Labs 'Trellis' system.

# Data visualisation in R

Three main approaches:

1. The original `graphics` package ((R Core Team 2018); shipped with the base R installation).
2. The `lattice` package (Sarkar 2008), an implementation of the original Bell Labs 'Trellis' system.
3. The `ggplot2` package (Wickham 2016), an implementation of Leland Wilkinson's 'Grammar of Graphics'.

ggplot2





## ggplot2 basics

Using `ggplot2` to generate a basic plot in R is quite simple. Basically, it involves three key points:

1. The data must be stored in a `data.frame/tibble` (in tidy format!).

## ggplot2 basics

Using `ggplot2` to generate a basic plot in R is quite simple. Basically, it involves three key points:

1. The data must be stored in a `data.frame/tibble` (in tidy format!).
2. The starting point of a plot is always the function `ggplot()`.

## ggplot2 basics

Using `ggplot2` to generate a basic plot in R is quite simple. Basically, it involves three key points:

1. The data must be stored in a `data.frame/tibble` (in tidy format!).
2. The starting point of a plot is always the function `ggplot()`.
3. The first line of plot code declares the data and the 'aesthetics' (e.g., which variables are mapped to the x-/y-axes):

## ggplot2 basics

Using `ggplot2` to generate a basic plot in R is quite simple. Basically, it involves three key points:

1. The data must be stored in a `data.frame/tibble` (in tidy format!).
2. The starting point of a plot is always the function `ggplot()`.
3. The first line of plot code declares the data and the 'aesthetics' (e.g., which variables are mapped to the x-/y-axes):

```
ggplot(data = my_dataframe, aes(x= xvar, y= yvar))
```

# Example data set: swiss

```
# load the R package  
library(tidyverse) # automatically loads ggplot2  
# load the data  
data(swiss)  
# get details about the data set  
# ?swiss  
# inspect the data  
head(swiss)
```

```
##           Fertility Agriculture Examination Education Catholic Infant.Mortality  
## Courtelary      80.2          17.0           15          12          9.96         22.2  
## Delemont        83.1          45.1            6           9         84.84         22.2  
## Franches-Mnt    92.5          39.7            5           5         93.40         20.2  
## Moutier         85.8          36.5           12           7         33.77         20.3  
## Neuveville      76.9          43.5           17          15          5.16         20.6  
## Porrentruy      76.1          35.3            9           7         90.57         26.6
```

# Add indicator variable

Code a province as 'Catholic' if more than 50% of the inhabitants are catholic:

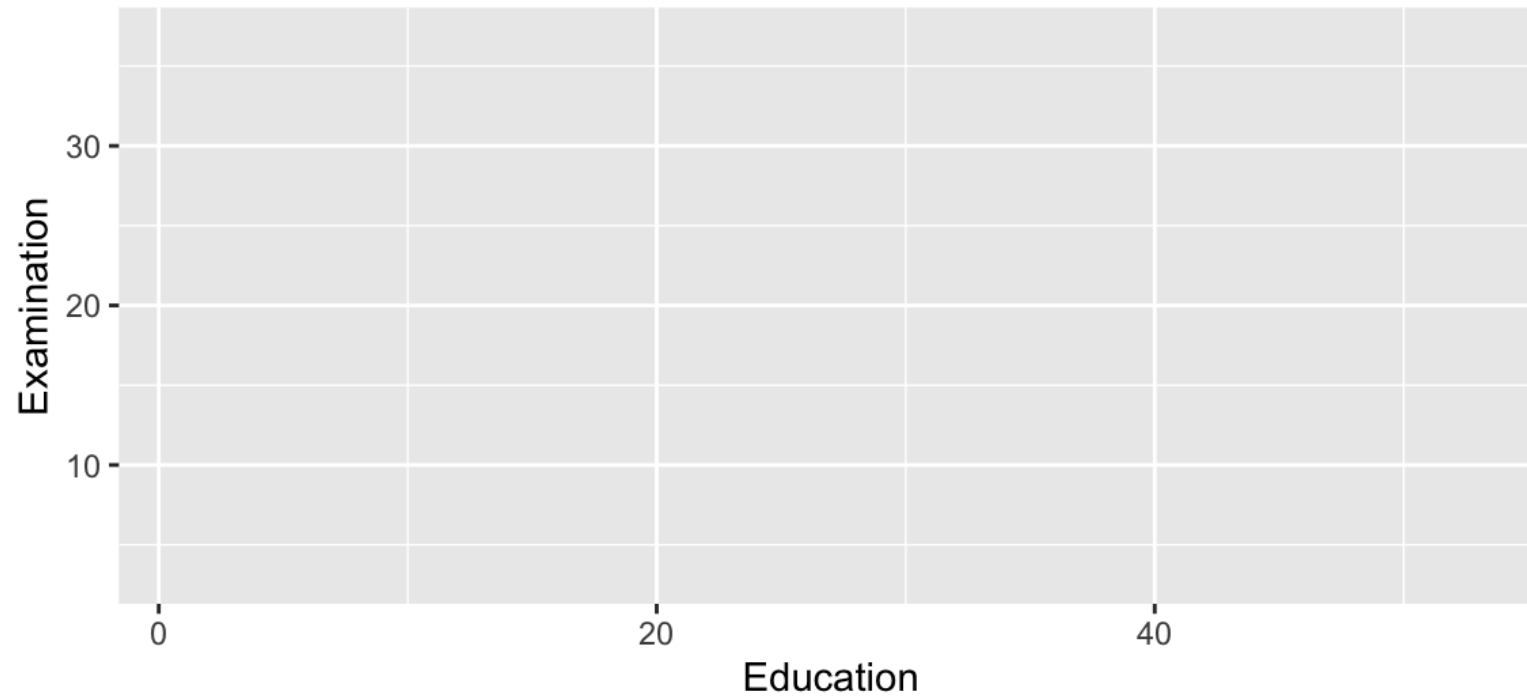
```
# via tidyverse/mutate
swiss <- mutate(swiss,
                Religion =
                  ifelse(50 < Catholic, 'Catholic', 'Protestant'))

# 'old school' alternative
swiss$Religion <- 'Protestant'
swiss$Religion[50 < swiss$Catholic] <- 'Catholic'

# set to factor
swiss$Religion <- as.factor(swiss$Religion)
```

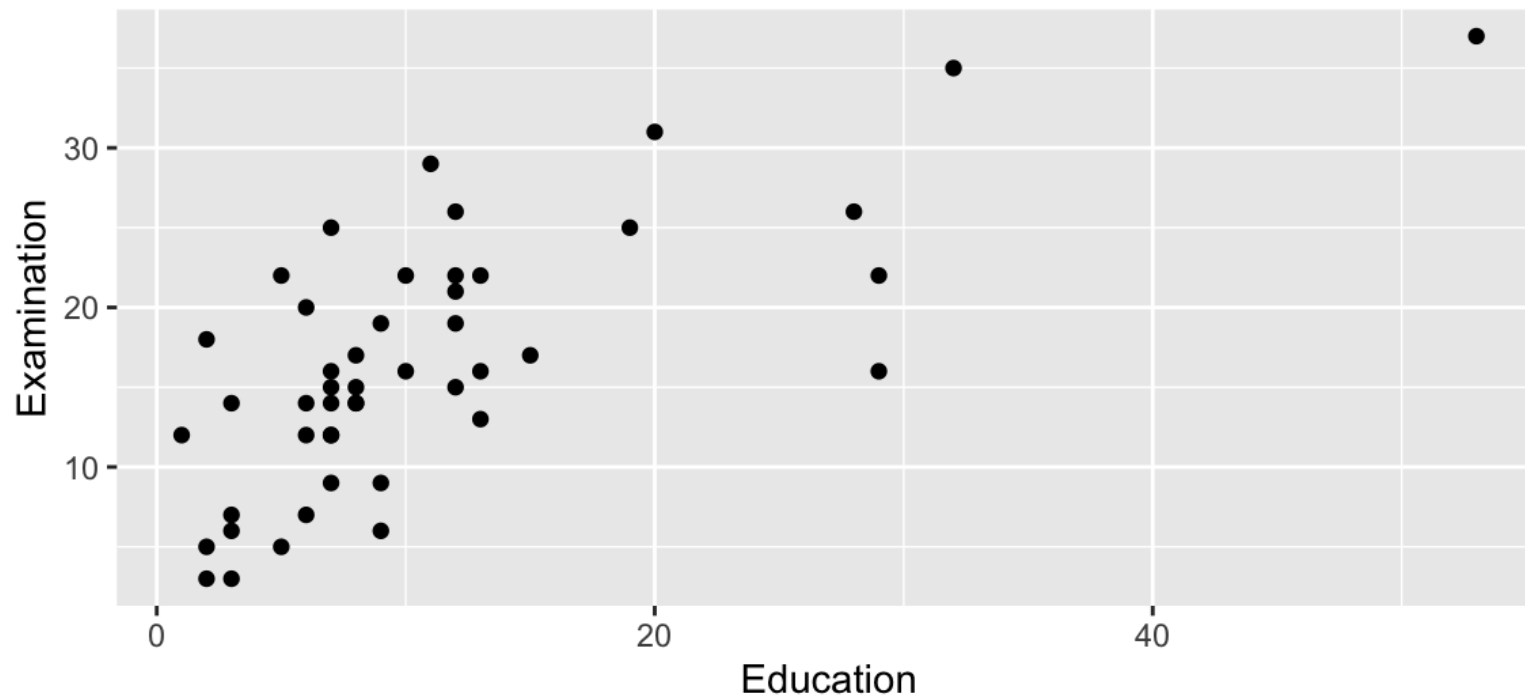
# Data and aesthetics

```
ggplot(data = swiss, aes(x = Education, y = Examination))
```



# Geometries (~the type of plot)

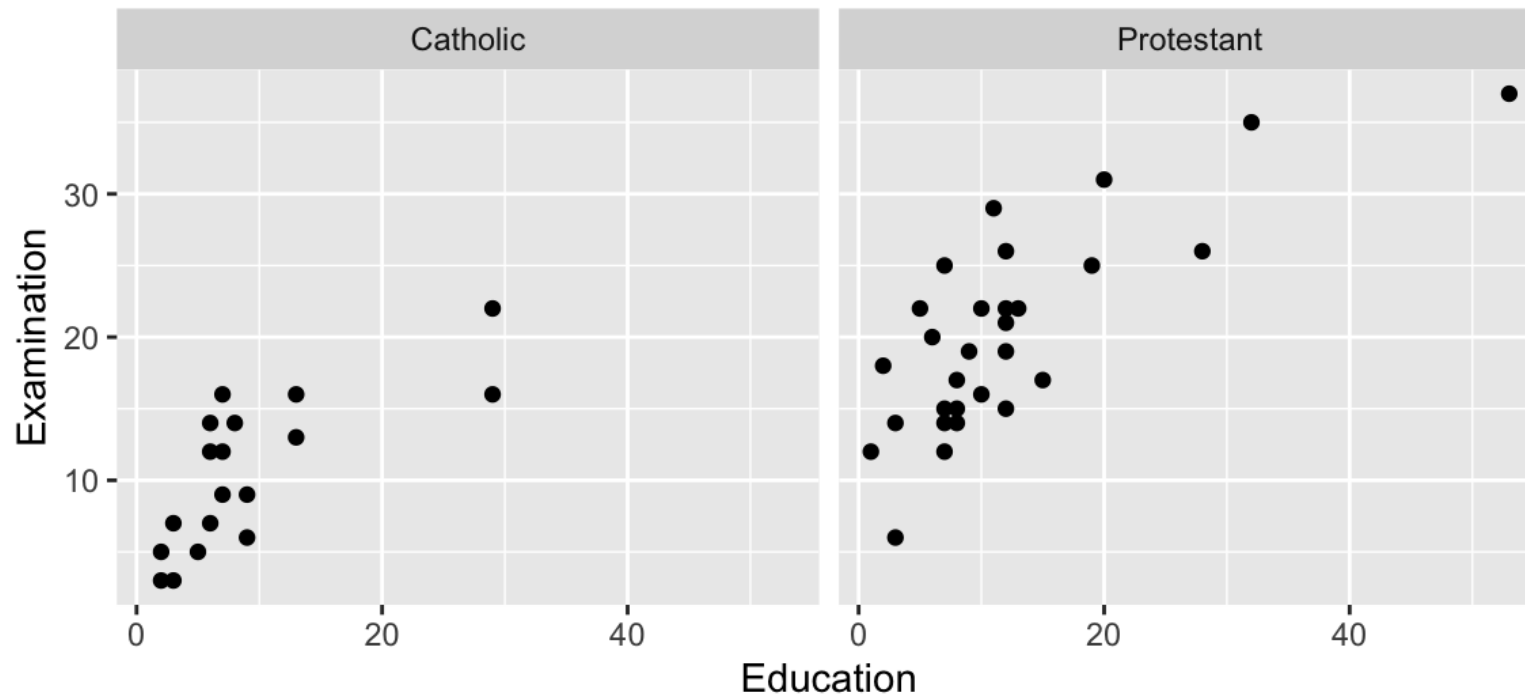
```
ggplot(data = swiss, aes(x = Education, y = Examination)) +  
  geom_point()
```





# Facets

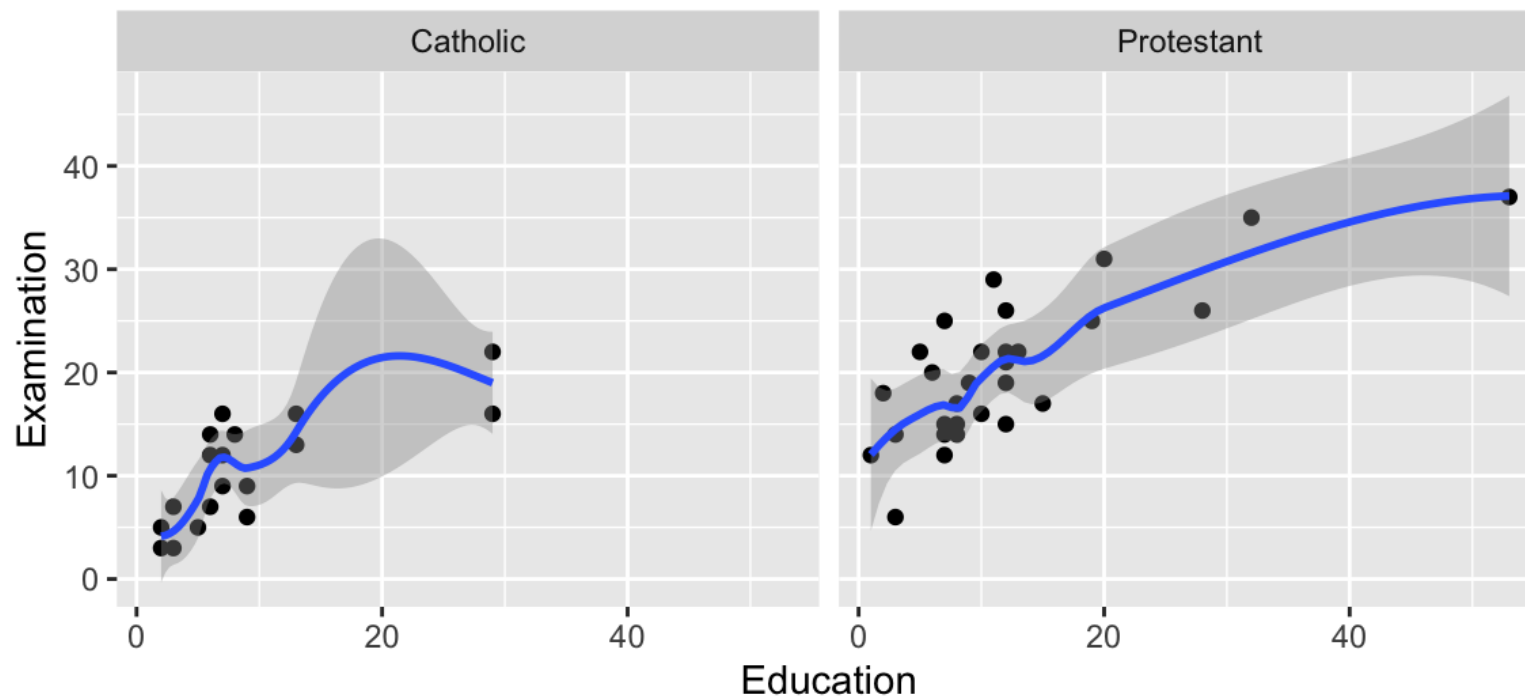
```
ggplot(data = swiss, aes(x = Education, y = Examination)) +  
  geom_point() +  
  facet_wrap(~Religion)
```



# Additional layers and statistics

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +  
  geom_point() +  
  geom_smooth(method = 'loess') +  
  facet_wrap(~Religion)
```

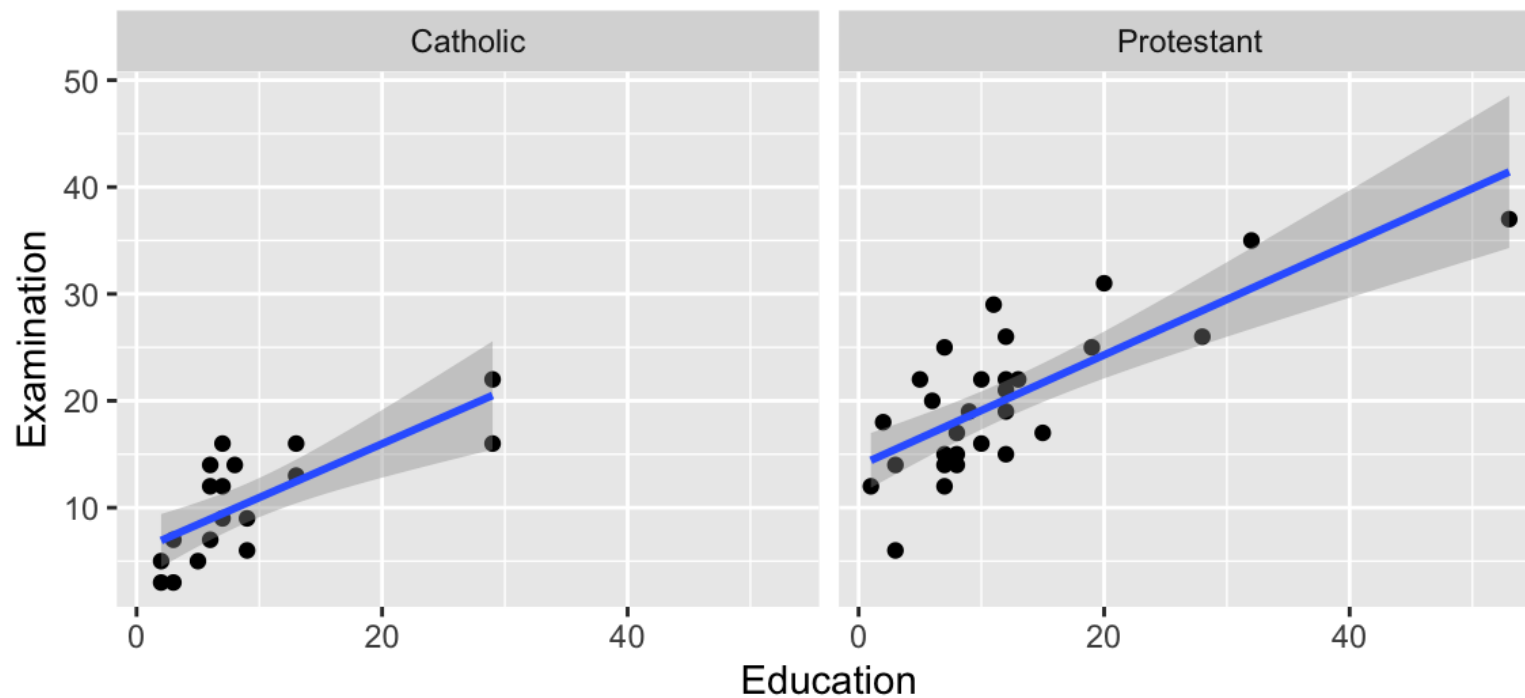
```
## `geom_smooth()` using formula 'y ~ x'
```



# Additional layers and statistics

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +  
  geom_point() +  
  geom_smooth(method = 'lm') +  
  facet_wrap(~Religion)
```

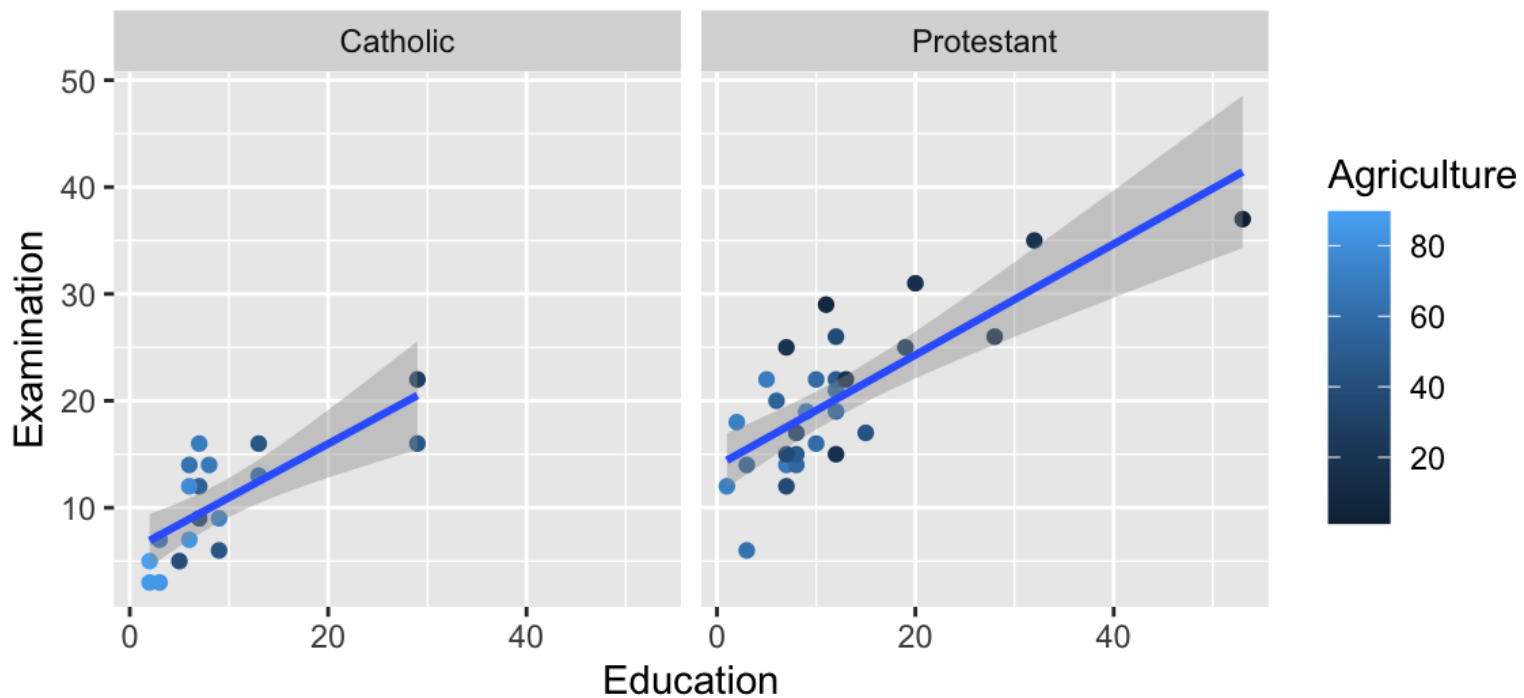
## `geom\_smooth()` using formula 'y ~ x'



# Additional aesthetics

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +  
  geom_point(aes(color = Agriculture)) +  
  geom_smooth(method = 'lm') +  
  facet_wrap(~Religion)
```

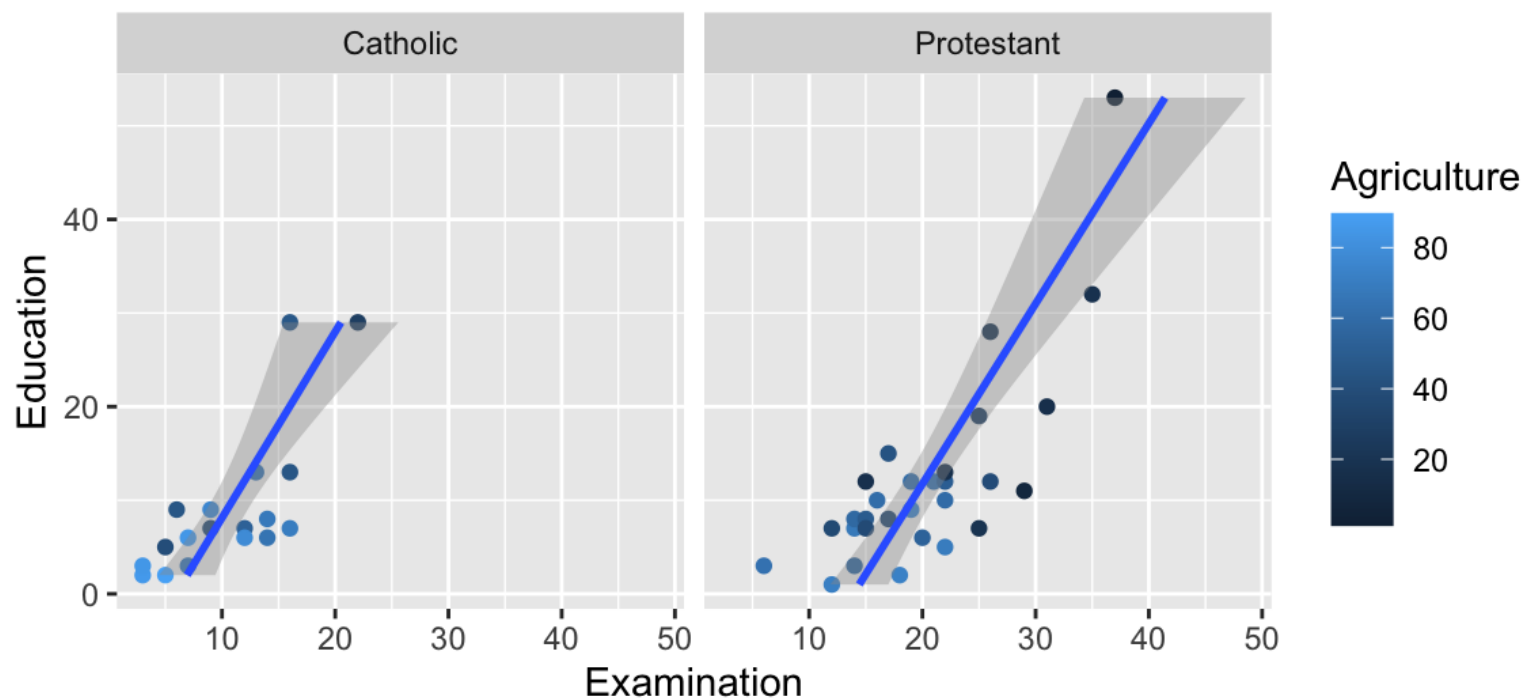
```
## `geom_smooth()` using formula 'y ~ x'
```



# Change coordinates

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +  
  geom_point(aes(color = Agriculture)) +  
  geom_smooth(method = 'lm') +  
  facet_wrap(~Religion) +  
  coord_flip()
```

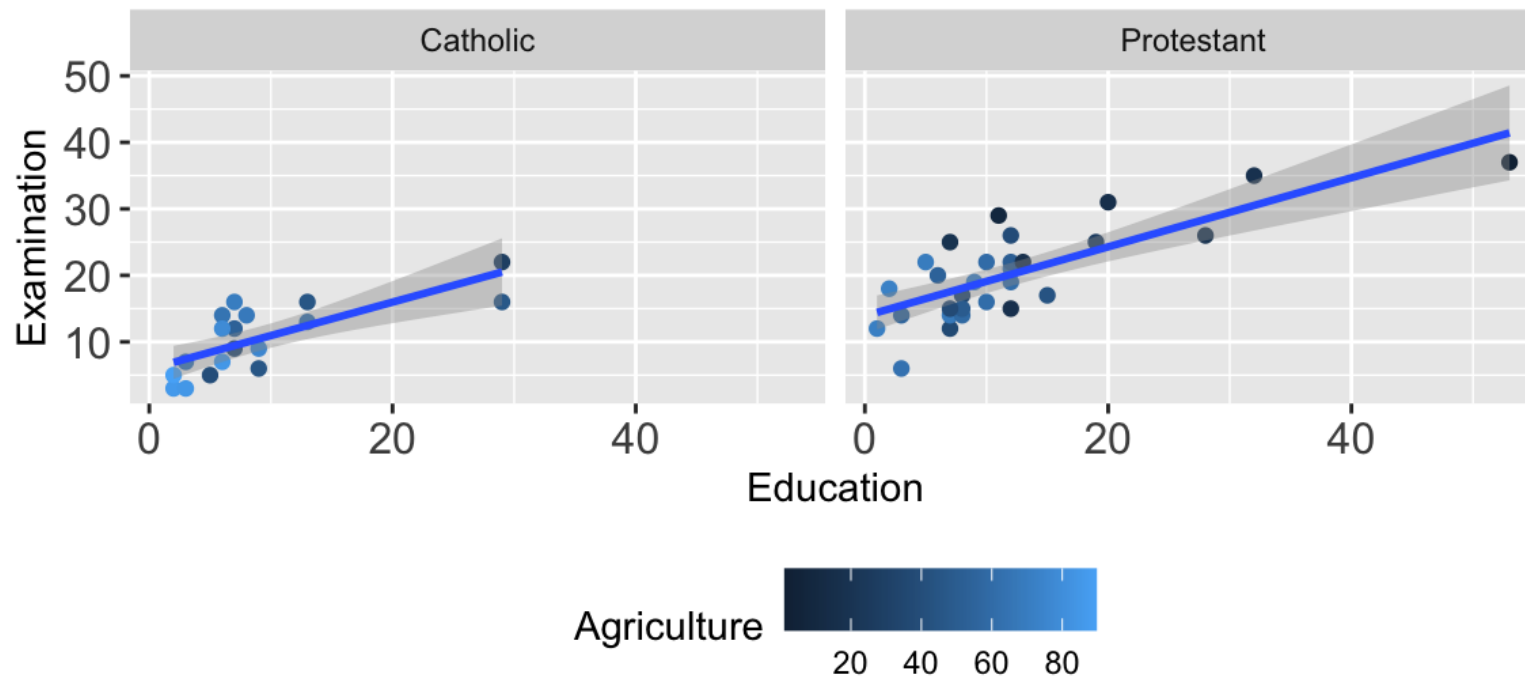
```
## `geom_smooth()` using formula 'y ~ x'
```



# Themes

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +  
  geom_point(aes(color = Agriculture)) +  
  geom_smooth(method = 'lm') +  
  facet_wrap(~Religion) +  
  theme(legend.position = "bottom", axis.text=element_text(size=12) )
```

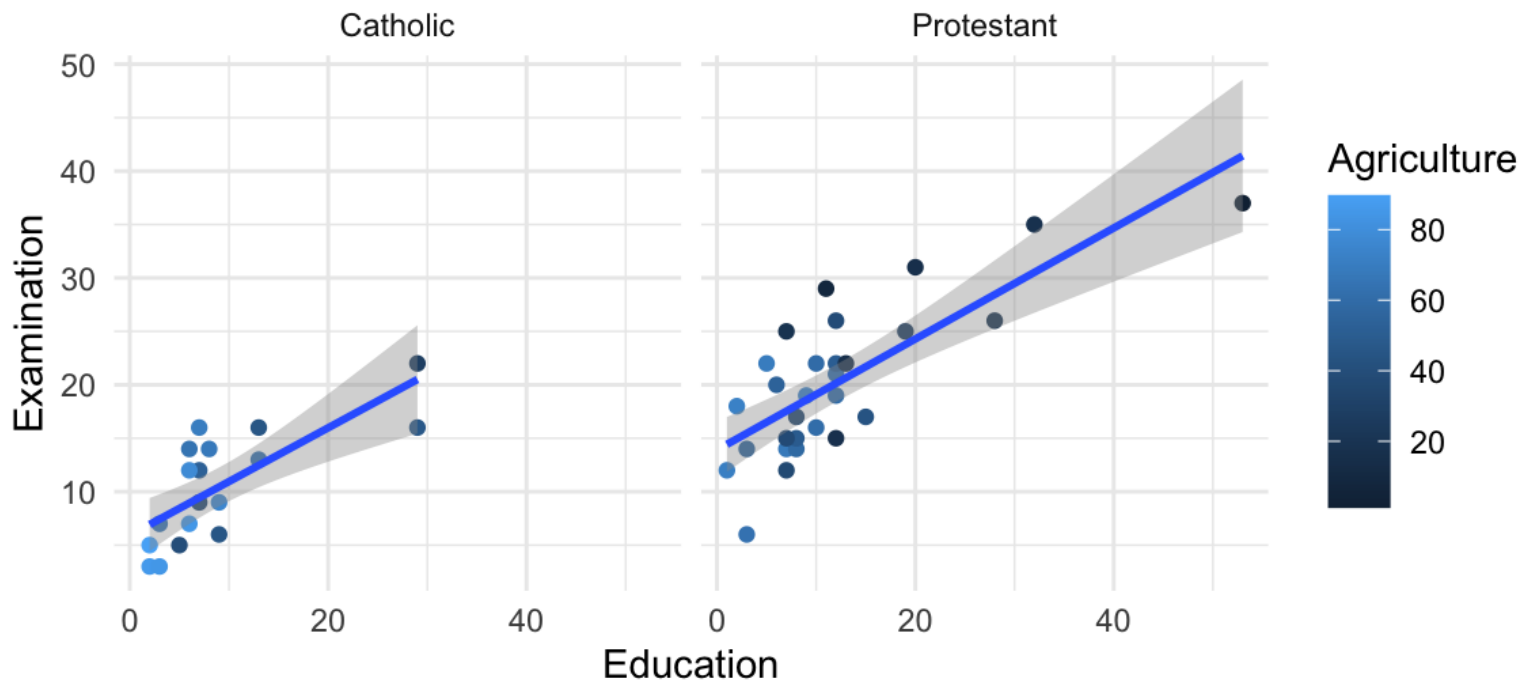
```
## `geom_smooth()` using formula 'y ~ x'
```



# Themes

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +  
  geom_point(aes(color = Agriculture)) +  
  geom_smooth(method = 'lm') +  
  facet_wrap(~Religion) +  
  theme_minimal()
```

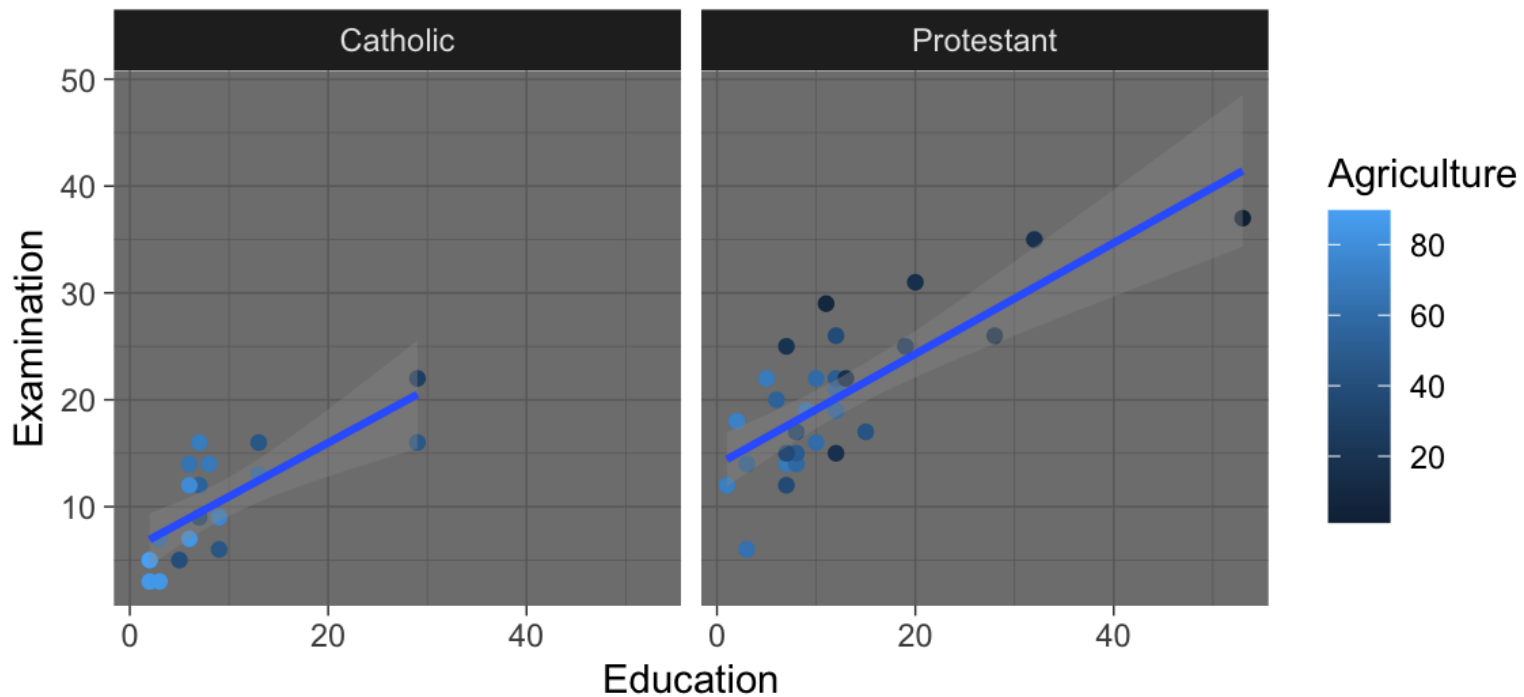
```
## `geom_smooth()` using formula 'y ~ x'
```



# Themes

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +  
  geom_point(aes(color = Agriculture)) +  
  geom_smooth(method = 'lm') +  
  facet_wrap(~Religion) +  
  theme_dark()
```

```
## `geom_smooth()` using formula 'y ~ x'
```





# Dynamic Documents

Q&A

# References

R Core Team. 2018. **R: A Language and Environment for Statistical Computing**. Vienna, Austria: R Foundation for Statistical Computing. <https://www.R-project.org/>.

Sarkar, Deepayan. 2008. **Lattice: Multivariate Data Visualization with R**. New York: Springer. <http://lmdvr.r-forge.r-project.org>.

Wickham, Hadley. 2016. **Ggplot2: Elegant Graphics for Data Analysis**. Springer-Verlag New York. <http://ggplot2.org>.