Utrecht University

Methodology & Statistics

Teaching Reproducibility to Research Master's Students

What resources to use? And which tools/elements are domain agnostic?

HI Oberman

Our course

We teach ‘Markup Languages and Reproducible Programming in Statistics’ in the second year of the Master’s programme ‘Methodology and Statistics for the Behavioural, Biomedical and Social Sciences’ at Utrecht University (UU).[1, 2]

Course aims include the development of a publication-ready **reproducible research compendium** that contains:

- a typeset manuscript following a markup language,
- data and code,
- everything that allows for successful reproduction and reuse of the materials (e.g. a license).

In our course, students are taught various tools and languages, such as *Quarto* markdown, version control with *git*, and reproducible environments for R with *renv*.

Reusable course elements

Our course materials draw from resources about the tools and languages that we teach, e.g. software manuals. In principle, these materials are reusable by others who teach reproducibility. But, of course, this is heavily reliant on the specific software framework: in our case, the statistical programming language R.

A more domain agnostic resource that we use is **The Turing Way**. The Turing Way is a handbook for reproducible, ethical and collaborative data science, developed by the Turing Way community.[3] We also use teaching materials from the research data management support workshops offered by the UU library.[4, 5] But… we might be missing some discipline-independent resources for teaching and testing reproducibility for graduate students.

Missing element(s)?

We do not yet make use of many domain agnostic initiatives for reproducible research. Just to name a few relevant collections: FORRT, Project TIER, and the Carpentries. But more importantly, we could implement **CODECHECK-**ing in our course to teach about computational reproducibility.

A CODECHECK is an attempted reproduction of research outcomes by independent execution of the analysis code. The CODECHECK initiative appoints codecheckers to CODECHECK (peer-reviewed) research articles.[6]

In our course, we could teach students how to do a CODECHECK. We could even use CODECHECK-ing as an assessment tool to evaluate whether students meet the learning objectives. Should we?

Figure 1

Elements in the UU course ‘Markup Languages and Reproducible Programming in Statistics’. What do you think is missing? What could you reuse?

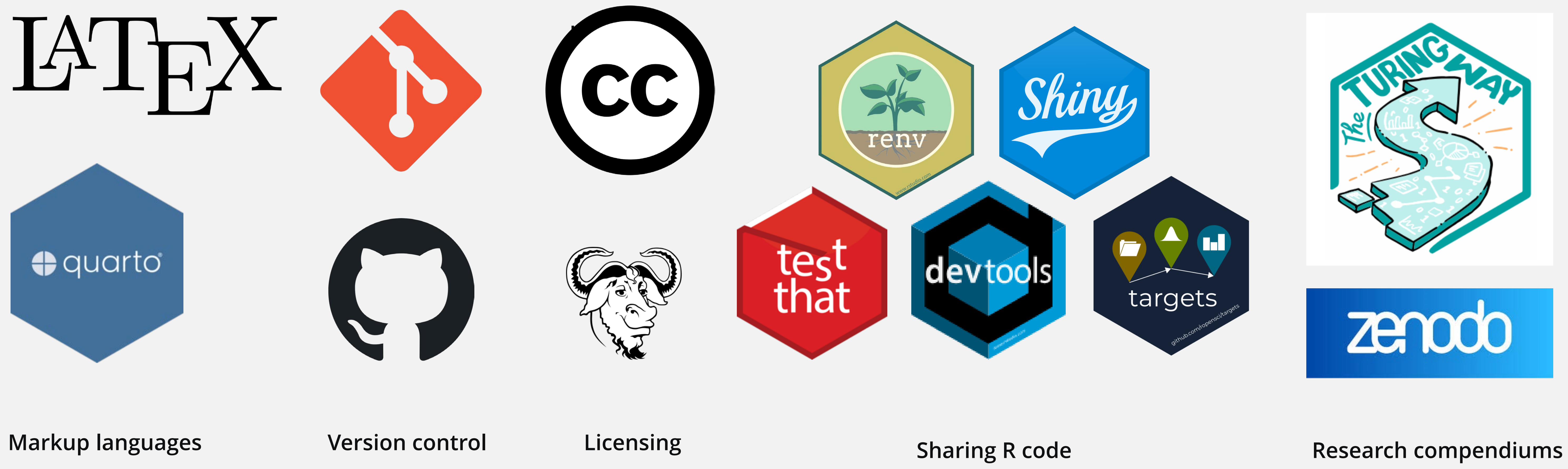

Figure 2

Resources not yet utilized in our course. Should they?

The CODECHECK principles

1. CODECHECKERS RECORD BUT DON’T INVESTIGATE OR FIX.
2. COMMUNICATION BETWEEN HUMANS IS KEY.
3. CREDIT IS GIVEN TO CODECHECKERS.
4. WORKFLOWS MUST BE AUDITABLE.
5. OPEN BY DEFAULT AND TRANSITIONAL BY DISPOSITION.

References

1. Markup Languages and Reproducible Programming in Statistics team (2024). Course materials. URL: www.gerkovink.com/markup
2. Utrecht University (2024). Course description. URL: https://osiris-student.uu.nl/#/onderwijscatalogus/extern/cursus?cursuscode=202000010&taal=en&collegejaar=huidig
3. The Turing Way Community (2022). The Turing Way: A handbook for reproducible, ethical and collaborative research (1.0.2). DOI: 10.5281/ZENODO.3233853

References

4. Utrecht University (2023). Best Practices for Writing Reproducible Code. URL: utrechtuniversity.github.io/workshop-computational-reproducibility
5. Utrecht University (2023). Writing Reproducible Manuscripts in R & Python. URL: utrechtuniversity.github.io/workshop-reproducible-manuscripts
6. Eglen, S., & Nüst, D., (2024). CODECHECK. URL: codecheck.org.uk

edu.nl/7nh73

hanneoberman.github.io/presentations