peer-reviewed, citable, reproducible demographic insights

Expanding the Lifespan of Software for Demographic Analysis with Containers: An Application of Spatial Sampling

Egor Kotov [1,2], and **Esther Denecke** [1,3]

[1] Max Planck Institute for Demographic Research, [2] Universitat Pompeu Fabra, [3] University of Rostock

Introduction

Software, such as specific R packages, evolve over time, which may prevent older analysis code from working as expected. For example, default values for arguments in a function can change. Therefore, for computational reproducibility, knowing which specific R and package versions were used to run the analysis is crucial. One popular solution in R is to use the `sessionInfo()` command to note down the R and package versions. However, a package may be removed from the official R repository (CRAN) altogether. For example, the package "MortalitySmooth" was removed from CRAN in 2020, and the popular spatial analysis packages "rgdal", "rgeos", and "maptools" were retired in October 2023 (Nowosad, 2023). As a consequence of that, many other dependent packages might have also been removed. Installing a deprecated R package is complicated, and running older R code will become more difficult and time-consuming. A practical solution to maintain reproducibility despite constantly changing software is using containers. Containers provide stable and consistent computational environments (see Figure 1).

This note aims to (i) introduce containers to the demographic community, (ii) demonstrate their use with deprecated R packages, and (iii) to spark a discussion on computational reproducibility in the demographic community.

The Denominator

OPEN ACCESS

Citation:

Kotov, E., & Denecke, E. (2024). Expanding the Lifespan of Software for Demographic Analysis with Containers: An Application of Spatial Sampling. https://doi.org/10.6069/WY8K-D973

Published: 2024-04-17

License: CC-BY 2.0

Template based on LaPreprint for Typst by Rowan Cockett (MIT License).

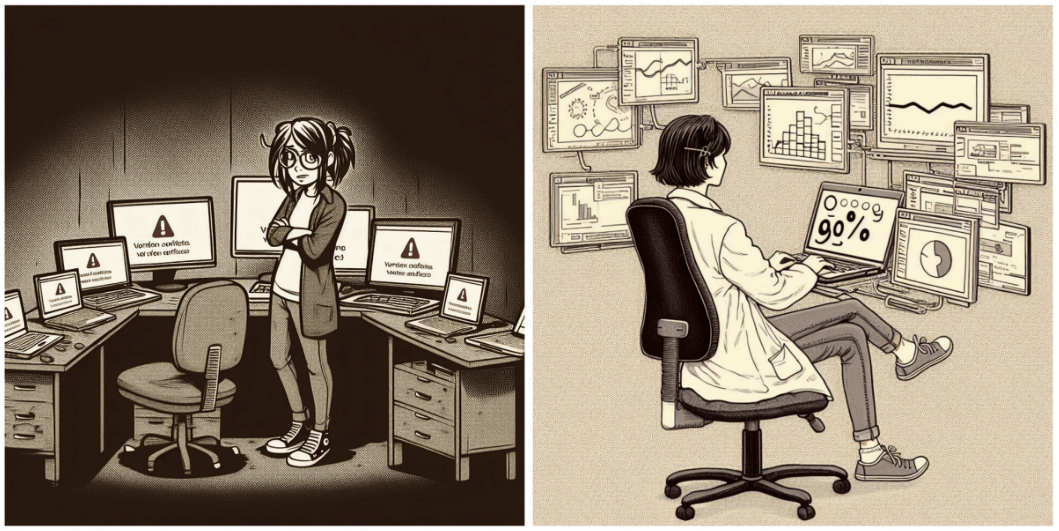


Figure 1: Reproducibility struggles. Left: failing to reproduce the exact software setup on multiple computers. Right: running countless software setups on a single laptop with the help of containers. Image source: Microsoft Image Creator powered by DALL-E 3. Generated with AI · 25 October 2023

What are containers? Why use them?

A container is a file that, similarly to a zip archive, contains other files. Specifically, it includes minimal operating system components and any other software of one's choice, including R, Python, and most other scientific software.

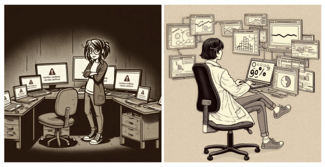
Unlike a zip archive, a container does not need to be unpacked to run the software. Hence, there is no waiting time to unpack or install specific software versions to work on a particular project. From within a container, R starts almost instantly, as if installed on a physical computer, and runs the analysis just as fast.

Any R and RStudio versions can be launched directly from a container on any computer. Any software inside a container is completely independent of the physical computer. Hence, an R version that might already be installed on the physical computer will never conflict with R inside the container.

Software from within a container can run on any popular operating system. Even though R will be running under Linux inside a container, it can be launched on Windows and macOS. One needs to install just one piece of container-running software (discussed in the next section) on their physical computer to independently run any number of different R versions from containers. Furthermore, collaborators can share containers so collaborators can run the same R and package versions without installing those on their physical computers.

Finally, publishing containers along with the analysis code is a good way to make it easier for the reviewers and other researchers to inspect and reproduce submitted work. This approach significantly reduces the barriers to scientific progress, eliminating the

Related Download



Increasing the Lifespan of Software for Demographic Analysis Many researchers face challenges with computational reproducibility. For instance, running analysis code written just a year earlier can be problematic. Even if it worked flawlessly and gave the expected results earlier, it might fail due to errors now. These issues are typically due to the use of newer versions of analysis software... read more

need for time-consuming setup processes and troubleshooting software incompatibilities.

How to use containers? What options are out there?

There are different ways to run containers, but all available approaches are conceptually similar. Essentially, one needs a container with the required versions of R, RStudio and packages and a container-running software installed on a physical computer or in the cloud.

The most popular software for creating, managing, and running containers is Docker (available for Windows, macOS, and Linux). Alternatively, there is Apptainer (previously known as Singularity) for Linux. However, both require installation on a physical computer and at least minimal skills with the terminal.

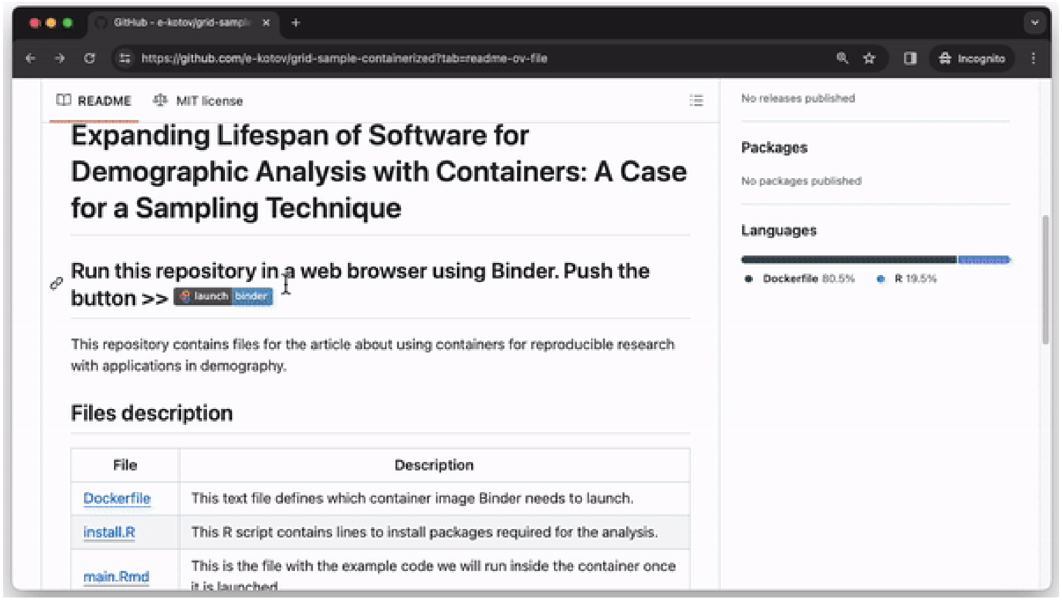
The Binder project provides an approachable way to get started with containers online without installing any software on the local computer. This free website allows one to run code in a container on a temporary cloud server with up to 1-2 GB of operating memory in a web browser for up to 6 hours at a time. After 6 hours, the container can be restarted to continue working. This is not enough for serious work, but enough to get familiar with the containerization technology and see its benefits firsthand. Due to this limitation, there are two primary use cases for Binder:

1. to make the final steps of a reproducible analysis available for inspection and execution by anyone. The "Launch in Binder" button can often be found in GitHub repositories with so-called "reproducible research compendiums": analysis code published alongside academic papers (some examples can be found here: 1, 2, 3). Typically, the code published this way produces only the final plots for the paper due to the memory, computational power, and time limits of mybinder.org. However, the complete, more resource-demanding analysis can still be fully reproducible with the same container and container software installed on a local computer.
2. to run some short, lightweight analyses for free. This is useful for self-education, experiments with containers, and tutorials for classes one might be teaching.

Running an analysis at mybinder.org requires just three steps. First, upload the analysis code and data to a GitHub repository (or a similar code storage website). Second, along with the code, upload a short plain text configuration file for mybinder.org with instructions on which container to use (see the The Download and GitHub repository accompanying this note). Finally, paste the link to the GitHub repository at https://mybinder.org/ or create a "Launch Binder" button in the repository for anyone to run a container with the analysis code in a single click.

Whenever someone clicks the "Launch Binder" launch binder button in the repository, mybinder.org will download the code and instructions from the repository and build and run a container in the cloud. In a few moments, the user can inspect and run the published code in a familiar RStudio interface as if it were running on their computer (see Figure 2).

Animation available in the online version of the article.



Expanding Lifespan of Software for Demographic Analysis with Containers: A Case for a Sampling Technique

Run this repository in a web browser using Binder. Push the button >>

This repository contains files for the article about using containers for reproducible research with applications in demography.

Files description

File	Description
Dockerfile	This text file defines which container image Binder needs to launch.
install.R	This R script contains lines to install packages required for the analysis.
main.Rmd	This is the file with the example code we will run inside the container once it is launched

Figure 2: RStudio running in a web browser using a container on mybinder.org (video speed increased, actual start time of the container may take a little longer.) Analysis with the GridSample R package is performed using the code from the GitHub repository accompanying this manuscript.

Where to get containers, and how to create your own?

For most applications, relying on one of the ready-to-use containers with pre-installed analysis software will be enough. For example, **The Rocker Project** provides various types of containers: basic installation of command-line R, R with RStudio, R with RStudio and Tidyverse packages, the same with geospatial packages, etc. Each container is versioned, meaning a user can request a specific R version. The pre-installed packages in a Rocker container will match this R version by release date. Additional packages not included in Rocker's standard containers can be installed by specifying them in a configuration file or an R script and these will also be installed from a CRAN snapshot available at the time of selected R release. If required, specific package versions can be installed inside the container using a package manager such as **renv** or **pacman**. Another project, similar to The Rocker Project, is **r-minimal** which is maintained by the R Consortium. Alternatively, custom containers with any operating system and software can be created manually from scratch. However, this is an advanced topic that will not be covered here.

To customize one of the pre-built containers for their own purposes (mostly by installing additional packages), the user must create two simple plain text files. These will be the same for local and cloud use, as demonstrated in the example below. Everything else will be handled by the container-running software.

Live demo

In our example, we focus on the R package GridSample (Thomson et al., 2017) to recreate primary sampling units with balanced urban and rural locations (see Figure 3). GridSample was removed from CRAN, thus making its functions unavailable to most researchers. We solve this with a container.

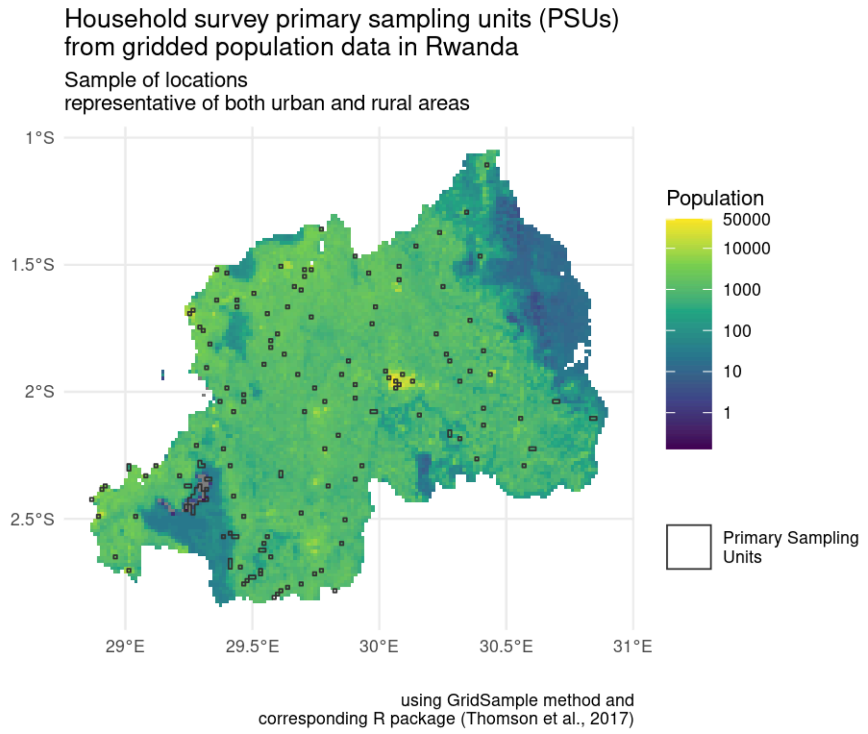

Household survey primary sampling units (PSUs)
from gridded population data in Rwanda
Sample of locations
representative of both urban and rural areas
1°S
1.5°S
2°S
2.5°S
29°E
29.5°E
30°E
30.5°E
31°E
Population
50000
10000
1000
100
10
1
Primary Sampling Units
using GridSample method and
corresponding R package (Thomson et al., 2017)

Figure 3: Result of generating primary sampling units (PSUs) in Rwanda using GridSample package vignette (Thomson et al., 2018) and WorldPop WorldPop (2020) data in a containerized computational environment with R version 4.0.1.

We created a minimal set of files to create and run GridSample from a container in the repository https://github.com/Population-Dynamics-Lab/grid-sample-containerized. We use just 4 lines of code to create a container with R version 4.0.1 released in 2020, the relevant version of RStudio, GridSample and its dependencies. Three of those lines are in a plain text file named **Dockerfile**. The lines:

1. describe which container to use–we use rocker/binder that has the most R packages and is designed to work well with mybinder.org;
2. instruct the container running software to copy all files from the repository inside the container so that a user can run the analysis; and
3. specify an R script file for package installation.

The final fourth line of code is in the R script **install.R**. It provides an R command to install the GridSample package, which is not included in the standard Rocker container.

A **main.Rmd** file contains analysis adapted from the GridSample R package vignette (Thomson et al., 2018) that the user can run after the container is started and RStudio is loaded (see Figure 2). Finally, a **README.md** contains a plain text commentary and code to create the "Launch Binder" button to start the container.

All lines in the text files in our example repository are extensively commented to provide context and the meaning of each command. Using this repository as a template, one can create their own reproducible environments with any version of R running in the cloud using mybinder.org/.

Computation & Reproducibility

All code necessary to implement the methods and reproduce the figures and results in Increasing the Lifespan of Software for Demographic Analysis has been archived as of publication on April 17, 2024 by the Population Dynamics Lab: https://github.com/Population-Dynamics-Lab/grid-sample-containerized.

The original repository maintained by Egor Kotov can be found here: https://github.com/e-kotov/grid-sample-containerized. Note: this repository is maintained by Egor Kotov and may differ from that originally used to produce the results in this publication.

References

Nowosad, J. (2023,). *Upcoming changes to popular R packages for spatial data: What you need to do*. https://geocompx.org/post/2023/rgdal-retirement

Thomson, D. R., Stevens, F. R., Castro, M. C., & Tatem, A. J. (2018,). *GridSample: Tools for Grid-Based Survey Sampling Design. R package version 0.2.2*. https://cran.r-project.org/package=gridsample

Thomson, D. R., Stevens, F. R., Ruktanonchai, N. W., Tatem, A. J., & Castro, M. C. (2017). GridSample: An R package to generate household survey primary sampling units (PSUs) from gridded population data. *International Journal of Health Geographics*, *16*(25), 1–19. https://doi.org/10.1186/s12942-017-0098-4

WorldPop. (2020,). *Global 100m Population total adjusted to match the corresponding UNPD estimate. Rwanda, 2010*. https://doi.org/10.5258/SOTON/WP00660