KANTON AARGAU

Menschen machen Zukunft

DEPARTEMENT
FINANZEN UND RESSOURCEN

Managing workflows in R with the {targets} package

October 26, 2022

Swiss Statistics Meeting 2022

Niklas Haffert, Tina Cornioley[1], Jan Wunder (Statistik Aargau)

[1] Statistique Vaud (since June 2022)

Throwback SST 2021

KANTON AARGAU
Menschen machen Zukunft
DEPARTEMENT
FINANZEN UND RESSOURCEN

Optimierung des Workflows in R
Améliorer le workflow dans R

6. September 2021

Schweizer Statistiktage 2021

Tina Cornioley, Jan Wunder (Statistik Aargau)

Tina Cornioley
Departement Finanzen und Ressourcen
Wissenschaftliche Mitarbeiterin

Jan Wunder
Departement Finanzen und Ressourcen
Wissenschaftlicher Mitarbeiter

Data Analysis Workflows

What we have

- Often grow and grow…
- Often consist of multiple data files, scripts…
- Can be computationally expensive

Data Analysis Workflows

What we have

- Often grow and grow…
- Often consist of multiple data files, scripts…
- Can be computationally expensive

What we want

- Stability/Correctness
- Reproducibility/Recyclability
- Speed
- (Automation)

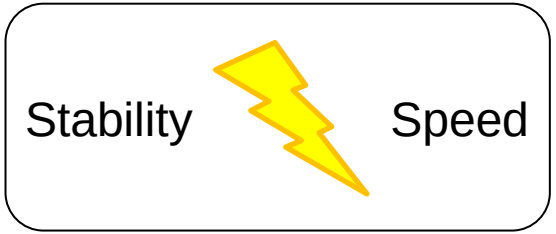
Data Analysis Workflows

What we have

- Often grow and grow…
- Often consist of multiple data files, scripts…
- Can be computationally expensive

What we want

- Stability/Correctness
- Reproducibility/Recyclability
- Speed
- (Automation)



Classic Project Structure

```
data/
results/
scripts/
├── analysis.R
├── clean_data.R
├── read_data.R
└── reporting.R
```

User needs to **figure out order of scripts** and execute them respectively

Classic Project Structure

Stability: +
Speed: +

```
data/
results/
scripts/
 ├─ analysis.R
 ├─ clean_data.R
 ├─ read_data.R
 └─ reporting.R
```

User needs to **figure out order of scripts** and execute them respectively

Enhanced Project Structure

```
data/
results/
scripts/
 ├─ 01_read_data.R
 ├─ 02_clean_data.R
 ├─ 03_analysis.R
 └─ 04_reporting.R
main.R
```

Numeration specifies order and script *main.R* executes everything, but user might **execute everything every time**

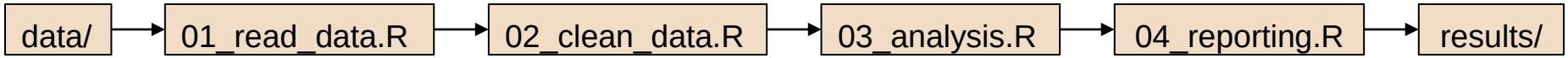
Enhanced Project Structure

Stability: **++**
Speed: **+**

```
data/
results/
scripts/
├─ 01_read_data.R
├─ 02_clean_data.R
├─ 03_analysis.R
└─ 04_reporting.R
main.R
```

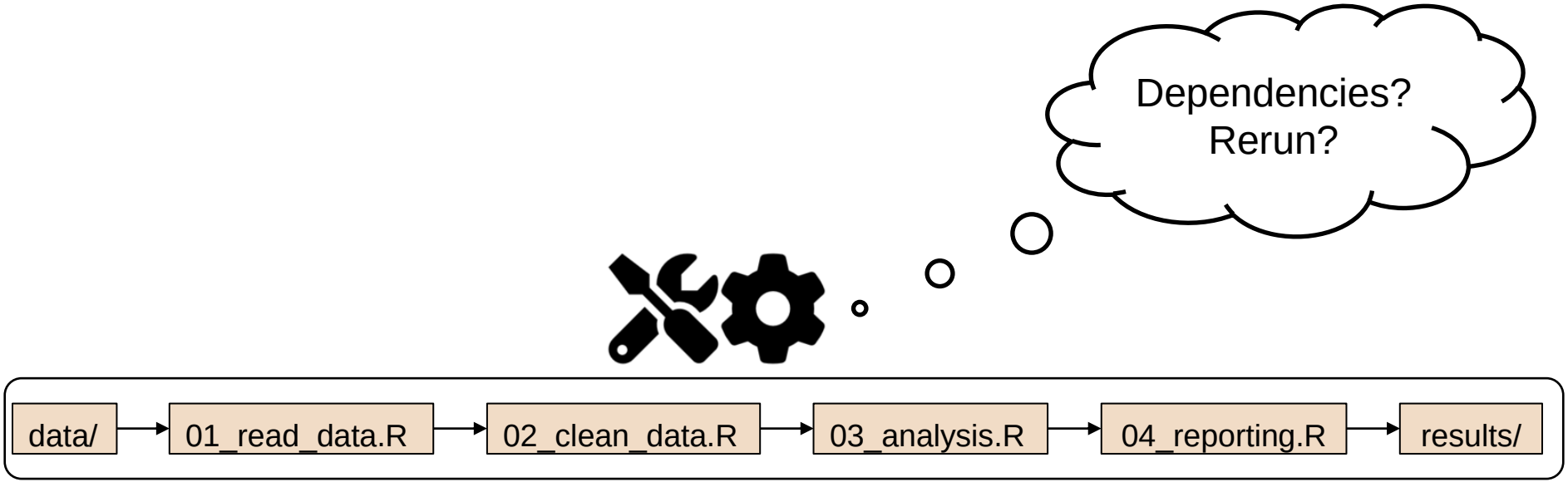
Numeration specifies order and script *main.R* executes everything, but user might **execute everything every time**

Workflow/Pipeline Project Structure



Order and dependencies clearly defined, visible which parts have to be executed

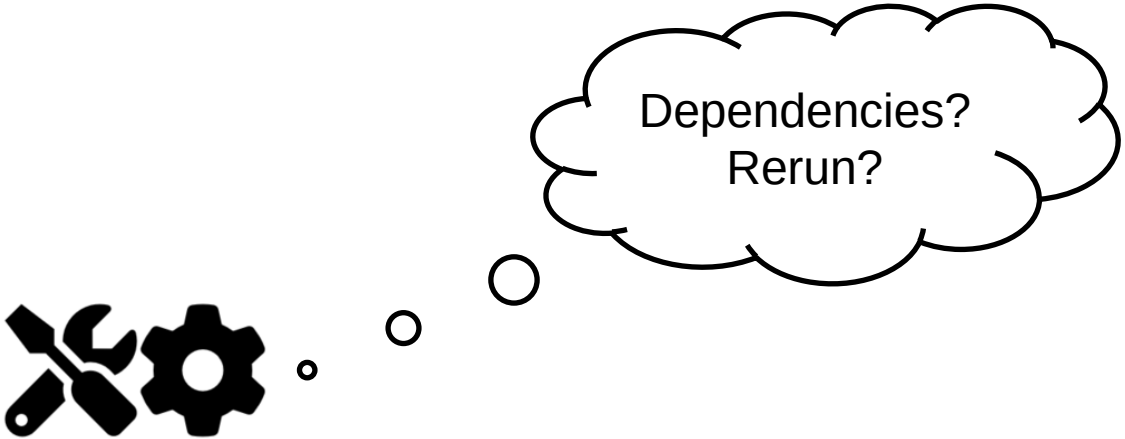
Workflow/Pipeline Tool



Order and dependencies clearly defined, tool tracks which parts have to be executed

Workflow/Pipeline Tool

Stability: +++
Speed: +++



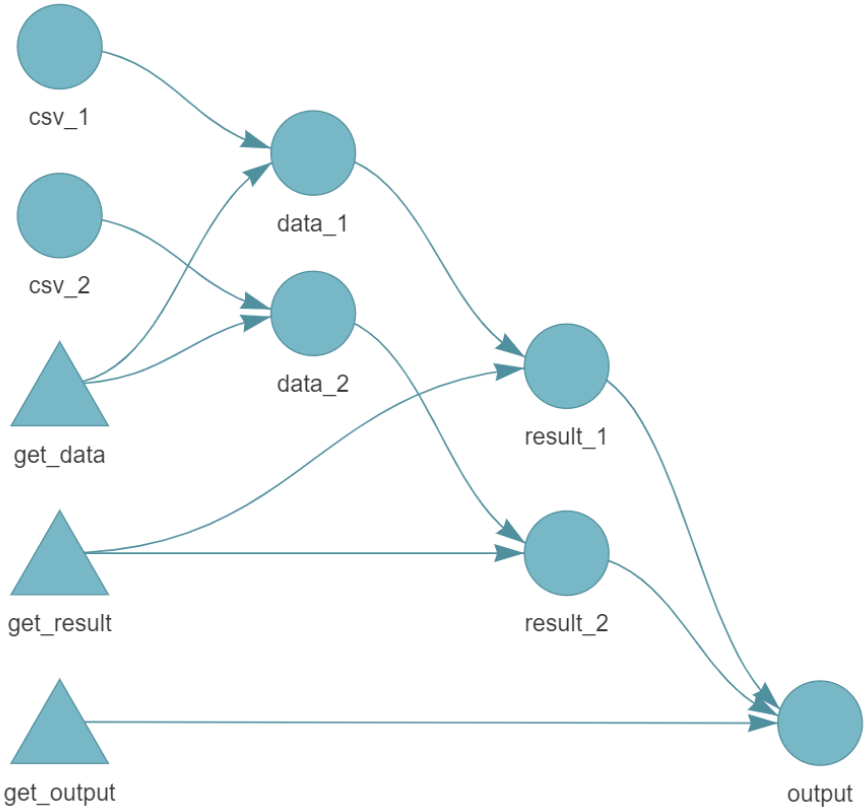
Order and dependencies clearly defined, tool tracks which parts have to be executed

That’s what {targets} does!

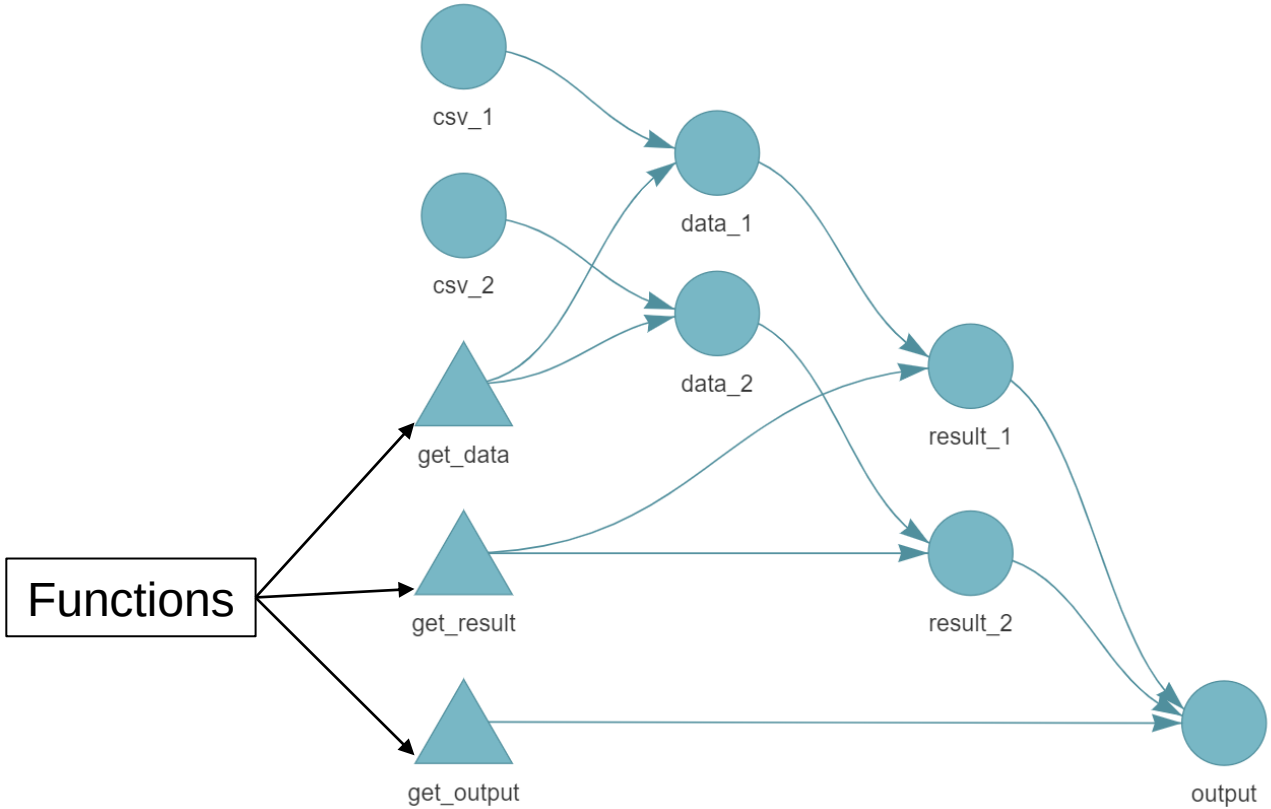
That’s what {targets} does!

- R-based pipeline tool
- Successor of {drake}
- Developed by Will Landau
- First version published in 2021

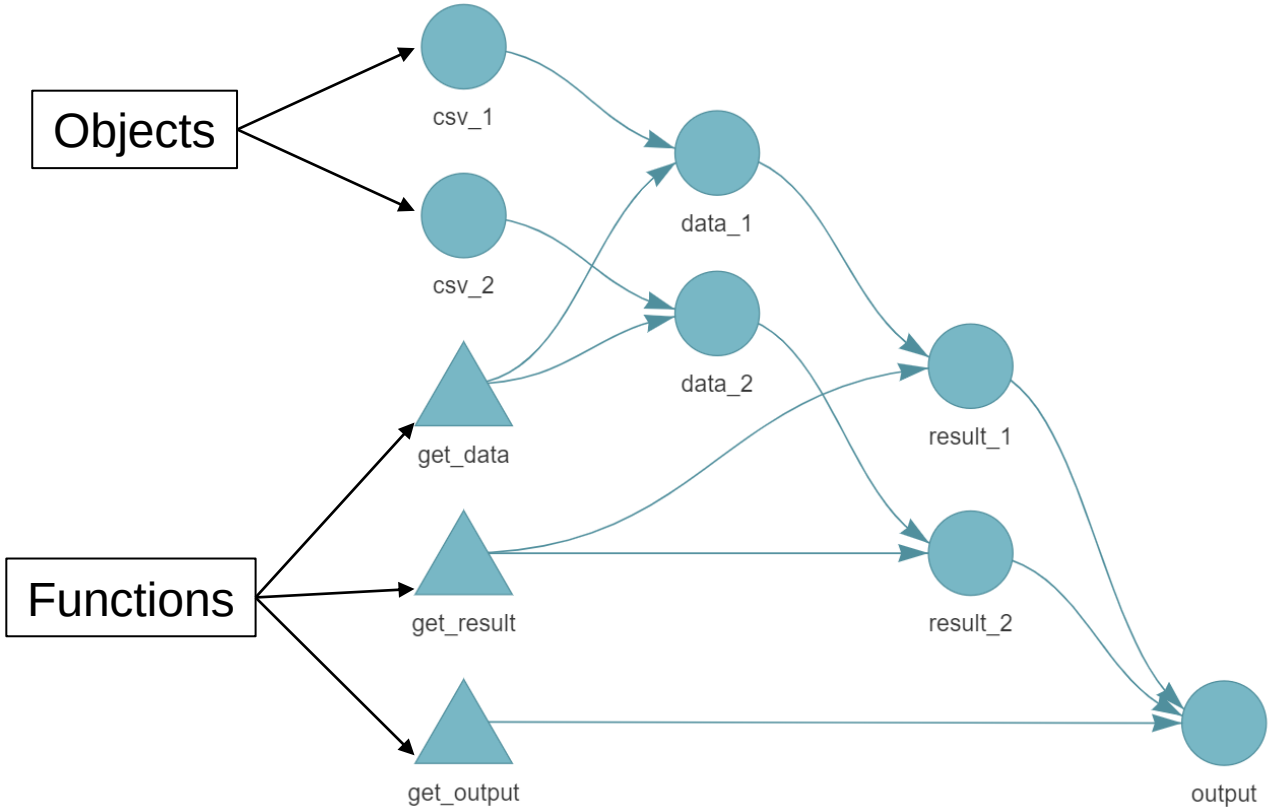
Example: {targets}-Workflow



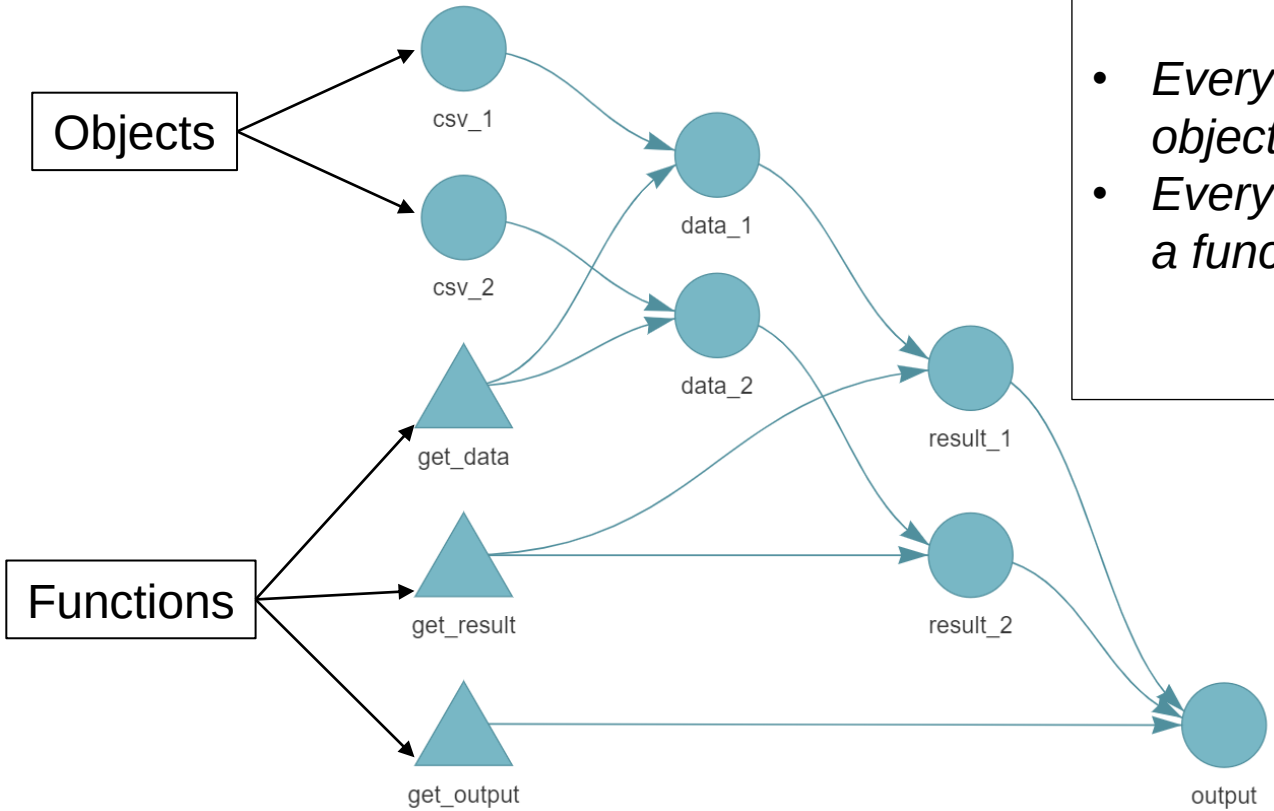
Example: {targets}-Workflow



Example: {targets}-Workflow



Example: {targets}-Workflow

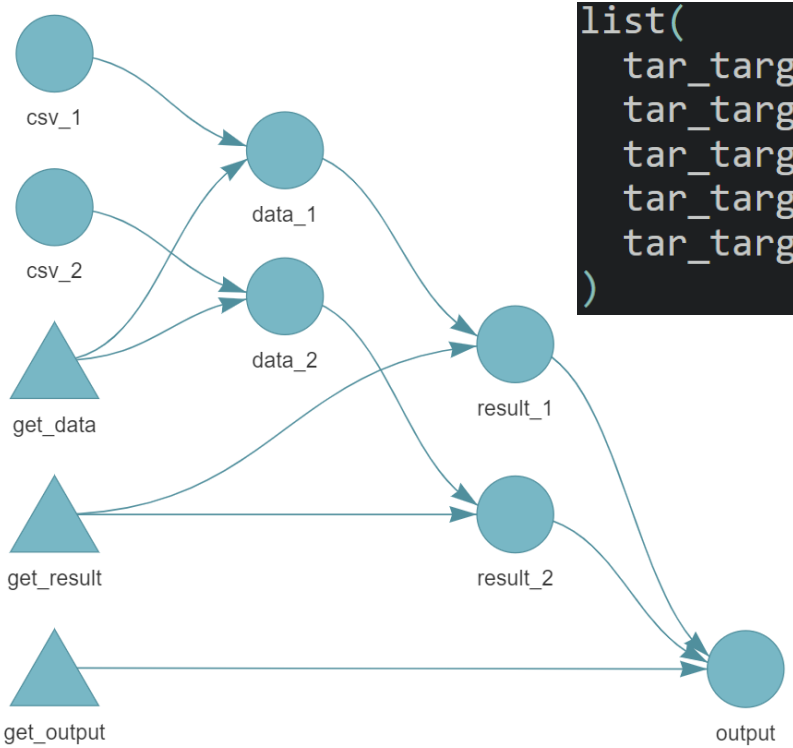


"To understand computations in R, two slogans are helpful:

- *Everything that exists is an object*
- *Everything that happens is a function call"*

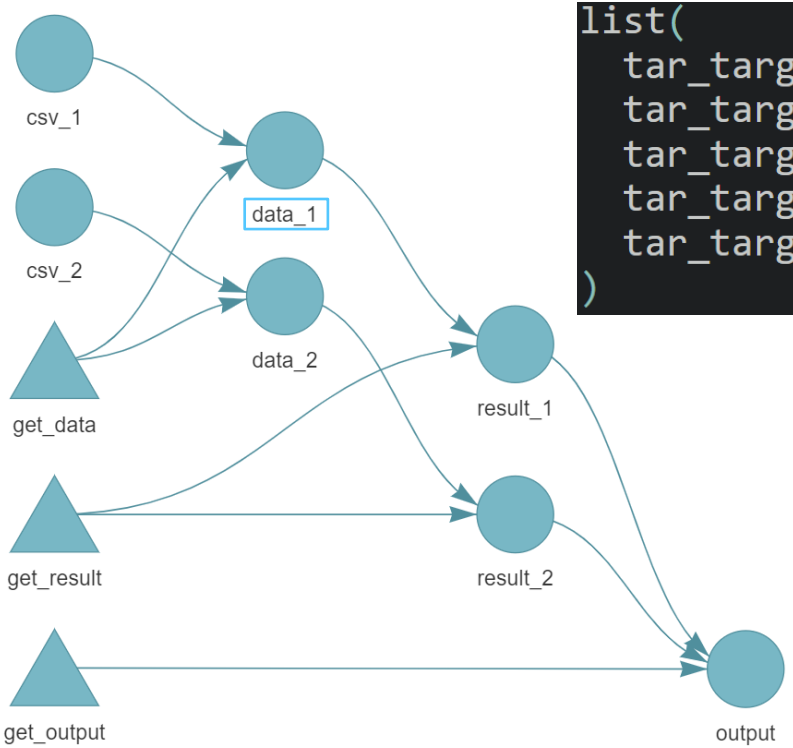
(John Chambers)

Example: {targets}-Workflow



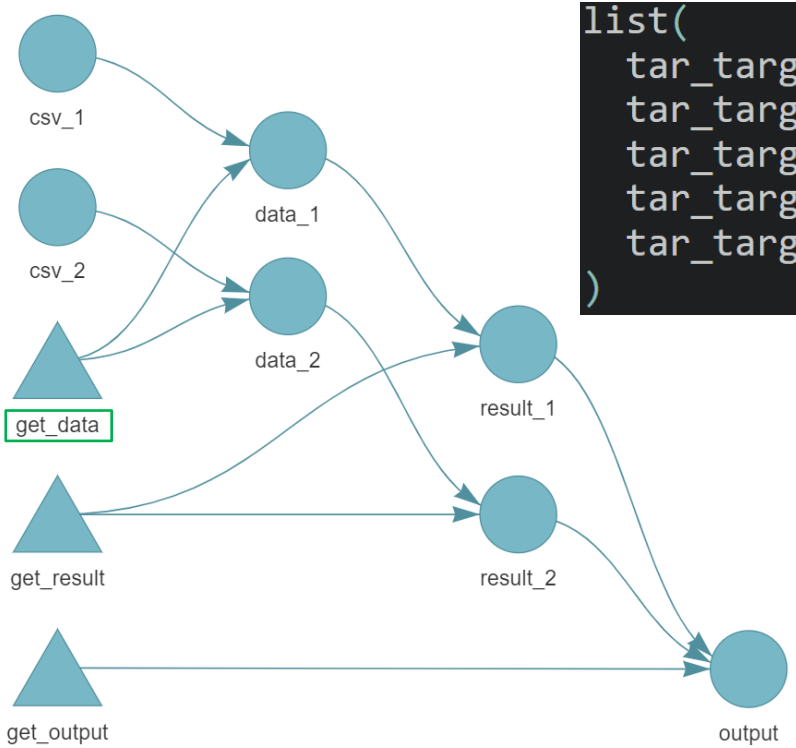

```
list(
  tar_target(data_1, get_data(csv_1)),
  tar_target(data_2, get_data(csv_2)),
  tar_target(result_1, get_result(data_1)),
  tar_target(result_2, get_result(data_2)),
  tar_target(output, get_output(result_1, result_2))
)
```

Example: {targets}-Workflow



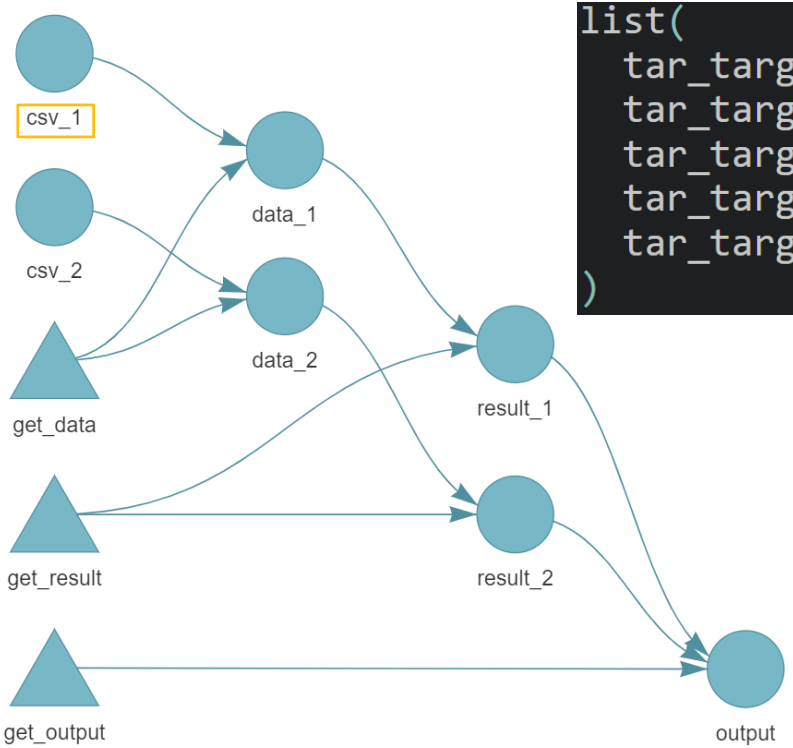

```
list(
  tar_target(data_1, get_data(csv_1)),
  tar_target(data_2, get_data(csv_2)),
  tar_target(result_1, get_result(data_1)),
  tar_target(result_2, get_result(data_2)),
  tar_target(output, get_output(result_1, result_2))
)
```

Example: {targets}-Workflow



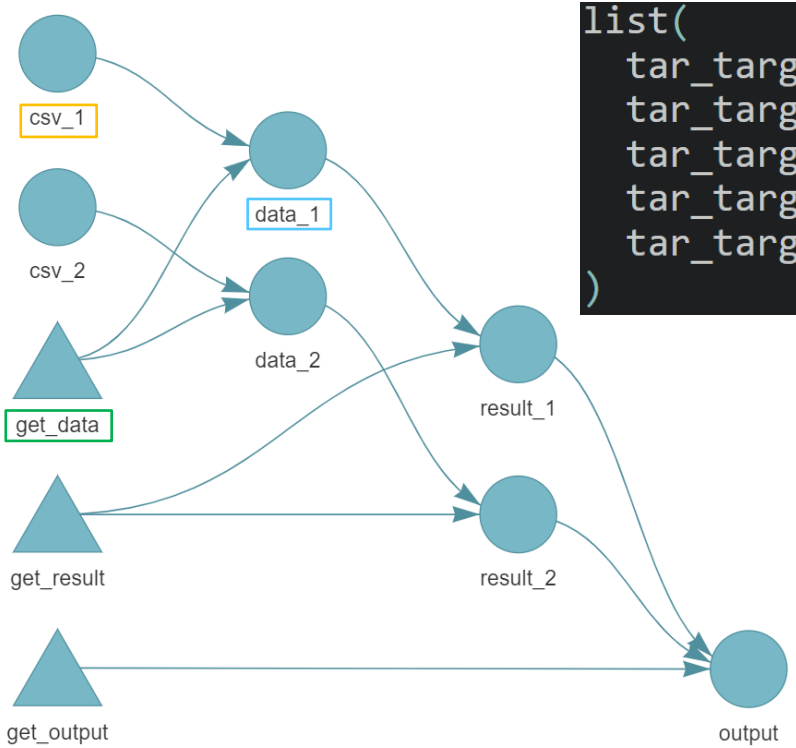

```
list(
  tar_target(data_1, get_data(csv_1)),
  tar_target(data_2, get_data(csv_2)),
  tar_target(result_1, get_result(data_1)),
  tar_target(result_2, get_result(data_2)),
  tar_target(output, get_output(result_1, result_2))
)
```

Example: {targets}-Workflow



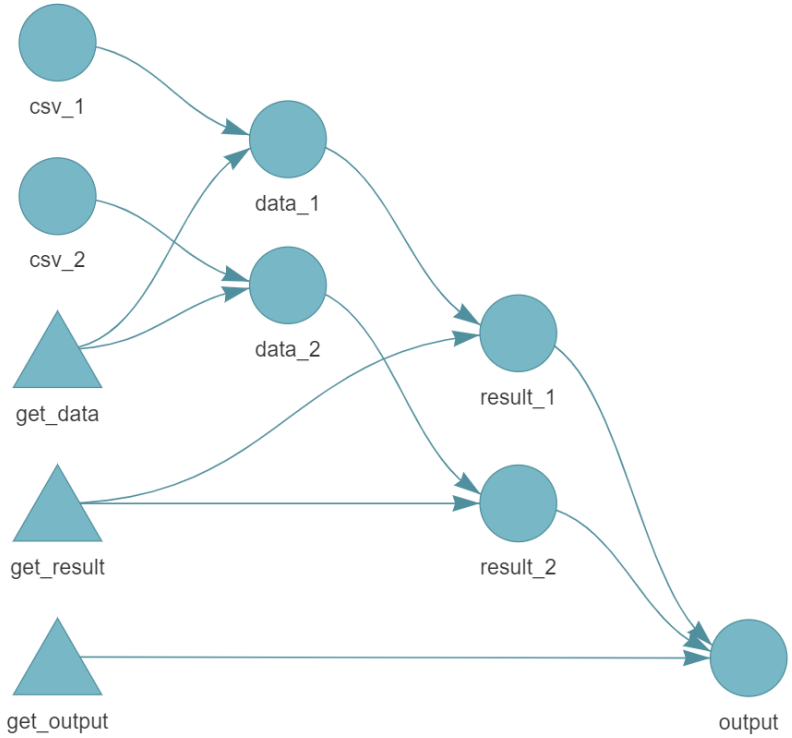

```
list(
  tar_target(data_1, get_data(csv_1)),
  tar_target(data_2, get_data(csv_2)),
  tar_target(result_1, get_result(data_1)),
  tar_target(result_2, get_result(data_2)),
  tar_target(output, get_output(result_1, result_2))
)
```

Example: {targets}-Workflow



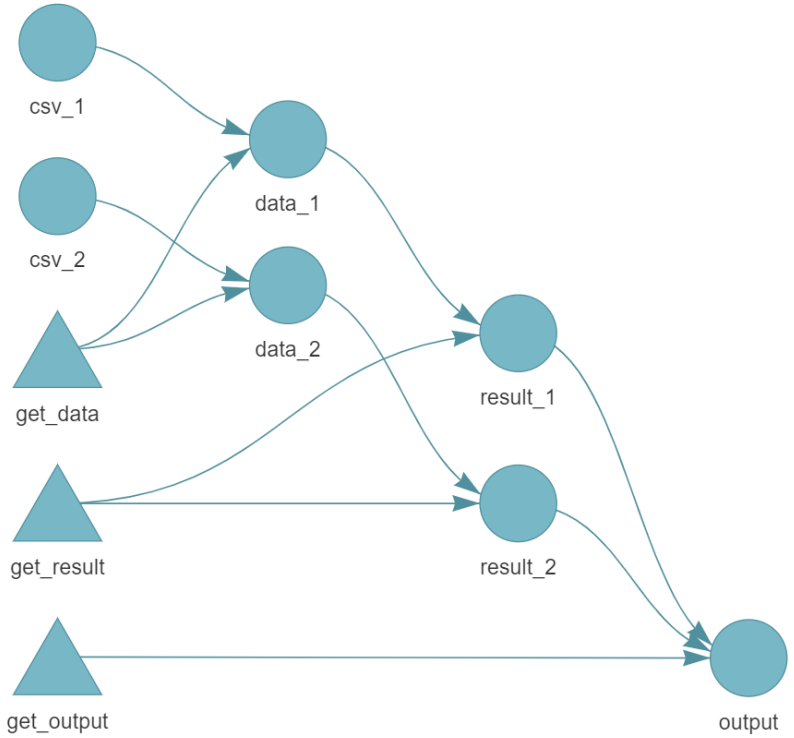

```
list(
  tar_target(data_1, get_data(csv_1)),
  tar_target(data_2, get_data(csv_2)),
  tar_target(result_1, get_result(data_1)),
  tar_target(result_2, get_result(data_2)),
  tar_target(output, get_output(result_1, result_2))
)
```

Advantages I: Stability/Correctness



- Connections in DAG (Directed Acyclic Graph)
- Visual control (order and dependencies)

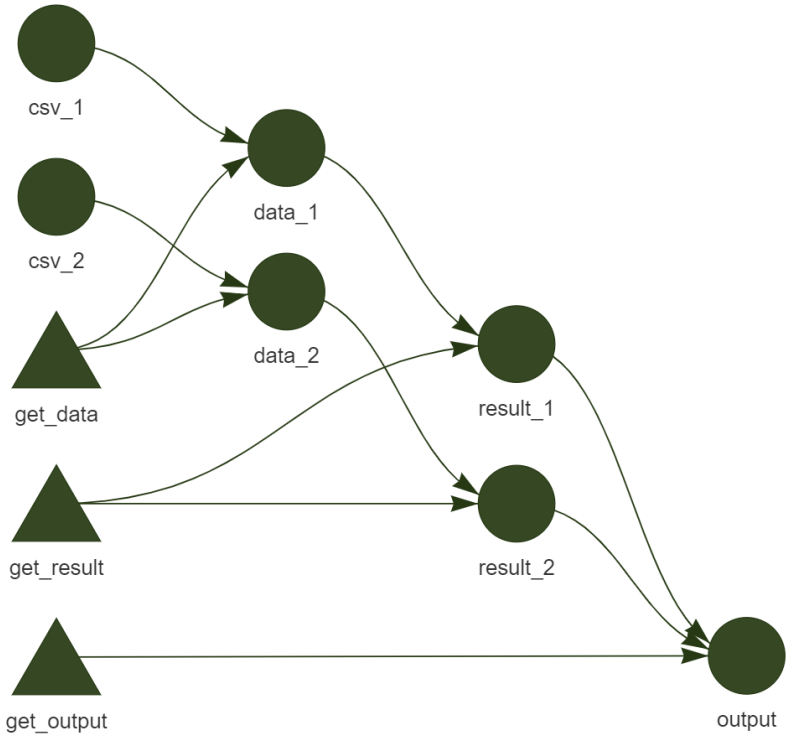
Advantages I: Stability/Correctness



- Connections in DAG (Directed Acyclic Graph)
- Visual control (order and dependencies)

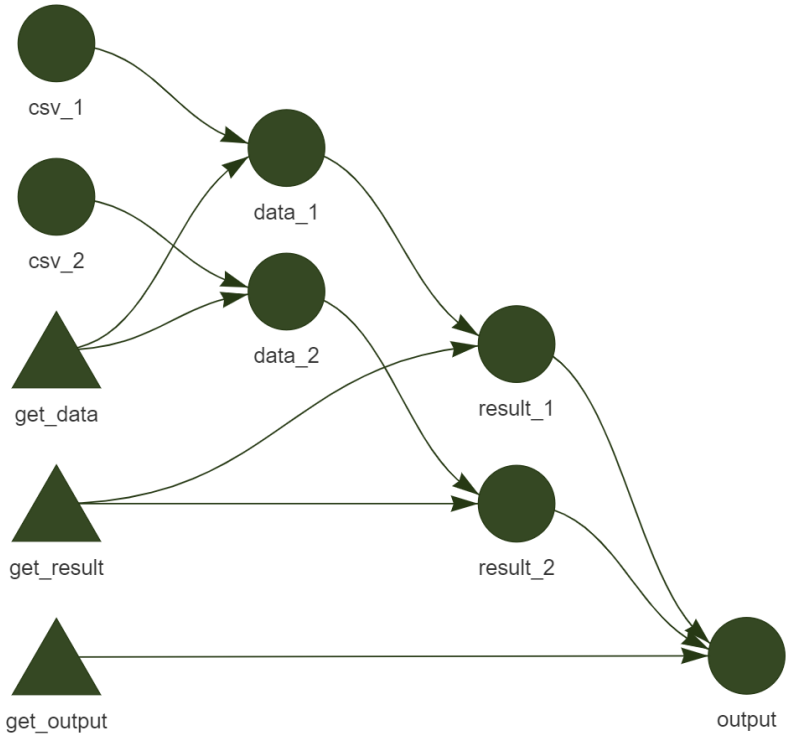
to start workflow: `tar_make()`

Advantages I: Stability/Correctness



- Connections in DAG (Directed Acyclic Graph)
- Visual control (order and dependencies)
- Everything up to date!

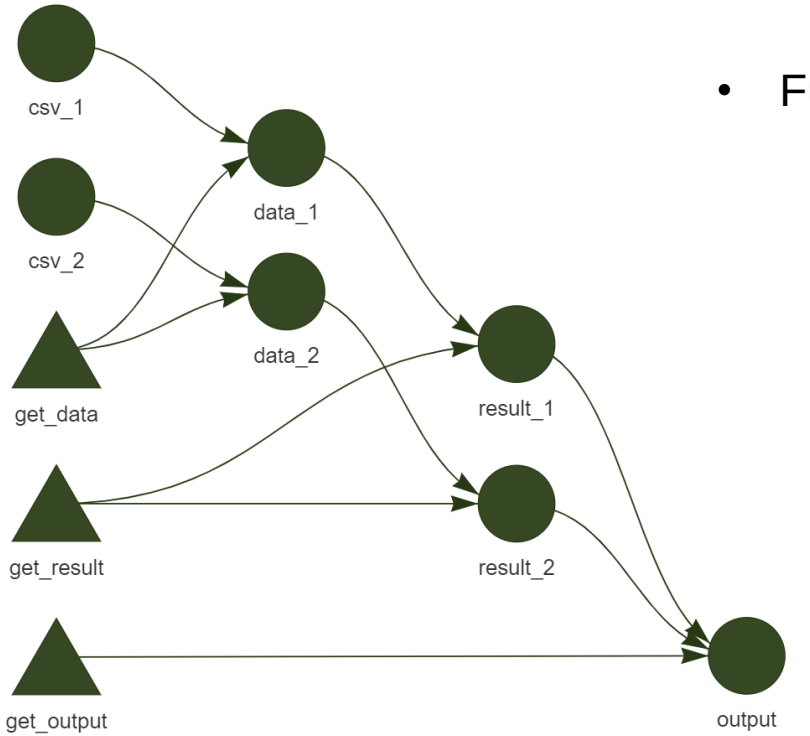
Advantages I: Stability/Correctness



- Connections in DAG (Directed Acyclic Graph)
- Visual control (order and dependencies)
- Everything up to date!

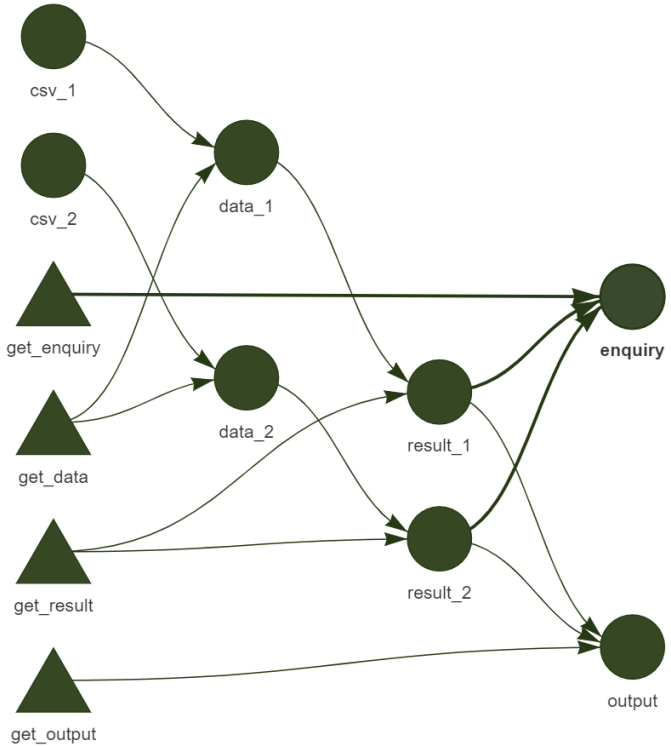
Increases transparency how results were generated

Advantages II: Reproducibility/Recyclability



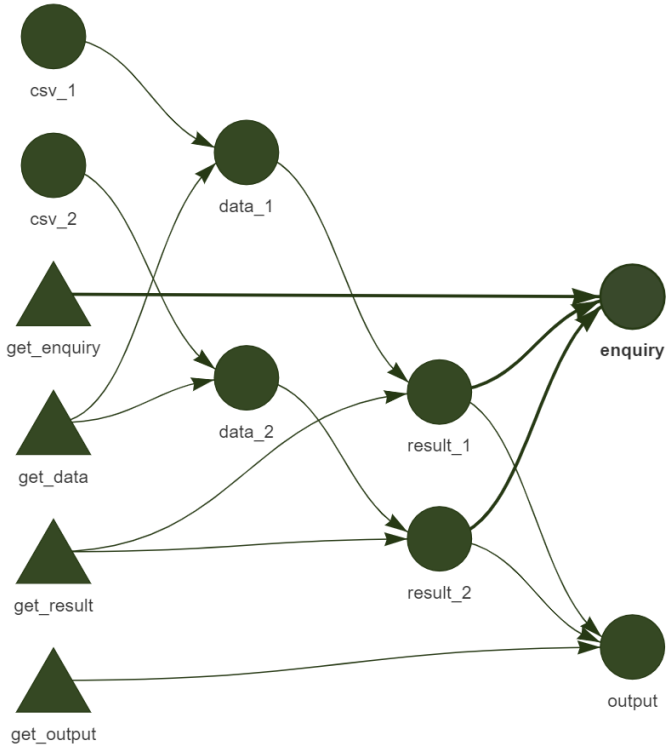
- Functions increase recyclability

Advantages II: Reproducibility/Recyclability



- Functions increase recyclability
- Enquiry/Extra analysis: start from normal workflow (use a target as starting point)

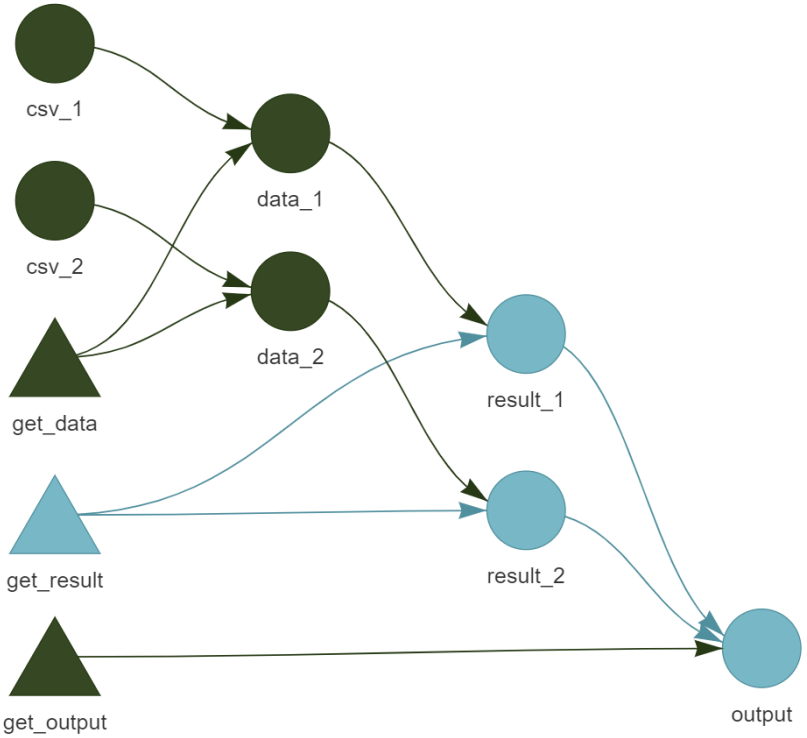
Advantages II: Reproducibility/Recyclability



- Functions increase recyclability
- Enquiry/Extra analysis: start from normal workflow (use a target as starting point)
- Transport project to next year (yearly statistics)
- Workflow logic (run project with little prior knowledge)

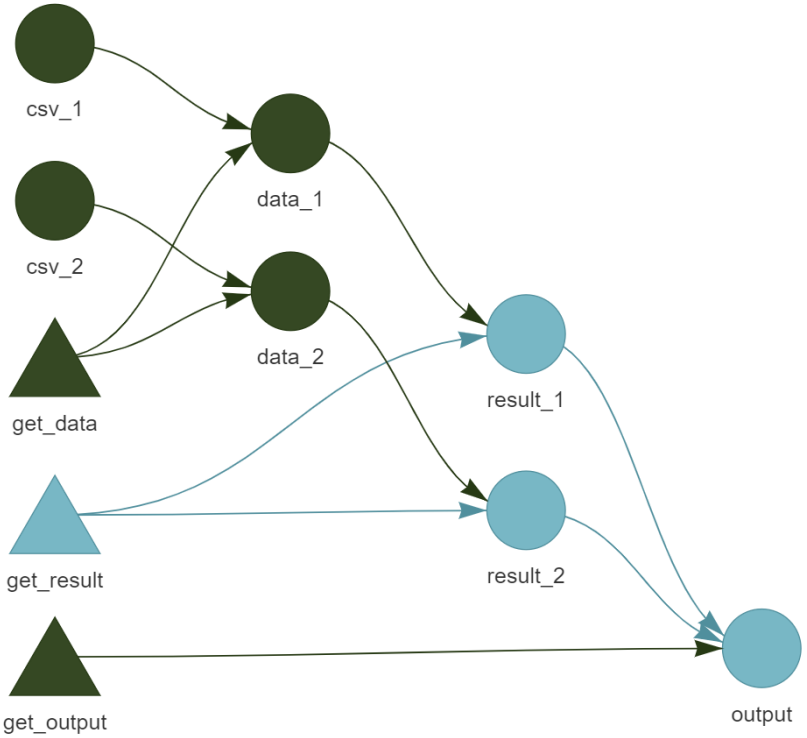
=> `tar_make()`

Advantages III: Speed



- Uses Caching
- Only rerun necessary parts

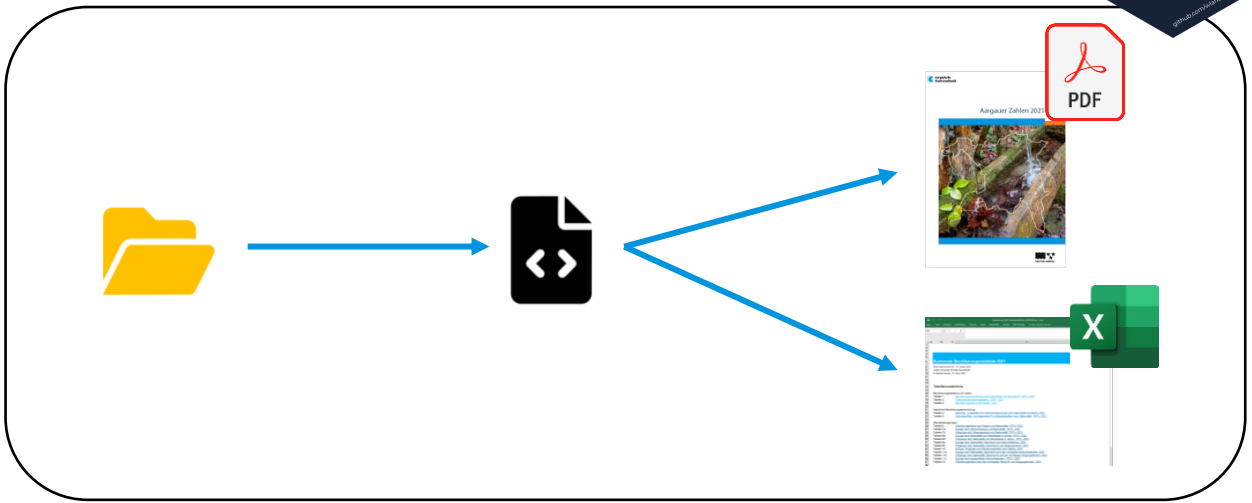
Advantages III: Speed



- Uses Caching
- Only rerun necessary parts
- Parallelisation possible
- Ideal for large data sets like Structural Survey or STATPOP (Machine Learning)

Advantage IV: Automation (Outlook)

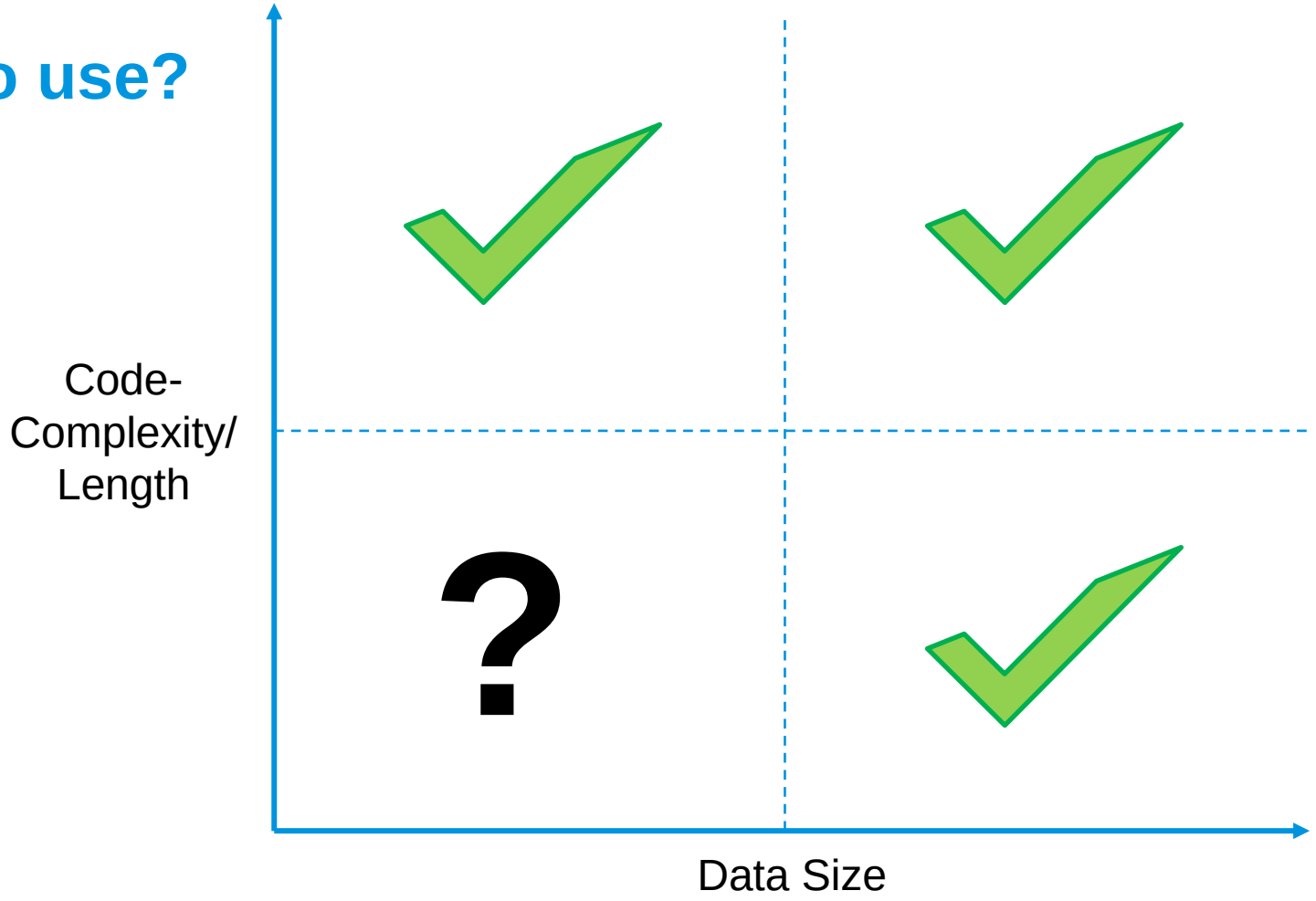
- {targets} as building block
- Controls structure and dependencies
- Helps to ensure correctness
- Workflow with multiple inputs/outputs



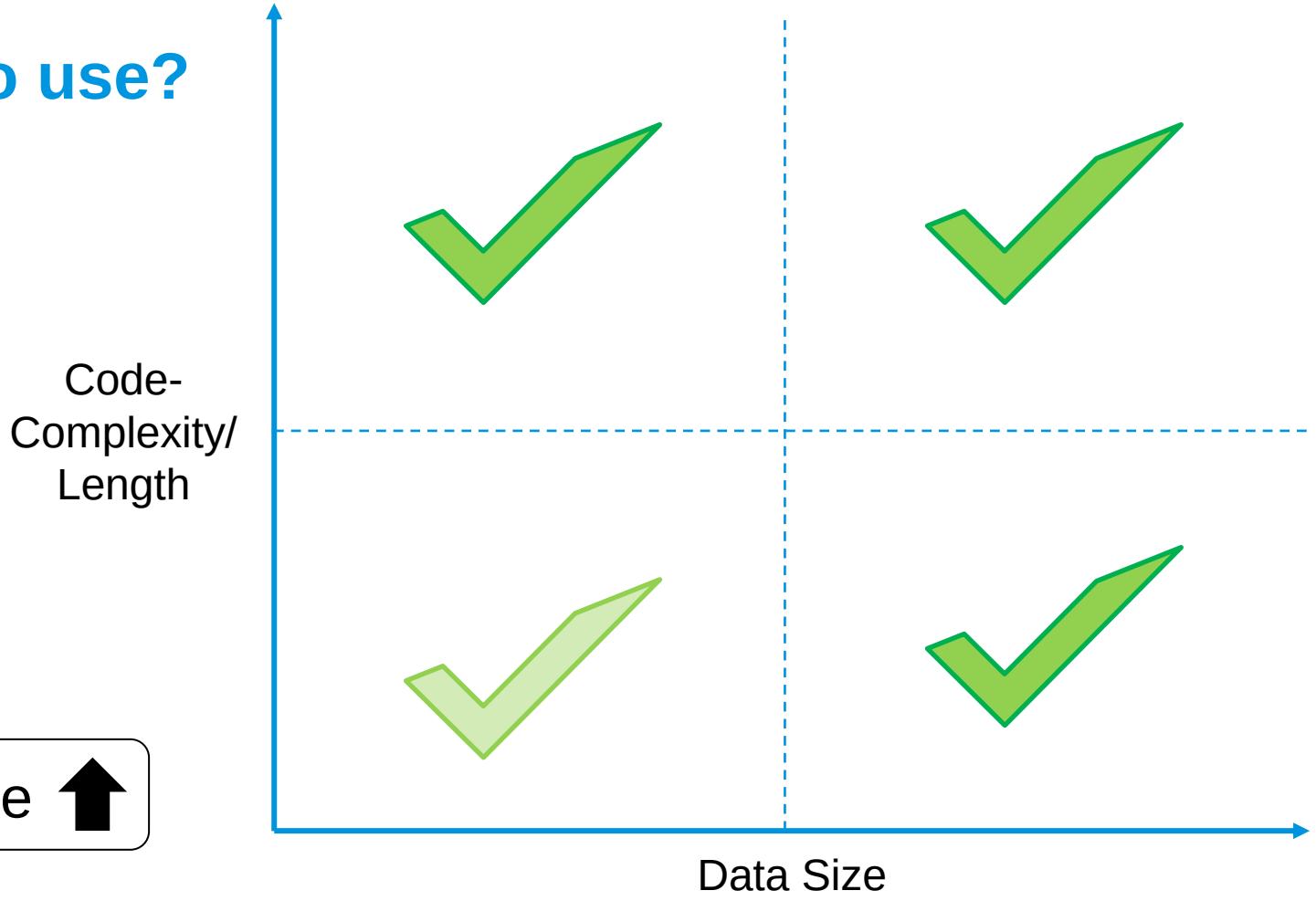
Challenges

- Relatively steep learning curve
- Requires function-oriented programming style
- Debugging complicated
- Keep an eye on the cache (sensitive data)

When to use?



When to use?



Experience ⬆

Conclusion

- Powerful tool
- Stability/reproducibility + speed
- Understand and overview dependencies
- Ideal for yearly projects

Conclusion

- Powerful tool
- Stability/reproducibility + speed
- Understand and overview dependencies
- Ideal for yearly projects
- But: relatively steep learning curve

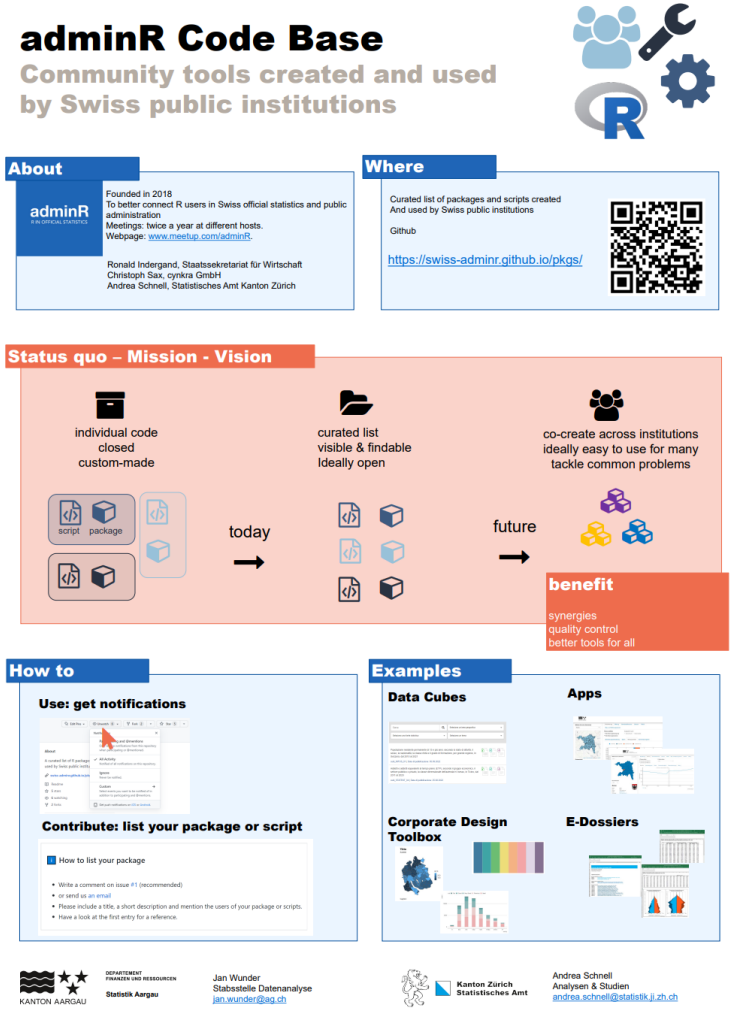
How to start

- {targets} User manual
- {targets} Reference (rOpenSci)
- {targets} Development repository

How to start

- {targets} User manual
- {targets} Reference (rOpenSci)
- {targets} Development repository
- https://swiss-adminr.github.io/pkgs/

  Poster@SST2022 (Wunder & Schnell)



Further questions?

- Please reach out:

niklas.haffert@ag.ch

jan.wunder@ag.ch

Thank you!

Team Statistik Aargau

cynkra GmbH

Questions?