

Crash course: Geospatial Datavisualisering

Jeppe Vierø

April 25, 2022

1 xx

2 Introduktion

3 Datastrukturer

4 Datakilder

5 Værktøjer i R

xx

Introduktion

Motivation

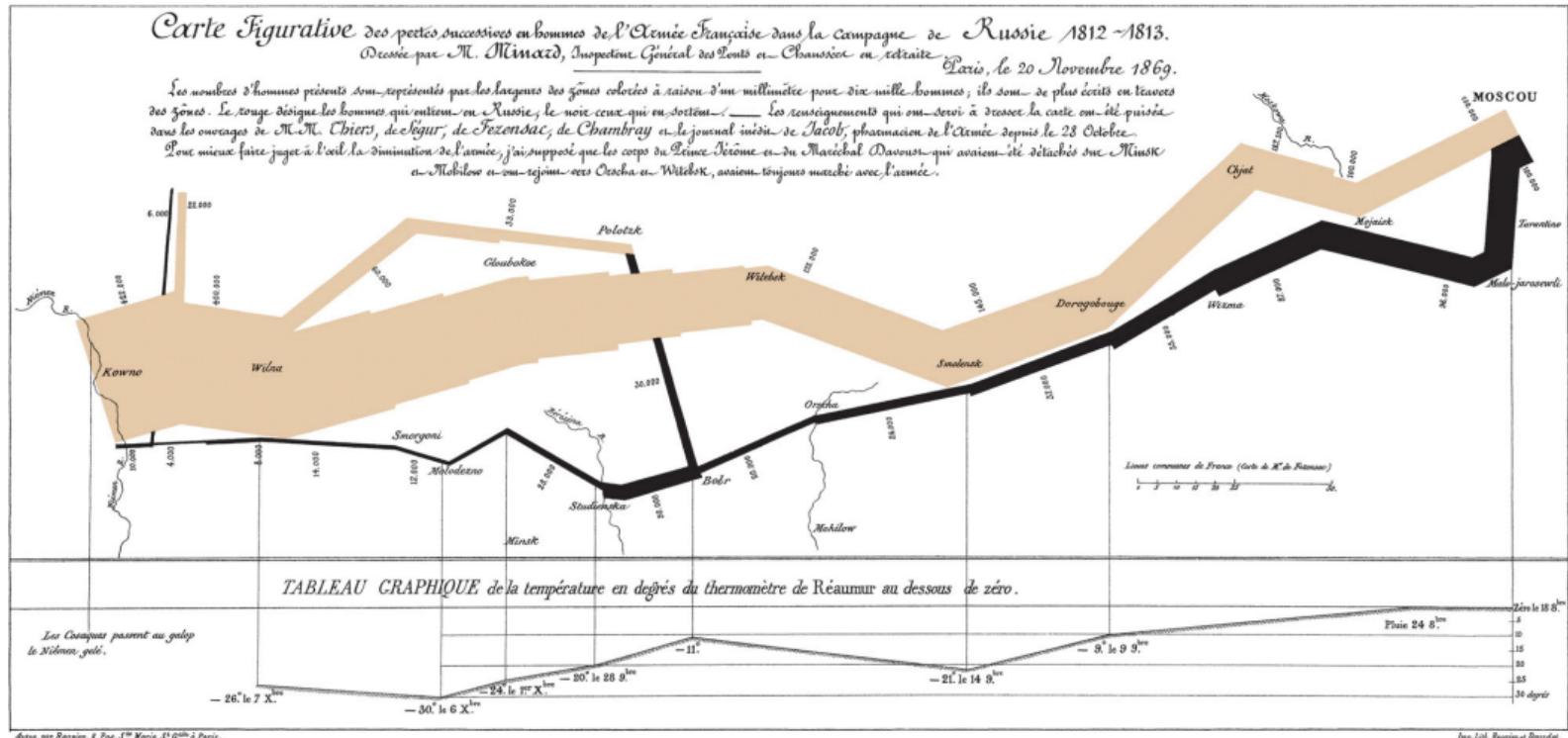


Figure 1: by Charles Joseph Minard, 1869

Afgrænsning

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

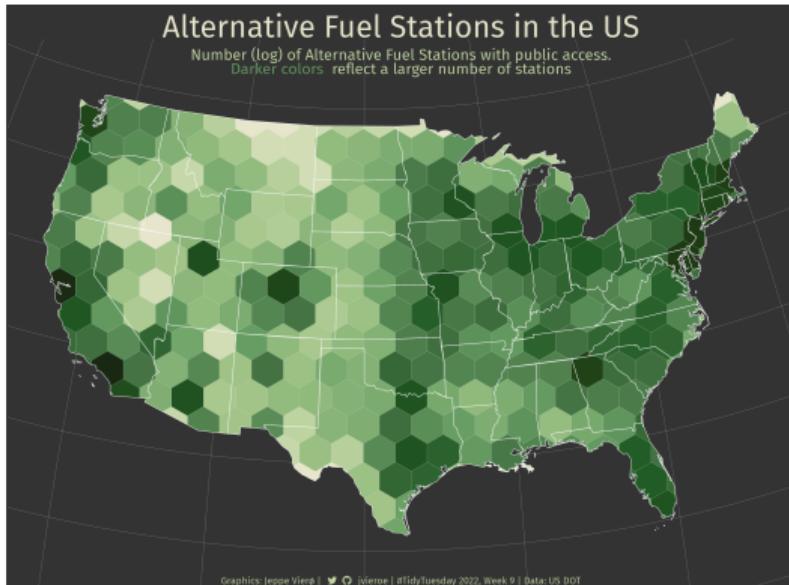
- Hvad **spatial** data er
- Hvordan vi kan bruge spatiale datakilder til at **visualisere** andet data
- Hvordan vi gør det i R

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

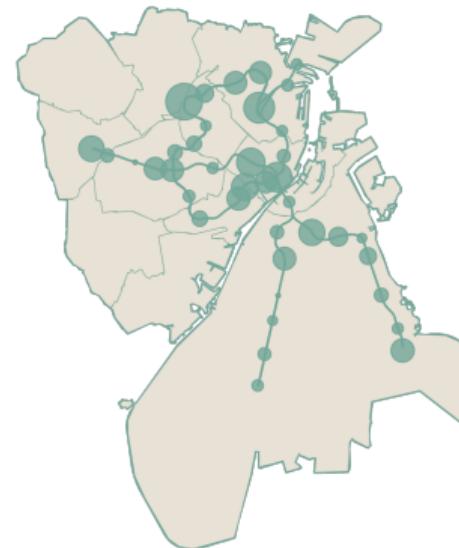
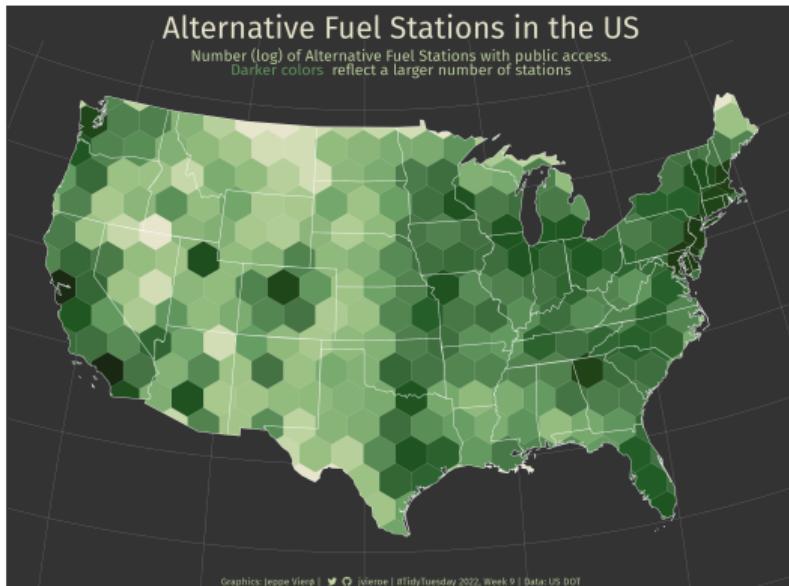
- Datawrangling og -manipulation med geospatial data
- Datavisualisering generelt

Hvordan kan vi bruge geodata til at visualisere vores pointer?

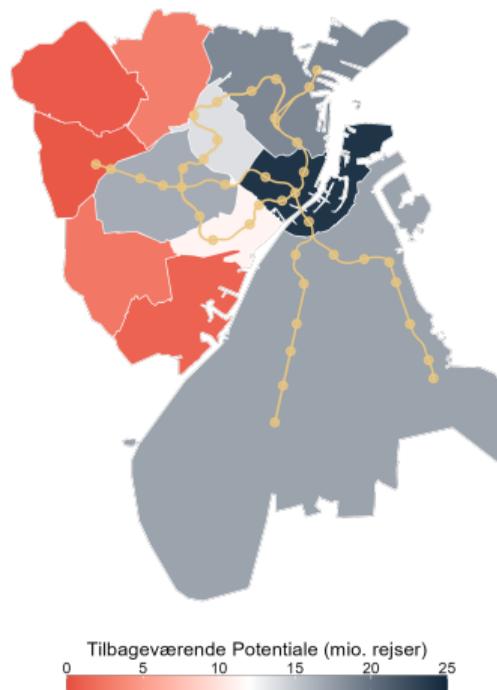
Hvordan kan vi bruge geodata til at visualisere vores pointer?



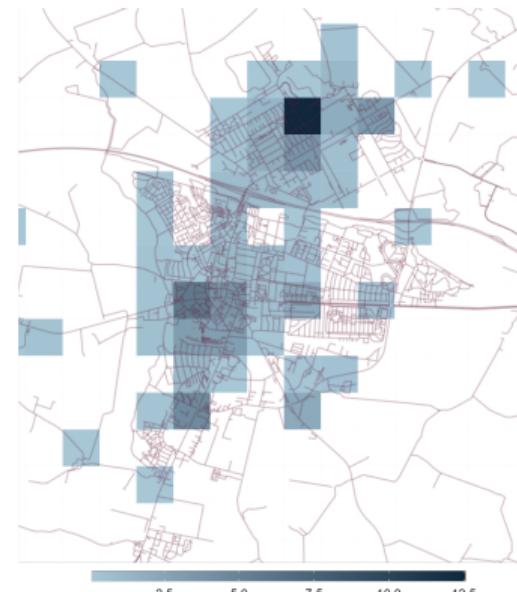
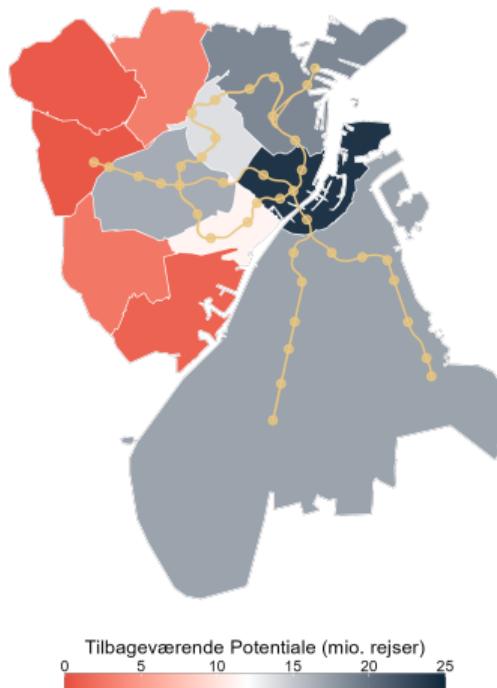
Hvordan kan vi bruge geodata til at visualisere vores pointer?



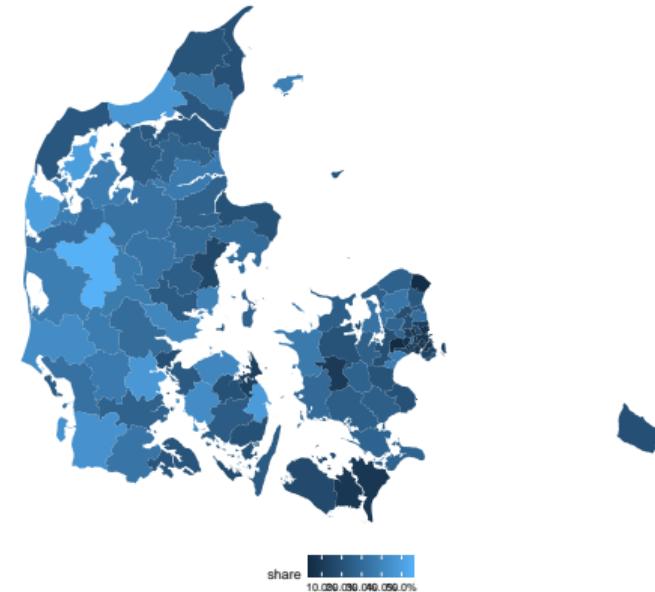
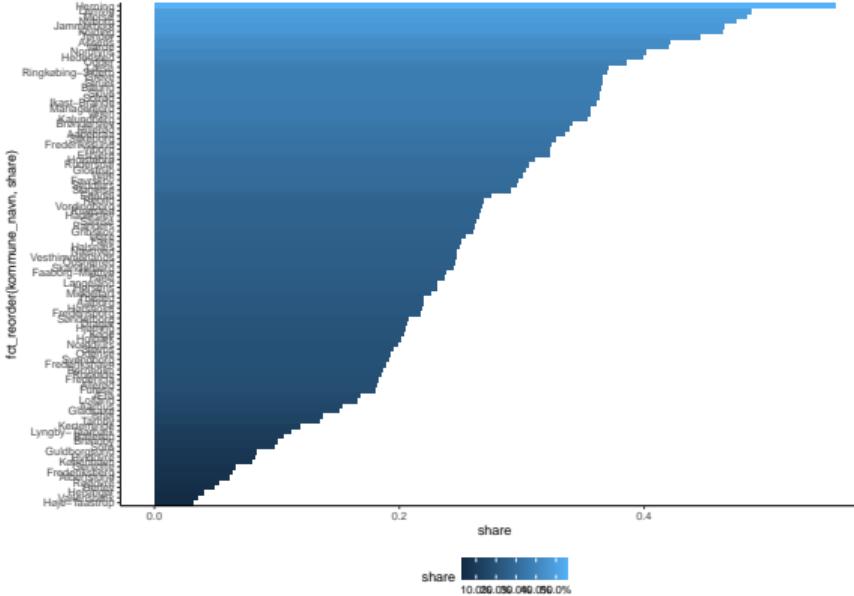
Hvordan kan vi bruge geodata til at visualisere vores pointer?



Hvordan kan vi bruge geodata til at visualisere vores pointer?



Hvorfor ikke bare bruge klassiske visualiseringer?



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...)

load

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

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

xxxx

xxx

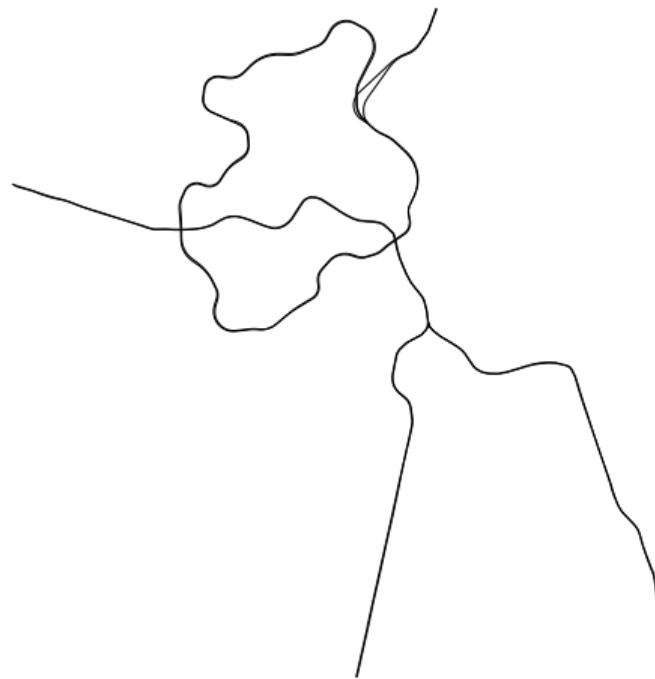
(1) Punkter

- 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 Administrative Geografiske Inddeling (DAGI)* beskriver landets administrative og geografiske inddeling i kommuner, regioner, sogne, retskredse, politikredse, postnumre, opstillingskredse og lignende.” – [DAWA](#)
- ... med andre ord; alt hvad vi kunne drømme om
- DAGI-data kan hentes via Styrelsen for Dataforsyning og Effektiviseringens [Datafordeler](#)
- Det er en ret håbløs hjemmeside, til gengæld er der masser at vælge mellem (inkl. historiske enheder!)



Styrelsen for
Dataforsyning og
Effektivisering

DAGI

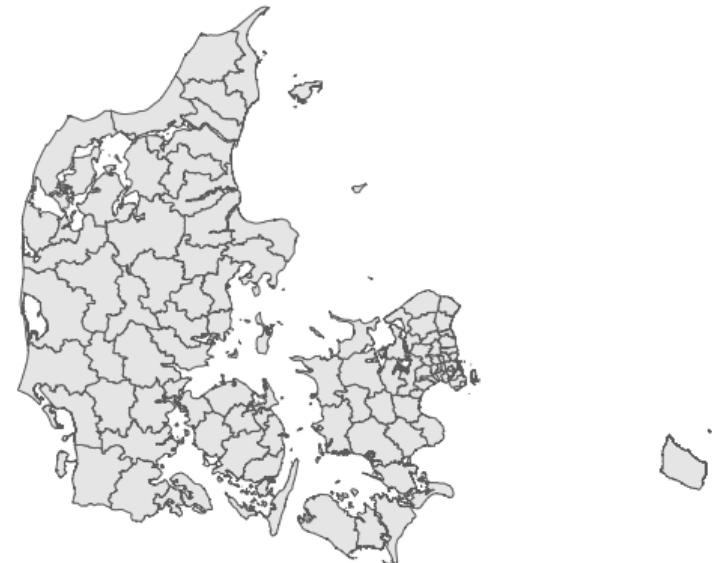
- Til de fleste formål kan vi hoppe uden om Datafordelen ved at bruge [DAWA \(Danmarks Adressers Web API\)](#) og den tilhørende [API](#)
- API'en er plug 'n play, hvor vi kan vælge de [enheder](#), vi skal bruge, og specificere [format](#):
- <https://api.dataforsyningen.dk/\protect\unhbox\voidb@x{\color{purple}kommuner}?format=\protect\unhbox\voidb@x{\color{teal}geojson}>

DAGI: et eksempel

```
# definér data
url <-
  "https://api.dataforsyningen.dk/kommuner?format=geojson"

# indlæs data
kommuner_raw <-
  read_sf(url)

# plot data
ggplot() +
  geom_sf(data = kommuner_raw) +
  epitheme_map()
```



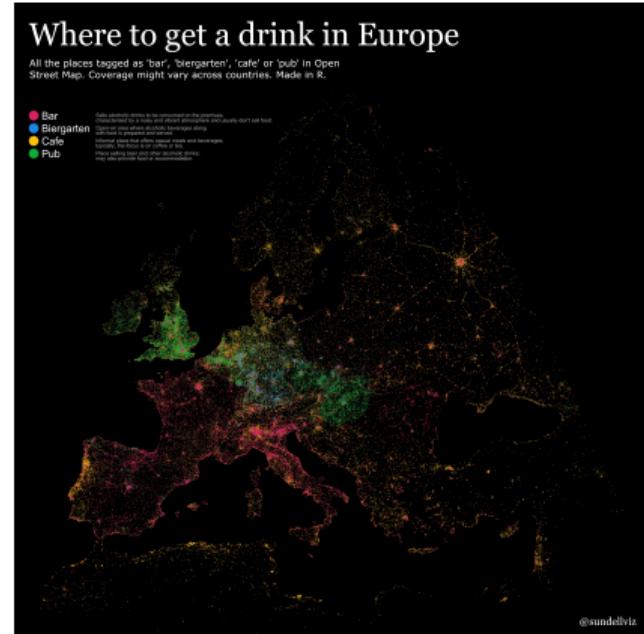
OpenStreetMap

- OpenStreetMap (OSM) er en crowd sourced geografisk database med detaljeret information om hele verden
- OSM indeholder data på (næsten) alt, hvad hjertet begærer
- OSM har en tilhørende [wiki](#), med en oversigt over de forskellige features



OpenStreetMap

- Vi kan bruge R-pakken `{osmdata}` til at hente OSM-data direkte i R
- Her skal vi bruge
 - En geografisk afgrænsning
 - Valg af features (vha. argumenterne key og value):



OpenStreetMap: et eksempel

```

library(osmdata)

# Hent OSM-data for Vejle
vejle <- kommuner_raw %>%
  filter(navn == "Vejle")

vejle_bbox <- vejle %>%
  st_bbox()

osm <- vejle_bbox %>%
  opq()

# Hent udvalgte veje
roads <- osm %>%
  add_osm_feature(key = 'highway',
                  value = c('motorway', 'trunk',
                           'primary', 'secondary',
                           'tertiary')) %>%
  osmdata_sf()

# Beskær
roads <- roads$osm_lines %>%
  st_intersection(., vejle)

# Plot vejene
ggplot() +
  geom_sf(data = vejle, fill = "white") +
  geom_sf(data = roads, aes(color = as.numeric(maxspeed))) +
  scale_color_viridis_c(direction = -1, name = "Speed limit (km/h)") +
  epitheme_map()

```

Data (c) OpenStreetMap contributors, ODbL 1.0. <https://www.openstreetmap.org/copyright>

Speed limit (km/h)

50 75 100 125

Værktøjer i R

text

xx