Efficient microdata

Efficient programming on the CBS microdata environment

Erik-Jan van Kesteren

Today

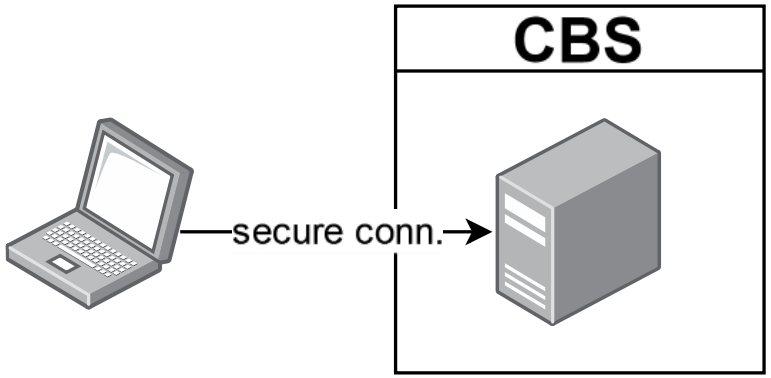
- The CBS RA fundamentals
- Project structure & reproducibility
- Efficient data handling
  - Storage
  - Memory

- Consultation & exercise!

CBS RA fundamentals

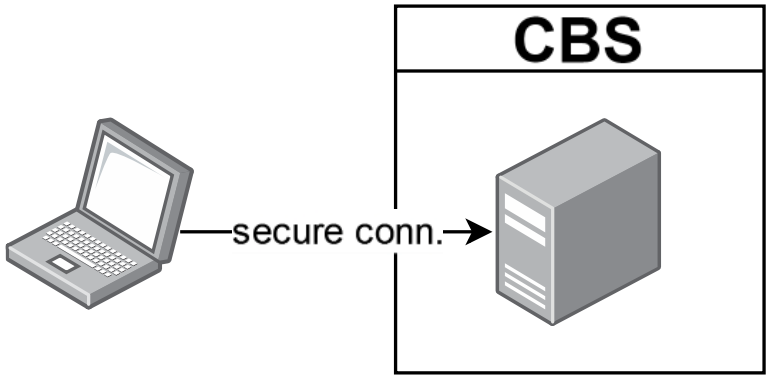
CBS Remote Access environment

- A virtual machine on a big server in the internal network
- Normal windows OS with 100GB storage
- R, python, SPSS, Stata, and more installed by default
- If you want specific libraries, ask the microdata team
- No fancy environment management



CBS Remote Access environment

- Data is made available via a drive on a per-project basis G:/microdata
- You can see all tables, but you have access only to those you requested
- Additional metadata is also available (for everyone)



Microdata at CBS

- Register data and questionnaires
- You can (subject to restrictions and costs) also upload your own data & link
- All these tables can be combined to do your research!

https://www.cbs.nl/nl-nl/onze-diensten/maatwerk-en-microdata/microdata-zelf-onderzoek-doen/catalogus-microdata

Catalogus microdata

Onder strikte voorwaarden kunnen instanties microdata gebruiken om zelf onderzoek te doen. Hieronder ziet u per thema de recentste documentatierapporten van de beschikbare microdatabestanden:

- Arbeid en sociale zekerheid
- Bedrijven
- Bevolking
- Bouwen en wonen
- Financiële en zakelijke diensten
- Gezondheid en welzijn
- Handel en horeca
- Inkomen en bestedingen
- Internationale handel
- Industrie en energie
- Landbouw
- Macro-economie
- Natuur en milieu
- Onderwijs
- Overheid en politiek
- Prijzen
- Veiligheid en recht
- Verkeer en vervoer
- Vrije tijd en cultuur

Microdata at CBS

- The tables are made by humans / different departments: manual work
- They are (mostly) SPSS .sav files
- Some files are huge! (SPOLISBUS)
- Their names / versions can change without warning

https://www.cbs.nl/nl-nl/onze-diensten/maatwerk-en-microdata/microdata-zelf-onderzoek-doen/catalogus-microdata

Catalogus microdata

Onder strikte voorwaarden kunnen instanties microdata gebruiken om zelf onderzoek te doen. Hieronder ziet u per thema de recentste documentatierapporten van de beschikbare microdatabestanden:

- Arbeid en sociale zekerheid
- Bedrijven
- Bevolking
- Bouwen en wonen
- Financiële en zakelijke diensten
- Gezondheid en welzijn
- Handel en horeca
- Inkomen en bestedingen
- Internationale handel
- Industrie en energie
- Landbouw
- Macro-economie
- Natuur en milieu
- Onderwijs
- Overheid en politiek
- Prijzen
- Veiligheid en recht
- Verkeer en vervoer
- Vrije tijd en cultuur

Additional data

- There are additional files to help with analysis
- Metadata & supplementary data. Translation files, key/value files, lists of existing postal codes, and more.
- These reside in a different location (not G:/)
- This location has also changed in the past & could change in the future too

Imports/exports

Exporting analysis results is subject to output check

- Ensures our privacy
- This is manual labour, done by microdata team member
- Each output costs time and money

You can also import and export code files

- This does not cost money!!
- More on this later

Any questions?

Structure

Efficient project folder structure

```
my_project/
├─ raw_data/
│   ├─ questionnaire_data.csv
├─ processed_data/
│   ├─ questionnaire_processed.rds
│   ├─ analysis_object.rds
├─ img/
│   ├─ plot.png
├─ 01_load_and_process_data.R
├─ 02_create_visualisations.R
├─ 03_main_analysis.R
├─ 04_output_results.R
├─ my_project.Rproj
├─ readme.md
```

Efficient project folder structure

```
my_project/
├─ raw_data/
│  ├─ questionnaire_data.csv
├─ processed_data/
│  ├─ questionnaire_processed.rds
│  ├─ analysis_object.rds
├─ img/
│  ├─ plot.png
├─ 01_load_and_process_data.R
├─ 02_create_visualisations.R
├─ 03_main_analysis.R
├─ 04_output_results.R
├─ my_project.Rproj
├─ readme.md
```

At CBS, this is on a different disk (G:/)!
Does not count towards your 100GB quote

Efficient project folder structure

```
my_project/
├─ raw_data/
│  ├─ questionnaire_data.csv
├─ processed_data/
│  ├─ questionnaire_processed.rds
│  ├─ analysis_object.rds
├─ img/
│  ├─ plot.png
├─ 01_load_and_process_data.R
├─ 02_create_visualisations.R
├─ 03_main_analysis.R
├─ 04_output_results.R
├─ my_project.Rproj
├─ readme.md
```

At CBS, this is on a different disk (G:/)!
Does not count towards your 100GB quote

Make these objects efficiently stored
Binary formats are better
later we'll also see database files

Efficient project folder structure

```
my_project/
├─ raw_data/
│  ├─ questionnaire_data.csv
├─ processed_data/
│  ├─ questionnaire_processed.rds
│  ├─ analysis_object.rds
├─ img/
│  ├─ plot.png
├─ 01_load_and_process_data.R
├─ 02_create_visualisations.R
├─ 03_main_analysis.R
├─ 04_output_results.R
├─ my_project.Rproj
├─ readme.md
```

At CBS, this is on a different disk (G:/)!
Does not count towards your 100GB quote

Make these objects efficiently stored
Binary formats are better
later we'll also see database files

Clear ordering & separation of tasks
Separating preprocessing & analysis

Efficient project folder structure

```
my_project/
├─ raw_data/
│  ├─ questionnaire_data.csv
├─ processed_data/
│  ├─ questionnaire_processed.rds
│  ├─ analysis_object.rds
├─ img/
│  ├─ plot.png
├─ 01_load_and_process_data.R
├─ 02_create_visualisations.R
├─ 03_main_analysis.R
├─ 04_output_results.R
├─ my_project.Rproj
├─ readme.md
```

At CBS, this is on a different disk (G:/)!
Does not count towards your 100GB quote

Make these objects efficiently stored
Binary formats are better
later we'll also see database files

Clear ordering & separation of tasks
Separating preprocessing & analysis

Use .Rproj file for portability

Live coding 1: example project

DOI 10.5281/zenodo.6504837

Reproducibility & offline workflow

Reproducibility

- After your project, export your structured code folder!
- Double check your documentation
- Create a DOI / archived version (OSF, Zenodo)
- https://odissei-soda.nl/tutorials/post-1/

- This way, others can benefit from your work
- Let ODISSEI know ☺
- https://odissei-data.github.io/ODISSEI-code-library/

Reproducibility

You cannot export data; create synthetic data

- Create “fake” version of input data outside the RA
- You can do this with a script
- Don’t do this for all input tables (too much work!)
- Do this for the intermediate processed table

Computational reproducibility

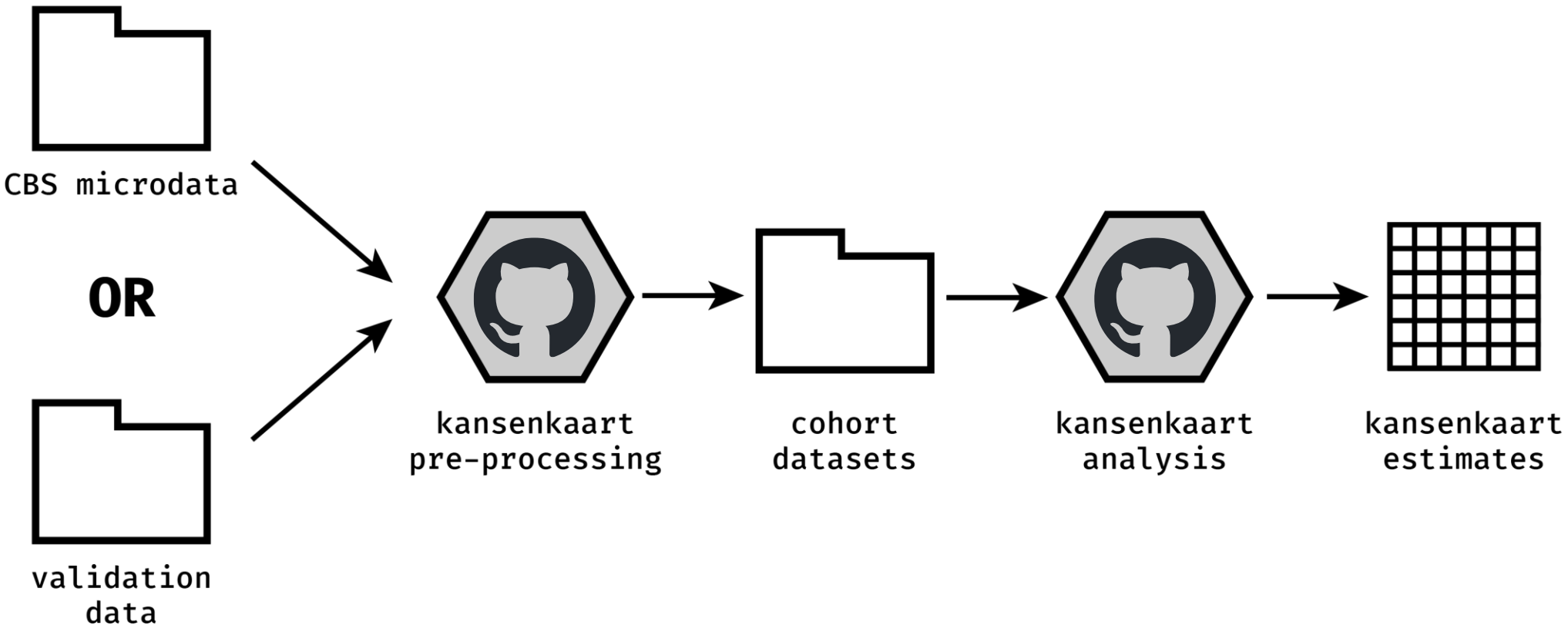
Offline workflow example

You might want to go a step further with version control

1. Develop preprocessing & analysis on GitHub
2. Upload to CBS RA
3. Do some small code fixes there to make it run
4. Run once, export results
5. Export code from CBS, update GitHub repository

Recommended for complex / multi-collaborator projects

Offline workflow example

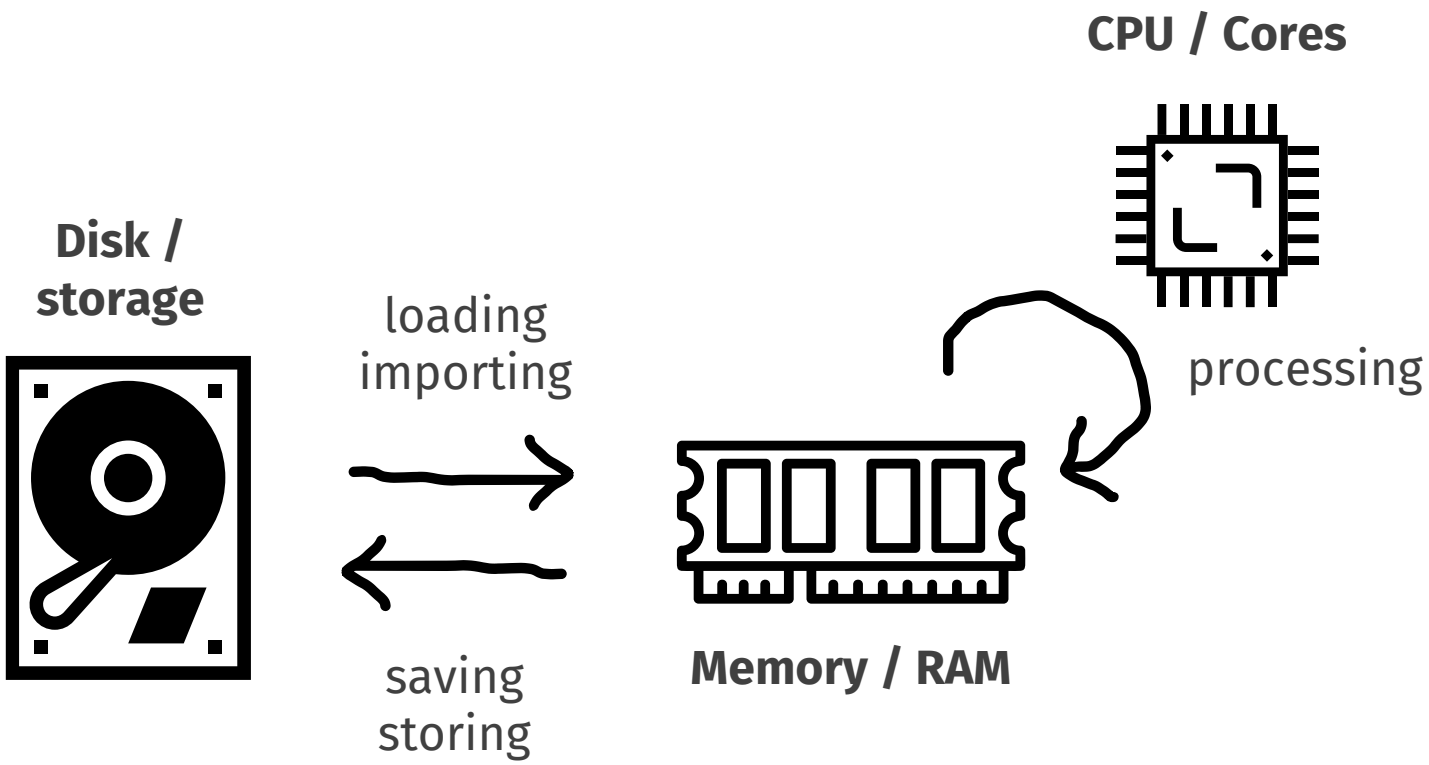


Offline workflow example

How far you should go depends a lot on the project

- Contact us in consultation time later or schedule a meeting with us

Efficient data handling


CPU / Cores
Disk / storage
loading
importing
processing
saving
storing
Memory / RAM

Storage

Geachte relatie,

Uit een meting op maandag 4 april 2022 blijkt dat project 0000, Titel van het project, een ruimtebeslag kent van **133** GB. De limiet voor het project is **100** GB.

Als u de extra capaciteit daadwerkelijk nodig heeft, dan kunt u een verzoek indienen om extra capaciteit bij te kopen. De kosten hiervoor bedragen 25 euro per 50 GB per maand.

Met vriendelijke groet,

Firstname Lastname
DBD Team Dataservices

CBS | Henri Faasdreef 312 | Postbus 24500 | 2490 HA Den Haag
Email: microdata@cbs.nl
Volg statistiekcbs op twitter | facebook | instagram

Efficiently storing large R datasets

Live coding

Try to create tidy data

Intermediate data should be **tidy**

- Each variable is a column; each column is a variable.
- Each observation is a row; each row is an observation.
- Each value is a cell; each cell is a single value

https://tidyr.tidyverse.org/articles/tidy-data.html

Make column names legible for humans and machines

`janitor::clean_names()`

Memory

In R:

```
Error: cannot allocate vector of size 745.1 Gb
```

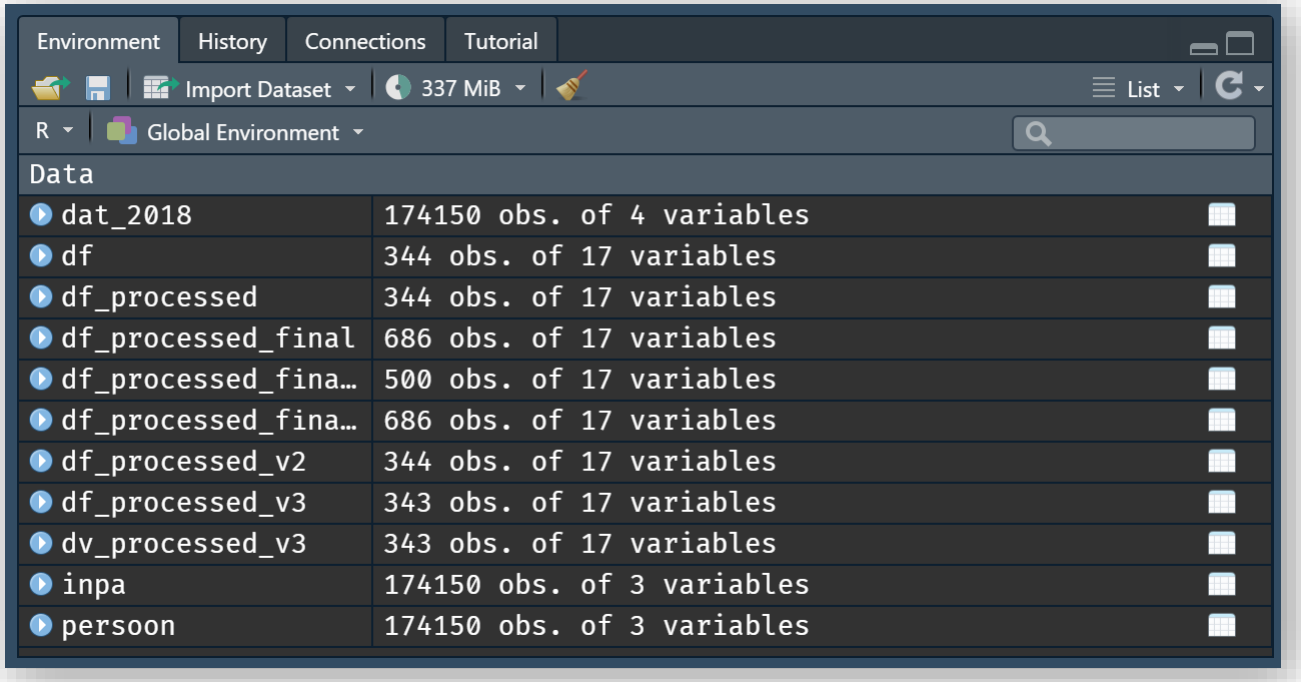
In Python (numpy)

```
numpy.core._exceptions._ArrayMemoryError:
Unable to allocate 745. GiB for an array with
shape (100000000000,) and data type float64
```

In Stata

(no clue, I really don’t use Stata??)

Clean your session / environment



Efficiently processing large datasets

Live coding 2

Larger-than-memory data

- Sometimes, your data really is larger-than-memory
- It is possible to do analyses on datasets which are on-disk

Two options:

- Create chunked data objects
- Create a proper database

Working with larger-than-memory data

Live coding 3

Questions?

Group consultations / exercise

Thank you!

SoDA

https://odissei-soda.nl