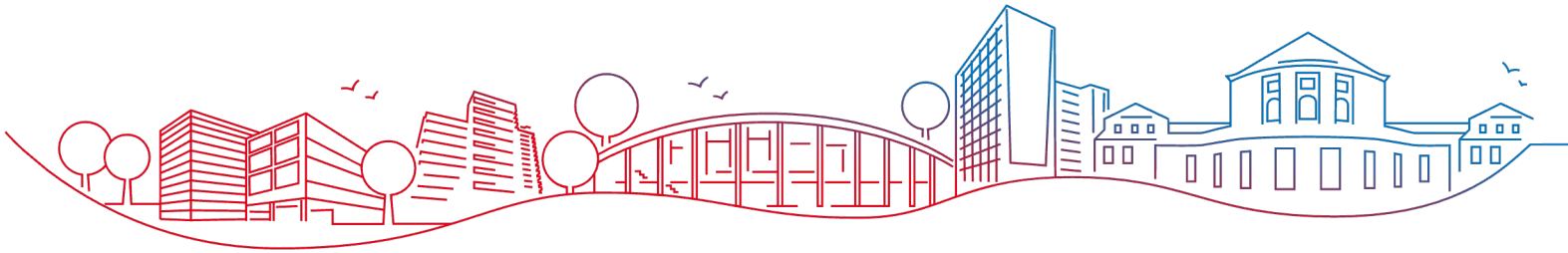




Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG



# Pitfalls of Reproducibility

09.10.2024

**David Seseke & Daniel Gotthardt**

# Questions

- Who uses Python or R?
- Who uses Virtual Environments in their development?
- Who uses Docker for development?

# Motivation

- Well-known replication problems in *traditional* quantitative social sciences (Open Science Collaboration 2015 et al.)
- Computational modeling could circumvent theory crisis (Oberauer & Levandowsky 2019, Guest & Martin 2021)
- A general lack of computational sociology methodology (Hox 2017)
- Our own experiences in research and teaching of semantic and network analysis in contrast to replication courses in statistics

*Disclaimer: We focus on R and Python as most common programming language.*

# Computational Irreproducibility in Social Sciences

- Reinhart & Rogoff's (2010) study on the impact of debt on growth prime example of irreproducibility partially through **under-documented decisions in spreadsheet analysis** (Herndon et al. 2014)
- Breznau et al. (2022) show **striking differences in numerical results** of reproductions of a given study by various research groups
  - a **garden of forking paths** (Gelman & Loken 2013)
- **Transparency and reproducibility** research practices
  - not adopted widely (Hardwicke et al. 2020)
  - weakly institutionalized (Freese et al. 2022, Gayle & Conelly 2022)

Get the Right Tool for the Job!



Friends Don't Let Friends  
Use Excel for Statistics!

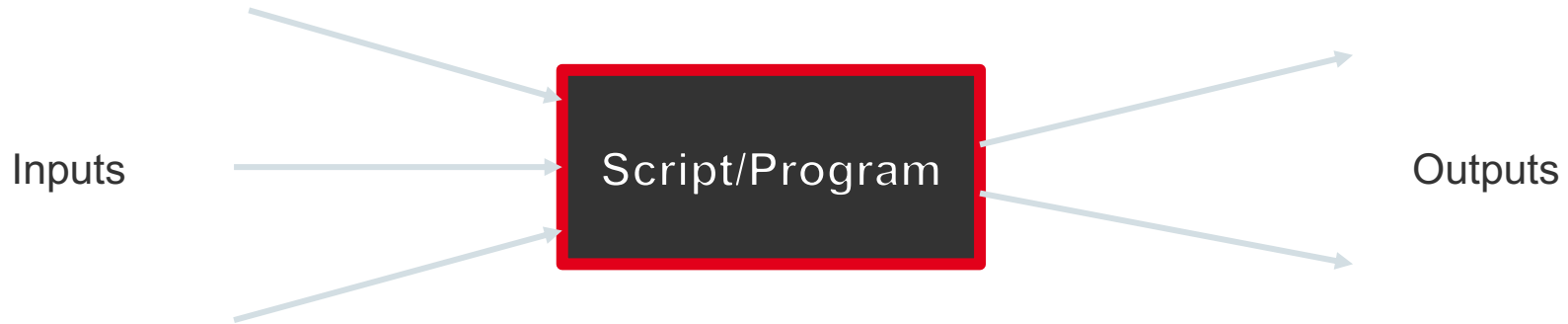
# Reproducibility in Computational Sociology

- Computational reproducibility is the *ability of others to verify the results of a study by repeating the same procedures on the same data* (Stodden et al. 2016, Freese & Peterson 2017)
- Previous recommendations focus on
  - reporting (Rahmandad & Stermann 2012),
  - sharing practices (Stodden et al. 2016),
  - calls for containerization (Liu & Salganik (2019)

# New Threats

- **Complex tools** (e.g., high-performance clusters, large analysis chains) generally introduce **deep variability**
  - can lead to **strong dependency on hardware/software environments** (Acher et al. 2024)
- **Random** but also **chaotic** deterministic **system models** susceptible to floating type rounding errors
  - can lead to **statistically distinguishable simulations**, see EC-Earth3 earth system model (Massonnet et al. 2020)
- **Machine learning** models could create new threats to reproducibility like “**data leakage**” (Gibney 2022)

# Eliminate Randomness in Your Process



- Randomness could happen **inside or outside the box**
  - Setting independent random number streams e.g. in Monte Carlo simulations (Morris et al. 2019) in parallel computing more involved (Hörmann et al. 2004)
- Difficulties:
  - Combining multiple package managers
  - Non existing-version pinning

# What do we want to ensure for the environment?

- Reproducibility: Availability of dependencies at runtime
- Portability: Your research team likes to have a look
- Unambiguity: Correct order of execution of scripts

*Implied: Your code (not your environment) adheres to the principles of Clean Code.*



# Learning from DevOps

1. Documentation for the installation process
2. Versioning (ensuring a history of process)
3. Package Management
4. Virtual Environments
5. Purely functional package management

Steps are cumulative. Minimum is level 3.

# Why no Docker?

- Likely better than your bash script
- Requires deliberation to make somehow reproducible (cf. Boettiger 2015)
- Not a tool for reproducibility, but a tool for portability and containerization
- Restrictive institutional administration policies

```

##### Begin with a base image containing a specific version of R #####
FROM r-ver:3.4.2
##### Copy the code files from the local file system into the Docker image #####
COPY . /stanescu
WORKDIR /stanescu
##### Install system dependencies #####
RUN apt-get update
RUN apt-get install -y libcurl4-openssl-dev=7.58.0-2
RUN apt-get install -y libssl-dev=1.1.0h-2

### Install the required R packages ###
RUN Rscript -e "install.packages('readr', dependencies=TRUE, repos='http://cran.rstudio.com')"
RUN Rscript -e "install.packages('https://cran.r-project.org/src/contrib/Archive/readr/readr_1.1.1.1.tar.gz')"

RUN Rscript -e "install.packages('Amelia', dependencies=TRUE, repos='http://cran.rstudio.com')"
RUN Rscript -e "install.packages('https://cran.r-project.org/src/contrib/Archive/Amelia/Amelia_1.7.4.tar.gz')"

RUN Rscript -e "install.packages('dplyr', dependencies=TRUE, repos='http://cran.rstudio.com')"
RUN Rscript -e "install.packages('https://cran.r-project.org/src/contrib/Archive/dplyr/dplyr_0.7.4.tar.gz')"

RUN Rscript -e "install.packages('tidyr', dependencies=TRUE, repos='http://cran.rstudio.com')"
RUN Rscript -e "install.packages('https://cran.r-project.org/src/contrib/Archive/tidyr/tidyr_0.8.0.tar.gz')"

RUN Rscript -e "install.packages('glmnet', dependencies=TRUE, repos='http://cran.rstudio.com')"
# If 2.0-16 is no longer the latest version of 'glmnet', uncomment the line below
# RUN Rscript -e "install.packages('https://cran.r-project.org/src/contrib/Archive/glmnet/glmnet_0.8.0.tar.gz')"

### Downgrade dependencies installed above to the specified version ###
RUN Rscript -e "install.packages('https://cran.r-project.org/src/contrib/Archive/Matrix/Matrix_1.2-11.tar.gz')"
RUN Rscript -e "install.packages('https://cran.r-project.org/src/contrib/Archive/Rcpp/Rcpp_0.12.16.tar.gz')"
# If 1.4.4 is no longer the latest version of 'foreach', uncomment the line below:
# RUN Rscript -e "install.packages('https://cran.r-project.org/src/contrib/Archive/foreach/foreach_1.4.4.tar.gz')"

##### Instruct Docker to start a shell process when an instance of the image is launched #####
CMD ["sh"]

```

# Purely Functional Package Management

- Available package manager: Guix/Nix
- Every dependency is explicitly defined and built
- **Determinism** through knowledge of all previous instructions
- **Pure evaluation:** Nothing besides the existing instructions influences the outcome



+



Environment  
definition

Script

Images, tables,  
reports



# Take aways

Tech. Knowledge/Time	Short (1-5 years)	Long (>5 years)
Low	Docs, Package Management and version pinning	?
High	{v,r}env/Docker	Guix/Nix

*Evaluate your own time and capacities regarding what is feasible and sensible!*

# Stumbling stones

- Ensuring reproducibility across a longer timespan is a non-trivial problem
- Non-deterministic code and long-term usage
- Computational literacy in Sociology
- Nix on MacOS & Linux well supported, less so on Windows
- Still no easy solution in sight:
  - HPCs are still clunky (see Bzeznik et al., 2017; Goswami et al., 2022)
  - “Packaging hell”

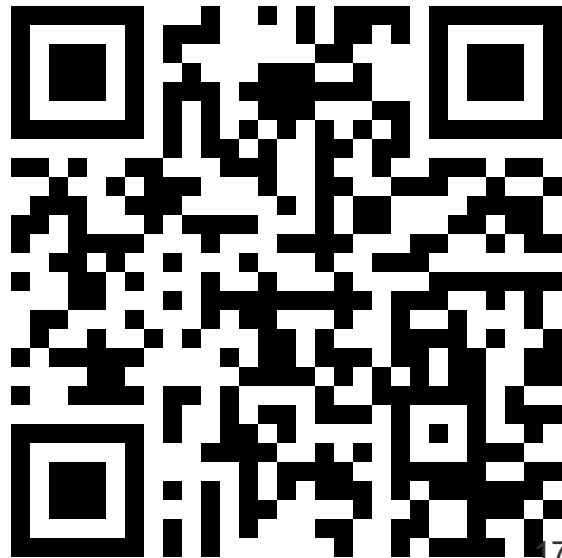






# Further links

- Example in presentation: [https://gitlab.rrz.uni-hamburg.de/bax8491/nix\\_r](https://gitlab.rrz.uni-hamburg.de/bax8491/nix_r)
- Guix and Reproducibility: <https://hpc.guix.info/blog/2023/06/a-guide-to-reproducible-research-papers/>
- Guix on HPC: Courtès & Wurmus (2015), Vallet et al. (2022)
- Nix on HPC: Bzeznik et al. (2017), Goswami et al. (2022)



# References

- Acher, M., Combemale, B., Randrianaina, G. A., & Jezequel, J.-M. (2024). Embracing Deep Variability For Reproducibility and Replicability. *Proceedings of the 2nd ACM Conference on Reproducibility and Replicability*, 30–35. <https://doi.org/10.1145/3641525.3663621>
- Bauer, G., Breznau, N., Gereke, J., Höffler, J. H., Janz, N., Rahal, R.-M., Rennstich, J. K., & Soiné, H. (2023). Teaching Constructive Replications in the Behavioral and Social Sciences Using Quantitative Data. *Teaching of Psychology*, Online first. <https://doi.org/10.1177/00986283231219503>
- Breznau, N., Rinke, E. M., Wuttke, A., Nguyen, H. H. V., Adem, M., Adriaans, J., Alvarez-Benjumea, A., Andersen, H. K., Auer, D., Azevedo, F., Bahnsen, O., Balzer, D., Bauer, G., Bauer, P. C., Baumann, M., Baute, S., Benoit, V., Bernauer, J., Berning, C., ... Zóltak, T. (2022). Observing many researchers using the same data and hypothesis reveals a hidden universe of uncertainty. *Proceedings of the National Academy of Sciences*, 119(44), e2203150119. <https://doi.org/10.1073/pnas.2203150119>
- Boettiger, C. (2015). An introduction to Docker for reproducible research, with examples from the R environment. *ACM SIGOPS Operating Systems Review*, 49(1), 71–79. <https://doi.org/10.1145/2723872.2723882>
- Bzeznik, B., Henriot, O., Reis, V., Richard, O., & Tavard, L. (2017). Nix as HPC package management system. Proceedings of the Fourth International Workshop on HPC User Support Tools, 1–6. <https://doi.org/10.1145/3152493.3152556>
- Courtès, L., & Wurmus, R. (2015). Reproducible and User-Controlled Software Environments in HPC with Guix. In S. Hunold, A. Costan, D. Giménez, A. Iosup, L. Ricci, M. E. Gómez Requena, V. Scarano, A. L. Varbanescu, S. L. Scott, S. Lankes, J. Weidendorfer, & M. Alexander (Eds.), *Euro-Par 2015: Parallel Processing Workshops* (pp. 579–591). Springer International Publishing. [https://doi.org/10.1007/978-3-319-27308-2\\_47](https://doi.org/10.1007/978-3-319-27308-2_47)
- Freese, J. (2007). Replication Standards for Quantitative Social Science: Why Not Sociology? *Sociological Methods & Research*, 36(2), 153–172. <https://doi.org/10.1177/0049124107306659>
- Freese, J., & Peterson, D. (2017). Replication in Social Science. *Annual Review of Sociology*, 43, 147–165. <https://doi.org/10.1146/annurev-soc-060116-053450>
- Freese, J., Rauf, T., & Voelkel, J. G. (2022). Advances in transparency and reproducibility in the social sciences. *Social Science Research*, 107, 102770. <https://doi.org/10.1016/j.ssresearch.2022.102770>
- Gayle, V., & Connelly, R. (2022). The Stark realities of reproducible statistically orientated sociological research: Some newer rules of the sociological method. *Methodological Innovations*, 15(3), 207–221. <https://doi.org/10.1177/2059799122111681>
- Guest, O., & Martin, A. E. (2021). How Computational Modeling Can Force Theory Building in Psychological Science. *Perspectives on Psychological Science*, 16(4), 789–802. <https://doi.org/10.1177/1745691620970585>
- Goswami, R., S., R., Goswami, A., Goswami, S., & Goswami, D. (2022). Reproducible High Performance Computing without Redundancy with Nix. 2022 Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC), 238–242. <https://doi.org/10.1109/PDGC56933.2022.10053342>
- Hardwicke, T. E., Wallach, J. D., Kidwell, M. C., Bendixen, T., Crüwell, S., & Ioannidis, J. P. A. (2020). An empirical assessment of transparency and reproducibility-related research practices in the social sciences (2014–2017). *Royal Society Open Science*, 7(2), 190806. <https://doi.org/10.1098/rsos.190806>
- Herndon, T., Ash, M., & Pollin, R. (2014). Does high public debt consistently stifle economic growth? A critique of Reinhart and Rogoff. *Cambridge Journal of Economics*, 38(2), 257–279. <https://doi.org/10.1093/cje/bet075>
- Hox, J. J. (2017). Computational Social Science Methodology, Anyone? *Methodology*, 13(Supplement 1), 3–12. <https://doi.org/10.1027/1614-2241/a000127>
- Liu, D. M., & Salganik, M. J. (2019). Successes and Struggles with Computational Reproducibility: Lessons from the Fragile Families Challenge. *Socius*, 5, 2378023119849803. <https://doi.org/10.1177/2378023119849803>
- Oberauer, K., & Lewandowsky, S. (2019). Addressing the theory crisis in psychology. *Psychonomic Bulletin & Review*, 26(5), 1596–1618. <https://doi.org/10.3758/s13423-019-01645-2>
- Rahmandad, H., & Sterman, J. D. (2012). Reporting guidelines for simulation-based research in social sciences. *MIT Web Domain*. <https://dspace.mit.edu/handle/1721.1/76300>
- Reinhart, C. M., & Rogoff, K. S. (2010). Growth in a Time of Debt. *American Economic Review*, 100(2), 573–578. <https://doi.org/10.1257/aer.100.2.573>
- Stodden, V., McNutt, M., Bailey, D. H., Deelman, E., Gil, Y., Hanson, B., Heroux, M. A., Ioannidis, J. P. A., & Tauber, M. (2016). Enhancing reproducibility for computational methods. *Science*, 354(6317), 1240–1241. <https://doi.org/10.1126/science.aah6168>

# Kontakt



**Daniel Gotthardt & David Seseke**

Soziologie  
Universität Hamburg  
Max-Brauer-Allee 60  
22765 Hamburg

[daniel.gotthardt@uni-hamburg.de](mailto:daniel.gotthardt@uni-hamburg.de)  
[david.seseke@uni-hamburg.de](mailto:david.seseke@uni-hamburg.de)

```

{
  description = "Initial draft for flake-based Nix&R deployments.";

  inputs = {
    flake-utils.url = "github:numtide/flake-utils";
    nixpkgs.url = "github:NixOS/nixpkgs/nixos-unstable";
  };

  outputs = { self, nixpkgs, flake-utils }:
    flake-utils.lib.eachDefaultSystem (system:
      let
        pname = "runCalc";
        pkgs = nixpkgs.legacyPackages.${system};
        rsienaMod = pkgs.rPackages.buildRPackage {
          name = "rsiena";
          src = pkgs.fetchFromGitHub{
            owner = "Kaladani";
            repo = "rsienafork";
            rev = "37880171c1583f51765c10f0ff6885268103155d";
            hash = "sha256-w24qqI1ujTJHOOEmswW/b5dRG3dzwgyVXjD93LsuZTI=";
          };
        };
        propagatedBuildInputs = with pkgs.rPackages; [ Matrix lattice MASS xtable ];
      };
      R-with-custom-packages = pkgs.rWrapper.override{ packages = with pkgs; [ R rsienaMod ]; };
      customBuildInputs = [
        R-with-custom-packages
      ];
      name = "run-calc";
      rRunCalc = (pkgs.writeScriptBin name (builtins.readFile ./test_rsiena_installation.R));

    in rec {
      defaultPackage = packages.${pname};
      packages = {
        RSiena = rsienaMod;
        ${pname} = pkgs.symlinkJoin {
          inherit name;
          paths = [ rRunCalc ] ++ customBuildInputs;
          buildInputs = [ pkgs.makeWrapper ];
          postBuild = "wrapProgram $out/bin/${name} --prefix PATH : $out/bin";
        };
      };
      devShells.default = pkgs.mkShell {
        packages = [ rRunCalc rsienaMod ] ++ customBuildInputs;
      };
    });
}

```