

# Crash course: Geospatial Datavisualisering

Jeppe Vierø

April 22, 2022

1 xx

2 Introduktion

3 section

4 Datastrukturer

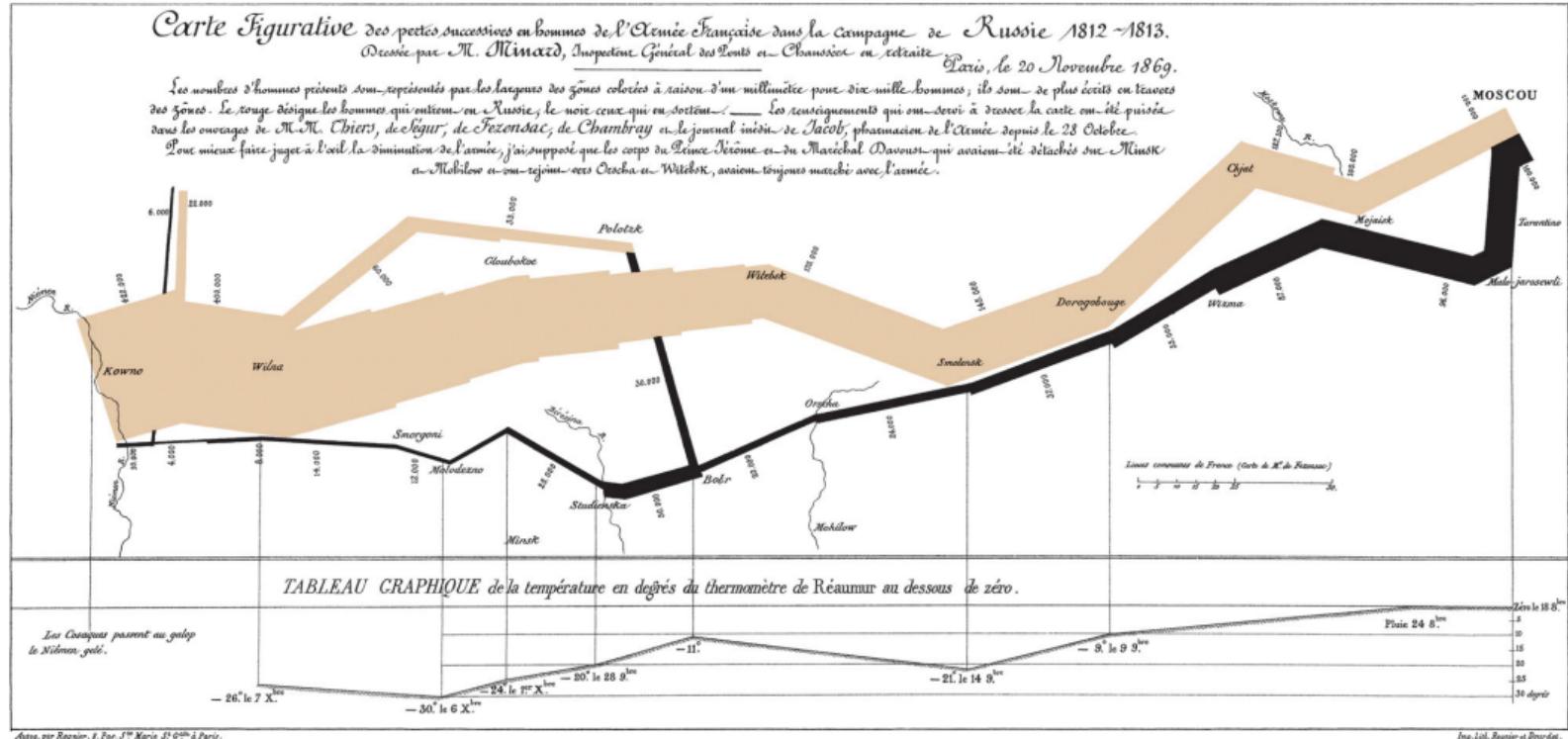
5 Datakilder

6 Indflyvning til værktøjer i R

xx

# Introduktion

# Motivation



# Afgrænsning

Jeg (regner med) at snakke **en del** om:

- Hvad **spatial** data er
- Hvordan vi kan bruge spatiale datakilder til at **visualisere** andet data
- Hvordan vi gør det i R

Jeg kommer **ikke** til at snakke (så meget) om:

- Datawrangling og -manipulation med geospatial data
- Datavisualisering generelt

## section

# section

# load

```
library(tidyverse)
library(janitor)
library(sf)
library(tmap)
library(repinion)

# Installér {repinion}, hvis I ikke har den:
# devtools::install_github("jviero/repinion")
```

XXXX

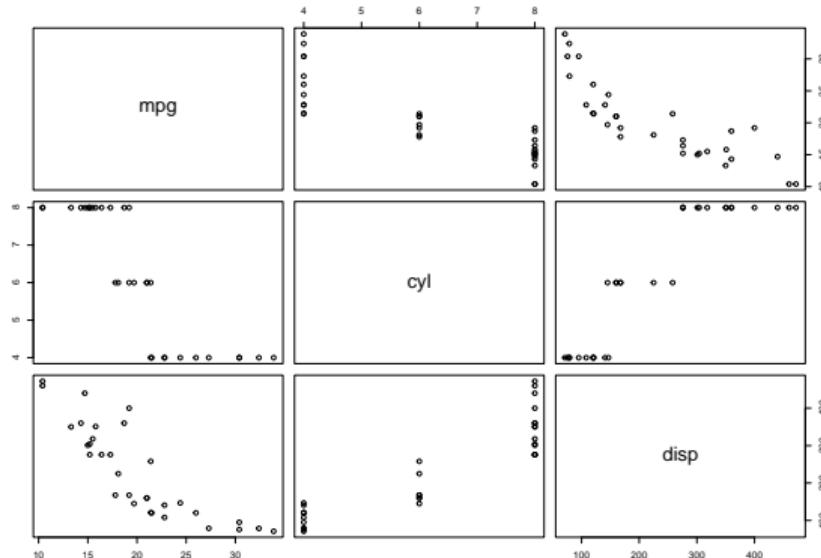
XXX

```
rejser <- readRDS("data/rejsekortdata.rds")
head(rejser, 5)
```

```
## # A tibble: 5 x 3
##   count station      share
##   <int> <chr>        <dbl>
## 1 5875 Nørreport St. 0.0688
## 2 5875 Kongens Nytorv St. 0.0652
## 3 5875 København H 0.0482
## 4 5875 Trianglen St. 0.0480
## 5 5875 Frederiksberg St. 0.0476
```

y

```
plot(mtcars[, 1:3])
```



# Datastrukturer

# Need to know om geodata

text

# Typer af geodata

Grundlæggende arbejder vi med **tre typer af geospationale datakilder**

Hver type har en (nogenlunde) parallel til graftyper, I er vant til at arbejde med:

## ① Punkter

- Tænk på dem som almindelige *punkter i et scatterplot*

## ② Linjer

- Tænk på dem som *linjer i et linechart*

## ③ Polygoner

- Her er parallelen ikke lige så tydelig
- ... men i en data viz-kontekst kan I tænke på dem som *søjler i et bar chart* (ish...)

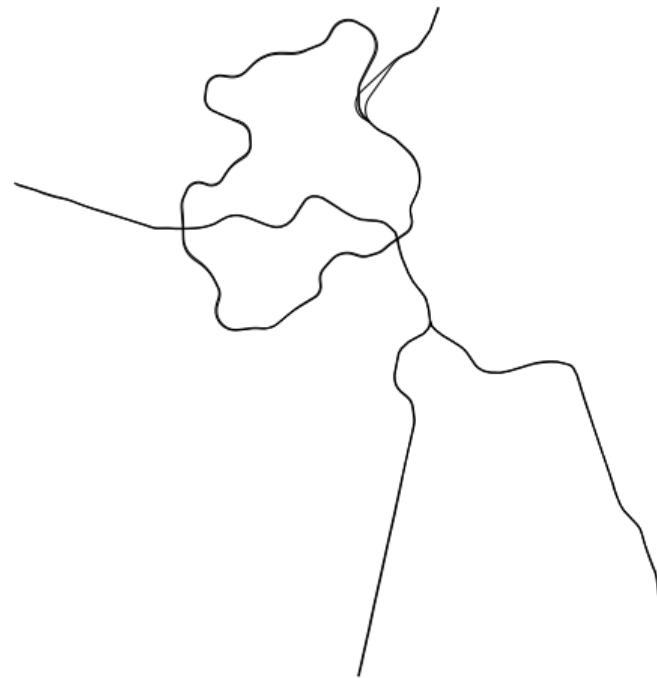
# (1) Punkter

- Punkter består af simple koordinater ( $x, y$ ), der refererer til en specifik lokation
- Punkter har ingen størrelse (og intet *areal*), de er uendeligt små
- Eksempler: byer, stationer, skoler osv.



## (2) Linjer

- Linjer består – grundlæggende – af punkter, der er kombineret til en *linestring* vha. en defineret rækkefølge
- Konstruktionen er sjældent noget, I skal bekymre jer om: linjedata ligger typisk opbevaret som linjer ( $\neq$  punkter). Her er det bare plug 'n play
- Linjer har intet *areal* (fordi de består af punkter)
- Eksempler: veje, floder, jernbanenetværk osv.



### (3) Polygoner

- Polygoner består – ligesom linjer – af punkter, der er kombineret til en *polygon* vha. en defineret rækkefølge. Igen, det er sjældent noget, I skal bekymre jer om
- Forskellen er, at polygoner er *lukkede linjer*, der former et afgrænset område
- De kan have alle tænkelige former. Det centrale er, at polygoner har et *areal*
- Eksempler: stater, kommuner, valgkredse osv.



# Datakilder

# DAGI

*“Danmarks Administratieve Geografiske Inddeling (DAGI) beskriver landets administrative og geografiske inddeling i kommuner, regioner, sogne, retskredse, politikredse, postnumre, opstillingskredse og lignende.” – DAWA*



Styrelsen for  
Dataforsyning og  
Effektivisering

DAGI

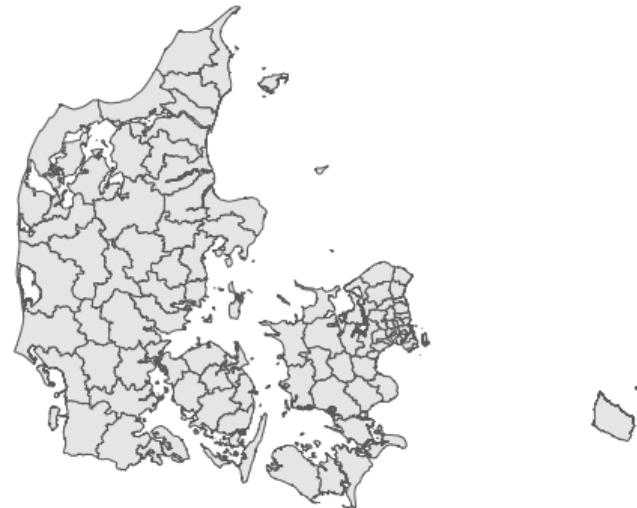
text [Danmarks Administrative Geografiske Inddeling \(DAGI\)](#)

# DAGI

- Vi kan let tilgå en masse crisp data på de danske administrative enheder vha. [den tilhørende API](#)
- API'en er plug 'n play, hvor vi kan vælge de **enheder**, vi skal bruge, og specificere **geojson**:
- <https://api.dataforsyningen.dk/\textcolor{purple}{\{kommuner\}}?format=\textcolor{teal}{\{geojson\}}>

# DAGI: et eksempel

```
url <-  
  "https://api.dataforsyningen.dk/kommuner?format=geojson"  
  
kommuner <-  
  read_sf(url)  
  
ggplot() +  
  geom_sf(data = kommuner) +  
  epitheme_map()
```



# OpenStreetMap

text [link](#)

## Indflyvning til værktøjer i R

text

xx