RANG: RECONSTRUCTING REPRODUCIBLE R COMPUTATIONAL ENVIRONMENTS

A PREPRINT

Chung-hong Chan

GESIS Leibniz-Institut für Sozialwissenschaften

David Schoch

GESIS Leibniz-Institut für Sozialwissenschaften

February 28, 2023

ABSTRACT

A complete declarative description of the computational environment is often missing when researchers share their materials. Without such description, software obsolescence and missing system components can jeopardize computational reproducibility in the future, even when data and computer code are available. The R package rang is a complete solution for generating the declarative description for other researchers to automatically reconstruct the computational environment at a specific time point. The reconstruction process, based on Docker, has been tested for R code as old as 2005 and does not depend on the long term availability of any commercial service. The description generated by rang satisfies the definition of a reproducible research compendium and can be shared as such. In this contribution, we show how rang can be used to make otherwise unexecutable code, spanning from fields such as computational social science and bioinformatics, executable again. We also provide instructions on how to use rang to construct reproducible and shareable research compendium of current research. The package is currently available from CRAN (https://cran.r-project.org/web/packages/rang/index.html) and GitHub (https://github.com/chainsawriot/rang).

Keywords R • reproducibility • docker

1 Introduction

The ability to reconstruct the computational environment is one of the most important aspects of reproducibility. In the realm of R, the important aspects are what and which version of installed R packages as well as the exact R version.

1.1 Existing solutions

`renv` (Ushey 2022) (and its derivatives such as `jetpack` and its predecessor `packrat`) takes a similar approach to Python's `virtualenv` and Ruby's `Gem` to pin down the exact version of R packages using a "lock file". `containerit` (Nüst and Hinz 2019) takes the current state of the computational environment and "containerize" it as a Dockerfile. But `containerit` does not pin down the exact version of R packages. Other solutions such as `groundhog` (Simonsohn and Gruson 2023) and `checkpoint` (Ooi, de Vries, and Microsoft 2022) depend on the availability of The Microsoft R Application Network (MRAN), which will be shut down on July 1st, 2023.

These solutions are better for prospective usage, i.e. using them now to ensure the reproducibility of the current research for future researchers. `rang` mostly targets retrospective usage, i.e. using `rang` to reconstruct historical R computational environments which have not been completely declared. One can think of `rang` as an archaeological tool. In the following examples, `rang` is used to enable the reproducibility of published literature.

2 Basic usage

There are two important functions of `rang`: `resolve()` and `dockerize()`.

`resolve()` queries various web services from the r-hub project of the R Consortium for information about R packages at a specific time point that is necessary for reconstructing a computational environment, e.g. (deep) dependencies, system requirements, and R version. For instance, if there was a computational environment constructed on 2020-01-16 (called "snapshot date") with the several natural language processing R packages, `resolve()` can be used to resolve all the dependencies of these R packages. Currently, `rang` supports CRAN, Bioconductor, and GitHub packages.

```
library(rang)
graph <- resolve(pkgs = c("openNLP", "LDAvis", "topicmodels", "quanteda"),
                 snapshot_date = "2020-01-16")
graph
```

The resolved result is an S3 object called `rang`. The information contained in a `rang` object can then be used to construct a computational environment in a similar manner as `containerit`, but with the packages and R versions pinned on the snapshot date. Then, the function `dockerize()` is used to generate the `Dockerfile` and other scripts in the `output_dir`.

```
dockerize(graph, output_dir = "docker")
```

For R >= 3.1, the images from the Rocker project are used (Boettiger and Eddelbuettel 2017). For R < 3.1 but >= 2.1, a custom image based on Debian Woody is used. As of writing, `rang` does not support R < 2.1, i.e. snapshot date earlier than 2005-04-19. The development to support R < 2.1 is in progress. There are two features of `dockerize()` that are important for future reproducibility.

1. By default, the container building process downloads source packages from CRAN and then compiles them from source. This step depends on the future availability of R packages on CRAN (which is extremely likely to be the case in the near future, given the continuous availability since 1997-04-23) [1]. However, it is also possible to cache (or archive) the source packages now. The archived R packages can then be used instead during the building process.

```
dockerize(graph, output_dir = "docker", cache = TRUE)
```

2. It is also possible to install R packages in a separate library during the building process to isolate all these R packages from the main library.

```
dockerize(graph, output_dir = "docker", cache = TRUE,
          lib = "anotherlibrary")
```

For the sake of completeness, the instructions for building and running the Docker container on Unix-like systems are included here.

```
cd docker
## might need to sudo
docker build -t rang .
docker run --rm --name "rangtest" -ti rang
```

[1] https://stat.ethz.ch/pipermail/r-announce/1997/000001.html

3 Case Studies

The following are some examples of how `rang` can be used to make otherwise shared, but unexecutable, R code runnable again. The examples were drawn from various fields spanning from political science, psychological science, and bioinformatics.

3.1 quanteda JOSS paper

The software paper of the text analysis R package `quanteda` was published on 2018-10-06 (Benoit et al. 2018). In the paper, the following R code snippet is included.

```
library("quanteda")
# construct the feature co-occurrence matrix
examplefcm <-
tokens(data_corpus_irishbudget2010, remove_punct = TRUE) %>%
tokens_tolower() %>%
tokens_remove(stopwords("english"), padding = FALSE) %>%
fcm(context = "window", window = 5, tri = FALSE)
# choose 30 most frequency features
topfeats <- names(topfeatures(examplefcm, 30))
# select the top 30 features only, plot the network
set.seed(100)
textplot_network(fcm_select(examplefcm, topfeats), min_freq = 0.8)
```

On 2023-02-08, this code snippet is not executable with the current version of `quanteda` (3.2.4). It is possible to install the "period appropriate" version of `quanteda` (1.3.4) using `remotes` on the current version of R (4.2.2). And indeed, the above code snippet can still be executed.

```
remotes::install_version("quanteda", version = "1.3.4")
```

The issue is that installing `quanteda` 1.3.4 this way installs the latest dependencies from CRAN. `quanteda` 1.3.4 uses a deprecated (but not yet removed) function of `Matrix` (`as(<dgTMatrix>, "dgCMatrix")`). If this function were removed in the future, the above code snippet would not be executable anymore.

Using `rang`, one can query the version of `quanteda` on 2018-10-06 and create a Docker container with all the "period appropriate" dependencies. Here, the `rstudio` Rocker image is selected.

```
library(rang)
graph <- resolve(pkgs = "quanteda",
                 snapshot_date = "2018-10-06",
                 os = "ubuntu-18.04")
dockerize(graph, output_dir = "quanteda_docker",
          image = "rstudio")
```

The above code snippet can be executed with the generated container without any problem.

3.2 Psychological Science

Crüwell et al. (2023) evaluate the computational reproducibility of 14 articles published in *Psyhocological Science*. Among these articles, the paper by Hilgard et al. (2019) has been rated as having "package dependency issues".

All data and computer code are available from GitHub with the last commit on 2019-01-17 [2]. The R code contains a list of R packages used in the project as `library()` statements, including an R package on GitHub that is written by the main author of that paper. However, we identified one package (`compute.es`) that was not written in those `library()` statements but used with the namespace operator, i.e. `compute.es::tes()`. This undocumented package can be detected by `renv::dependencies()`.

[2] https://github.com/Joe-Hilgard/vvg-2d4d

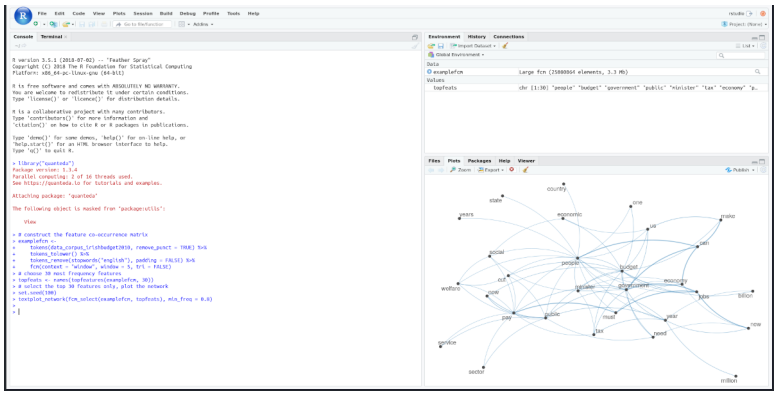

Figure 1: The code snippet running in a R 3.5.1 container created with rang

Based on the above information, one can run `resolve()` to obtain the dependency graph of all R packages on 2019-01-17.

```
graph <- resolve(c("readxl", "rms", "ordinal", "tidyverse", "lubridate", "MBESS",
                   "censReg", "BayesFactor", "psych", "Joe-Hilgard/hilgard", "lsmeans",
                   "compute.es"),
                 snapshot_date = "2019-01-17")
```

When running `dockerize()`, one can take advantage of the `materials_dir` parameter to transfer the shared materials from Hilgard et al. (2019) into the Docker image.

```
dockerize(graph, "hilgard", materials_dir = "vvg-2d4d", cache = TRUE)
```

We then built the Docker and launch a Docker container. For this container, we changed the entry point from R to bash so that the container goes to the Linux command shell instead.

```
cd hilgard
docker build -t hilgard .
docker run --rm --name "hilgardcontainer" --entrypoint bash -ti hilgard
```

Inside the container, the materials are located in the `materials` directory. We used the following shell script to test the reproducibility of all R scripts.

```
cd materials
rfiles=(0_data_aggregation.R 1_data_cleaning.R 2_analysis.R 3_plotting.R)
for i in ${rfiles[@]}
do
    Rscript $i
    code=$?
    if [ $code != 0 ]
    then
        exit 1
    fi
done
```

All R scripts ran fine inside the container and the figures generated are the same as the ones in Hilgard et al. (2019).

3.3 Political Analysis

The study by Trisovic et al. (2022) evaluates the reproducibility of R scripts shared on Dataverse. They found that 75% of R scripts cannot be successfully executed. Among these failed R scripts is an R script shared by Beck (2019).

This R script has been "rescued" by the author of the R package `groundhog` (Simonsohn and Gruson 2023), as demonstrated in a blog post [3]. We were wondering if `rang` can also be used to rescue the concerned R script.

The date of the R script, as indicated on Dataverse, is 2018-12-12. This date is used as the snapshot date.

```
graph <- resolve(c("foreign", "bife"), snapshot_date = "2018-12-12")
dockerize(graph, output_dir = "nat", materials_dir = "nathaniel")
```

```
cd nat
docker build -t nat .
docker run --rm --name "natcontainer" --entrypoint bash -ti nat
```

Inside the container

```
cd materials
Rscript fn_5.R
```

The same file can also be "rescued" by `rang`.

3.4 Recover a removed R package

The R package `maxent` introduces a machine learning algorithm with a small memory footprint and was available on CRAN until 2019. A software paper was published by the original authors in 2012 (Jurka 2012). The R package was also used in some subsequent automated content analytic papers (e.g. Lörcher and Taddicken 2017).

Despite the covert editing of the package by a staffer of CRAN [4], the package was removed from CRAN in 2019 [5]. We attempted to install the second last (the original submitted version) and last (with covert editing) versions of `maxent` on R 4.2.2. Both of them didn't work.

Using `rang`, we are able to reconstruct a computational environment with R 2.15.0 (2012-03-30) to run all code snippets published in Jurka (2012) [6]. For removed CRAN packages, we strongly recommend querying the Github read-only mirror of CRAN instead (https://github.com/cran). It is because in this way, the resolved system requirements have a higher chance of being correct.

```
maxent <- resolve("cran/maxent", "2012-06-10")
dockerize(maxent, "maxentdir", cache = TRUE)
```

3.5 Recover a removed Bioconductor package

4 Generating a research compendium

5 Conclusion

References

Beck, Nathaniel. 2019. "Estimating Grouped Data Models with a Binary-Dependent Variable and Fixed Effects via a Logit Versus a Linear Probability Model: The Impact of Dropped Units." *Political Analysis* 28 (1): 139–45. https://doi.org/10.1017/pan.2019.20.

Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, and Akitaka Matsuo. 2018. "Quanteda: An R Package for the Quantitative Analysis of Textual Data." *Journal of Open Source Software* 3 (30): 774. https://doi.org/10.21105/joss.00774.

[3] http://datacolada.org/100

[4] https://github.com/cran/maxent/commit/9d46c6aad27a1f41a78907b170ddd9a586192be9

[5] https://cran-archive.r-project.org/web/checks/2019/2019-03-05_check_results_maxent.html

[6] On an interesting historical note: The original paper reported, based on a benchmark, that "the algorithm is very fast; `maxent` uses only 135.4 megabytes of RAM and finishes in 53.3 seconds." On a modest computer in 2023 with a dockerized R 2.15.0, the benchmark finishes in 4 seconds.

Boettiger, Carl, and Dirk Eddelbuettel. 2017. "An Introduction to Rocker: Docker Containers for R." *The R Journal* 9 (2): 527. https://doi.org/10.32614/rj-2017-065.

Crüwell, Sophia, Deborah Apthorp, Bradley J. Baker, Lincoln Colling, Malte Elson, Sandra J. Geiger, Sebastian Lobentanzer, et al. 2023. "What's in a Badge? A Computational Reproducibility Investigation of the Open Data Badge Policy in One Issue of Psychological Science." *Psychological Science*, February, 095679762211408. https://doi.org/10.1177/09567976221140828.

Hilgard, Joseph, Christopher R. Engelhardt, Jeffrey N. Rouder, Ines L. Segert, and Bruce D. Bartholow. 2019. "Null Effects of Game Violence, Game Difficulty, and 2D:4D Digit Ratio on Aggressive Behavior." *Psychological Science* 30 (4): 606–16. https://doi.org/10.1177/0956797619829688.

Jurka, P., Timothy. 2012. "Maxent: An r Package for Low-Memory Multinomial Logistic Regression with Support for Semi-Automated Text Classification." *The R Journal* 4 (1): 56. https://doi.org/10.32614/rj-2012-007.

Lörcher, Ines, and Monika Taddicken. 2017. "Discussing Climate Change Online. Topics and Perceptions in Online Climate Change Communication in Different Online Public Arenas." *Journal of Science Communication* 16 (02): A03. https://doi.org/10.22323/2.16020203.

Nüst, Daniel, and Matthias Hinz. 2019. "Containerit: Generating Dockerfiles for Reproducible Research with r." *Journal of Open Source Software* 4 (40): 1603. https://doi.org/10.21105/joss.01603.

Ooi, Hong, Andrie de Vries, and Microsoft. 2022. *checkpoint: Install Packages from Snapshots on the Checkpoint Server for Reproducibility*. https://CRAN.R-project.org/package=checkpoint.

Simonsohn, Uri, and Hugo Gruson. 2023. *groundhog: Version-Control for CRAN, GitHub, and GitLab Packages*. https://CRAN.R-project.org/package=groundhog.

Trisovic, Ana, Matthew K. Lau, Thomas Pasquier, and Mercè Crosas. 2022. "A Large-Scale Study on Research Code Quality and Execution." *Scientific Data* 9 (1). https://doi.org/10.1038/s41597-022-01143-6.

Ushey, Kevin. 2022. *renv: Project Environments*. https://CRAN.R-project.org/package=renv.