RESCIENCE C

Reproduction / Astronomy

[¬Rp] Reproducibility of 'Poincaré dodecahedral space parameter estimates'

Boudewijn F. Roukema[1,2]

[1]Institute of Astronomy, Faculty of Physics, Astronomy and Informatics, Nicolaus Copernicus University, Grudziadzka 5, 87-100 Toruń, Poland – [2]Univ Lyon, Ens de Lyon, Univ Lyon1, CNRS, Centre de Recherche Astrophysique de Lyon UMR5574, F–69007, Lyon, France

Edited by
Pierre de Buyl

Reviewed by
Konrad Hinsen

Received
05 May 2020

Published
14 July 2020

DOI
10.5281/zenodo.3943750

Abstract

Is a scientific research paper based on (i) public, online observational data files and (ii) providing free-licensed software for reproducing its results easy to reproduce by the same author a decade later? This paper attempts to reproduce a cosmic topology observational paper published in 2008 and satisfying both criteria (i) and (ii). The reproduction steps are defined formally in a free-licensed git repository package "0807.4260" and qualitatively in the current paper. It was found that the effort in upgrading the Fortran 77 code at the heart of the software, interfaced with a C front end, and originally compiled with g77, in the content of the contemporary gfortran compiler, risked being too great to be justified on any short time scale. In this sense, the results of RBG08 are not as reproducible as they appeared to be, despite both (i) data availability and (ii) free-licensing and public availability of the software. The software and a script to reproduce the steps of this incomplete reproduction are combined in a new git repository named 0807.4260, following the ArXiv identity code of RBG08.[1]

1 Introduction

This paper studies the reproducibility of the main observational results of a cosmic topology research paper published by myself and co-authors in 2008[1]. The paper used the surface-of-last-scattering optimal cross-correlation method of finding a preferred orientation of the fundamental domain of the spatial section of the Universe, under the working hypothesis that the spatial section is a Poincaré dodecahedral space[2]. The code was developed by me, with comments provided by my coauthors. The results that should be reproduced are those that use the method described in Section 3.2 of RBG08, and the observational analysis results described in Section 4.2, displayed in Figs. 3, 4, 5 and given numerically in Tables 2 and 3 of RBG08. Related cosmic topology papers by other authors are published with no references to software package details or software licences.

The reason for attempting and documenting the reproducibility of this paper is that not only are many papers in astronomy[3] and other fields still published without providing the full empirical data sets and source code under free-software licences, but even those that provide free-licensed software and input data may be difficult to reproduce[4,5,6]. While observational data in cosmology are usually made available online with high-quality documentation, often after an embargo period, free-licensed software in the field of cosmic topology, in particular, library functions for defining

Correspondence should be addressed to Boudewijn F. Roukema (boud astro.uni.torun.pl)
The authors have declared that no competing interests exist.
Code is available at https://codeberg.org/boud/0807.4260. – SWH swh:1:dir:4f1fe8cf5a01bb4637e31ea938eaa5bc25a2b87b.
Data is available at https://lambda.gsfc.nasa.gov/data/map/dr3/dfp/wmap_ilc_5yr_v3.fits.
Open peer review is available at https://github.com/ReScience/submissions/issues/41.

matched circles in the cosmic microwave background or matched *discs* in extragalactic 3-dimensional comoving space[7], is only recorded in the scientific literature in papers published by my research group.

To document and help analyse the success and difficulties in reproducing scientific results in this context, the editors of *ReScience C* posed the "Ten Years Reproducibility Challenge", a request that scientists attempt to reproduce the main results of *their own* peer-reviewed scientific research papers that had been published before 1 January 2010, and document the method and results in *ReScience C*[8].

2 Method

The first steps for trying to reproduce the original results of RBG08 were to (re-)read the appropriate sections of the paper, initially taking the view of a non-author.

1. Section 2.1[1] states that the analysis method of Section 3.2 requires the three files at URLs listed in footnotes 1, 2, 3 on the same page. These files represent two versions of an all-sky map of the Universe mostly representing cosmic microwave background emission at $10h^{-1}$ Gpc (comoving) from the Earth as observed by the Wilkinson Microwave Anisotropy Probe (WMAP)[9], and the "kp2" mask to enable analysis that avoids the most contaminated regions of the sky. These files need to be downloaded.
2. Footnote 7[1] indicates that CIRCLES-0.3.2.1, to be found at the URL http://cosmo.torun.pl/GPLdownload/dodec/, provides the software for generating the figures and tables. This software needs to downloaded from http://cosmo.torun.pl/GPLdownload/dodec/circles-0.3.2.1.tar.gz.

The next step was to develop a script on a GIT repository server that satisfies the requirements of the international scientific community, specifically the International Science Council[10], by not blocking access to scientists of any countries or territories. During 2018 and 2019, several of the most popular GIT repository servers partially blocked access to scientists and other residents of several countries and territories (Github[11,12,13], Bitbucket[14], Gitlab[15]; the GITLAB software is free-licensed and can be installed independently of the Gitlab online service). The bans have presumably continued into 2020. A shift of my own software to servers acceptable under international scientific ethical standards is underway, but incomplete as of early 2020. I chose a community-based server, *Codeberg*, not currently listed on the Wikipedia list of source code hosting facilities[1]. In 2019, the *Investigating & Archiving the Scholarly Git Experience* project team expressed its concerns about the bans, describing them as having "far-reaching and chilling consequences for open source, open scholarship, and for the open exchange of information and ideas"[16].

The remaining planned steps were to implement the minimal number of updates to make the code work and replicate the original results, using modern hardware and a modern software environment. Footnote 7 of RBG08 warns that "These [CIRCLES-0.3.2.1 and CIRCLES-0.3.8] and earlier versions of the software require medium to advanced GNU/LINUX, FORTRAN77 and C experience for a scientific user." There is no statement regarding the particular compiler(s) used. As far as I recall, it's very likely that the widely used GNU fortran compiler of the time, G77, was used together with GCC, as selected automatically by AUTOTOOL packages.

The system and hardware chosen for the reproduction project were an AMD computer running with a DEBIAN GNU/LINUX 9.12 system on an X86_64 LINUX-4.9.0 KERNEL. The Fortran compiler chosen was GNU FORTRAN (DEBIAN 6.3.0-18+DEB9U1) 6.3.0 20170516.

3 Results

The overall script intended to carry out the full sequence of downloads, configuring of packages, compiling of packages, subdirectory user-level installation of packages,

[1] https://en.wikipedia.org/wiki/Comparison_of_source-code-hosting_facilities

ded213c1c4cfdfe2ef92f7155b27d58c	wmap_ilc_5yr_v3.fits
fbc8b2518fdddf0a1e7b5acde99a748e	wiener5yr_map.fits
5aa3267dc6d69bf8c5f0a3a893e23960	wmap_kp2_r9_mask_3yr_v2.fits
afbd67d8120c11e949eb0c414c2775f5	circles-0.3.2.1.tar.gz

Table 1. Checksums (md5sums) of the data and the main software source code files of RBG08, use for the present reproducibility test.

setting up of calculation parameters, and running the main code, was set up as a BASH script `reproduce_RBG08.sh`.

The full package aiming to reproduce the figures and tables listed above is provided at https://codeberg.org/boud/0807.4260, named after the ArXiv identity of RBG08.

3.1 Downloading data and software source code

1. The URL in footnotes 1, 2 and 3[1] gave clickable links that were split into two and not correctly clickable. The user needs to cut/paste the two halves of each URL in order to obtain the three data files. The data files were downloaded with no apparent problem, with md5sums as indicated in Table 1.
2. The file CIRCLES-0.3.2.1.TAR.GZ with the md5sum indicated in Table 1 was downloaded. It was included in the main git repository in its original form. Subsequent changes are recorded in the git history at https://codeberg.org/boud/0807.4260.

3.2 Compiling/debugging

Fixes needed in order to successfully compile CIRCLES include:

1. A Fortran 77 line that ended on one line with a + symbol and started on the next line with another + symbol (within the valid columns for standard Fortran 77) was apparently accepted by the GCC family fortran compiler in 2008, but not now (2020). One of the + symbols was removed.
2. A fitting algorithm GSL_MULTIFIT_COVAR available in GNU Scientific Library (GSL) versions 1.x was obsoleted; it is no longer present in modern 2.x versions of GSL. With the aim of minimising the interventions required in the system, GSL-1.10 was downloaded and compiled from source, re-creating part of the original software environment.
3. Using the modern GFORTRAN compiler options `-fcheck=bounds -Wall` to highlight likely sources of bugs due to insufficiently standard coding led to many warnings. Checking of these warnings motivated many fixes that could be expected to either solve errors in running the main CIRCLES package, or reduce the chance of calculational errors.
4. The autotools `autoreconf` command was run in the main CIRCLES-0.3.2.1 directory and its subdirectories.
5. The COSMDIST package provided by default in a subdirectory of CIRCLES-0.3.2.1 was replaced by a download/configure/compile/install section of the main reproduction script, since COSMDIST is now available in an online git repository. The aim was to reduce the chance of COSMDIST being a blocking factor in reproduction of the calculations.
6. Memory allocation errors that occur for running CIRCLES without previously definining environment variables for key values such as input filenames are present in CIRCLES-0.3.2.1. These were most likely not noticed in RBG08 because of the use of environment variables providing these values. Fixes to the front-end C file CIRCLES.C were made with the intention of avoiding memory allocation errors, which are typically reported to the user as segmentation faults. However, memory allocation errors remained, most likely due to Fortran 77–C

interfacing issues. This is described in the §3.4 on attempted running of the code.

3.3 Dependencies

The full list of dependencies listed – not necessarily really used – in the final GFORTRAN compile operation that creates the CIRCLES binary executable is: `-lpgplot -lpng lcosmdist -lisolat -lastromisc -llapack -lcblas -lf77blas -latlas -lgfortran -lquadmath -lcfitsio -lcosmdist -lgsl -lgslcblas -lm -lgcc -lX11 -lm`.

3.4 Running and a software evolution block

Front end user-friendliness — At the time of writing the original paper, the aim was that the use of GNU tools to provide a free-licensed package configurable and compilable with `./configure && make` and a detailed `./circles --help` command would be sufficient to enable easy reproduction by a scientifically competent user. For example, invoking the CIRCLES help option was intended to show both single-hyphen, one-character options and their equivalent double-hyphen, long options, such as `-i, --cmb_file_raw=FILE cmb fits file of input data`. The freshly compiled version of the code did this correctly, providing the user with a list of available options as expected.

Scripts — My private notes of what were intended to record the most significant steps taken in carrying out the project, along with more minor steps, were used in the attempted reproduction of RBG08. However, in trying to reproduce these steps now, it is clear that the original notes were not as complete and unambiguous as they should be. For the purpose of the current exercise in reproducibility, a completely fresh BASH script was prepared, which verifies the sha512 checksums of input data files and software packages that are downloaded from source rather than provided within the security context of the host system. The style of the new script is partly based on more recent attempts at reproducibility in my own recent papers in which GIT commit hashes of the software[17] and a BASH script for running the full software[18] were provided. Some inspiration was taken from the MAKE-based reproducibility framework[19] recently renamed MANEAGE[20], but the structure of the script is much less modular; it is a simple linear script with a few minimal checks.

This situation illustrates the problem of "insider knowledge" being required for the reproducibility of a paper, where "insider" also includes knowledge that may still be coded in the scientist's brain, but not in written form.

Fortran 77–C interfacing — A more fundamental problem in terms of coding and software environment evolution is that this code uses a C front end and a Fortran 77 backend, configured and compiled together using AUTOCONF tools, in a way that, to the best of my knowledge and that of my co-authors, worked correctly in 2008. The key element for interfacing of Fortran 77 and C code that was recommended at the time was the use of AC_F77_WRAPPERS in the CONFIGURE.AC file[21]. Three of the modular packages called by the main code – COSMDIST, ASTROMISC and ISOLAT – also use AC_F77_WRAPPERS.

The main files are `circles_f77.f`, 2023 lines long, named to emphasise the expected obsolescence of the Fortran 77 language standard; 21 Fortran 77 source files with a total of 11,616 lines of code in `lib/`; and 3415 lines of Fortran 77 code in the auxiliary package `astromisc/lib/`. For modernisation of this code, a minimum approach would be to convert from Fortran 77 to Fortran 2008, in which case Fortran–C compatibility conventions that are reasonably well developed and implemented by the GCC/GFORTRAN family could be used. Alternatively, F2C, which continues to be available in distributions such as Debian GNU/Linux, could be used to convert the Fortran code into C. Any remaining bugs in the interfacing would be solvable within

the standard requirements of C, bypassing the issue of interlanguage communication.

For the purposes of this reproducibility test, the required re-coding effort would be more than is presently justified. While this is, as far as I know, the only free-licensed code for identified-circles matching for the purposes of cosmic topology analysis for which peer-reviewed research has been published, the techniques developed in codes that are not publicly available under either free or non-free licences have developed considerably since 2008. Moreover, the research field is extremely high risk: an observational confirmation of the spatial topology of the Universe would be a historically important discovery, but whether or not this measurement is feasible remains highly speculative. To serve as a basis for long-term projects in this particular field, the software would best be rewritten according to modern standards of C and/or Fortran; and the best tested, accurate, fast, well-coded free-licensed auxiliary libraries, such as CGAL for geometrical purposes, could be used to avoid "reinventing the wheel".

Given that a hurried attempt at a major refactoring of the code would not only tend to extend beyond the scope of the aims of the "Ten Years Reproducibility Challenge", it would also be pointless (a systematic upgrade would better be done properly and thoroughly), this reproducibility attempt was terminated, leaving it with an untraced Fortran–C interfacing memory bug.

4 Discussion

Compiler evolution and Fortran/C compatibility are the main elements of the difficulty in reproducing RBG08, together with the presence of some coding that was not sufficiently robust to allow for the interface change. In 2008, G77 was an obvious choice of Fortran compiler in the stable distribution of Debian GNU/Linux. Since the Debian community already had at the time a solid reputation in terms of software security, verification of licensing and fully transparent and participatory decision-making, this seemed like a wise choice for reproducibility. Using a well-tested, widely used compiler and code standard seemed like a better long-term sustainable choice than using a compiler whose role was still being debated. The main developer of G77 had already announced his intention to stop maintaining the project in 2001[22], but even by 2010, two years after RBG08 was published, the community remained unclear regarding the relationship between G95, based on GCC, versus GFORTRAN, part of GCC[23].

However, also by 2010, GCC already claimed to implement compatibility between Fortran 2003 (ISO/IEC 1539-1:2004(E)) and ISO C99 (ISO/IEC 9899:1999)[24], implemented at the coding level by `use ISO_C_BINDING` and `bind(C, ...)` declarations and the ability to declare C types in a Fortran module.

The IT group at the University of Oxford Department of Physics recommend the modern interfacing methods, and describe the G77 interfacing with C quite colourfully, stating that "Part of the reason for the transition from g77 to gfortran is to make mixing-in with C code simpler, and avoid (most of) the acts of cruel and unusual programming which were previously required to get the compilers' outputs to co-operate. Inevitably, the results of said acts were almost inevitably fragile and non-portable."[25]

An interesting question is whether a reproducibility framework such as MANEAGE[20], which aims at a very high standard of reproducibility with maximum modularity and minimal dependencies, would have enabled easy reproducibility of the original project. The MANEAGE system would, in principle, have configured, compiled, and installed all the original software environment, including GSL and G77 and a contemporaneous version of GCC, and would have encouraged the original project to be modular enough with sufficiently many verification tests to survive a decade of evolution of the software environment. A key question would be whether a more modern version of GCC could have compiled the old versions of G77 and GCC. It is quite realistic to expect that a MANEAGE reproduction of RBG08 would succeed more than the BASH script method provided in this project.

Would this project have been more easily reproducible had it been written in a

higher level language, such as python? The question did not really arise at the time, since as is stated in the abstract of RBG08, one of the key results was a speed-up in computation time, a critical bottleneck for this type of research, in which case the computational overhead of higher level languages renders them impractical.

The aim of the *Ten Years' Reproducibility Challenge* is primarily to get "old code to run on modern hardware/software (with minimal modifications)". Nevertheless, an interesting question would be how much shifting back to an old software environment would be required to run the existing code. This raised a simple practical question: where can we find an official, archived version of the source code of G77 in the version likely to have been used when the paper was calculated and published? This turned out to be non-trivial, because the nature of the relation between G77 and GCC was not sufficiently known by me, nor was it easy to find using search engines. The Debian GNU/Linux developers' guide to Fortran updating from G77 to GFORTRAN, lasted edited in 2016, states that "g77 and g77-3.4 have been removed from the archive"[26]. It was only when the main developer of G77, James Craig Burley, kindly responded to a question posed using a social networking feature provided on a git repository server, that the packaging of G77 *inside* GCC tarballs became known to me. Thus, while finding G77 source code was not as easy as expected, once the detailed information about its relation with GCC was known, finding an archived version on a reputable webserver was found[2]. However, attempts at compiling G77-3.4.6 within GCC-3.4.6 on Debian GNU/Linux stable (9.12) were unsuccessful.

5 Conclusion

It is ironical that in the field of cosmic topology, not only are most software packages only available privately with unknown licences (as tends to be the case in astronomy as recently as 2015[3]), but the code that is explicitly free-licensed, publicly distributed and having peer-reviewed published results has turned out to be less easy to reproduce than expected. This is, unfortunately, consistent with typical reports on science research paper reproducibility[4,5,6]. The specific bottleneck suspected of leading to memory errors in this case was that the effort required to update the Fortran 77 files at the heart of the code, interfaced with a C front end, and compiled with the current GFORTRAN compiler from within GCC rather than with the older, discontinued[22] G77 compiler, risked being too great to be justified on any short time scale. While Fortran has remained actively used by scientists since more than half a century ago and in its modern standards continues to be used actively, and the original CIRCLES package was prepared using the powerful GNU AUTOTOOLS, a robust interface and standards for compiling C and Fortran code together have only evolved quite recently[24].

While the results of the paper are not as trivially reproducible as they appeared to be, the requirement of the Ten Years Challenge for the code to be placed in an online git repository, which in this case is https://codeberg.org/boud/0807.4260, resulted in confirmation that the source code is fully free-licensed, including all libraries and other auxiliary software packages, and the input data files remain publicly available online. Refactoring the code into a format such as MANEAGE[19,20] would be a potential way forward of shifting the research field towards a more completely *open-science* phase. Anyone interested in modernising the software and completing the reproduction of the original results is welcome to contact the author of this paper.

Acknowledgments

Thank you to Konrad Hinsen (the reviewer) and James Craig Burley for several helpful comments. Part of this research has been supported by the "A next-generation worldwide quantum sensor network with optical atomic clocks" project of the TEAM IV programme of the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund. Part of this research has been supported by the Polish MNiSW grant

[2] https://ftp.gnu.org/gnu/gcc/gcc-3.4.6/

DIR/WK/2018/12. Part of this research has been supported by the Poznań Supercomputing and Networking Center (PSNC) computational grant 314.

References

1. B. F. Roukema, Z. Buliński, and N. E. Gaudin. "Poincaré dodecahedral space parameter estimates." In: **Astronomy & Astrophysics** 492 (July 2008), p. 657. arXiv: 0807.4260.
2. J. Luminet, J. R. Weeks, A. Riazuelo, R. Lehoucq, and J. Uzan. "Dodecahedral space topology as an explanation for weak wide-angle temperature correlations in the cosmic microwave background." In: **Nature** 425 (Oct. 2003), p. 593. arXiv: astro-ph/0310253.
3. A. Allen, P. J. Teuben, and P. W. Ryan. "Schroedinger's Code: A Preliminary Study on Research Source Code Availability and Link Persistence in Astrophysics." In: **The Astrophysical Journal Supplement Series** 236.1, 10 (May 2018), p. 10. arXiv: 1801.02094 [astro-ph.IM].
4. J. P. A. Ioannidis et al. "Repeatability of published microarray gene expression analyses." In: **Nature Genetics** 41 (2009), p. 149.
5. A. C. Chang and P. Li. "Is Economics Research Replicable? Sixty Published Papers from Thirteen Journals Say "Usually Not"." In: **Finance and Economics Discussion Series 2015-083** (2015), p. 1.
6. V. Stodden, J. Seiler, and Z. Ma. "An empirical analysis of journal policy effectiveness for computational reproducibility." In: **Proceedings of the National Academy of Sciences** 115.11 (2018), p. 2584.
7. B. F. Roukema and T. A. Kazimierczak. "The size of the Universe according to the Poincare dodecahedral space hypothesis." In: **Astronomy & Astrophysics** 533 (Sept. 2011), A11. arXiv: 1106.0727 [astro-ph.CO].
8. ReScience C Editors. **Ten Years Reproducibility Challenge**. Archived at Wayback. 2019.
9. G. Hinshaw et al. "Five-Year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: Data Processing, Sky Maps, and Basic Results." In: **Astrophys.J.Supp.** 180 (Feb. 2009), pp. 225–245. arXiv: 0803.0732.
10. International Science Council. **Freedoms & Responsibilities of Scientists**. Archived at Wayback. 2020.
11. Github. **GitHub and Trade Controls**. Archived at Wayback. 2020.
12. R. Liao and M. Singh. **GitHub confirms it has blocked developers in Iran, Syria and Crimea**. Archived at Archive.today. 2019.
13. L. Tung. **GitHub starts blocking developers in countries facing US trade sanctions**. Archived at Wayback. 2019.
14. S. Jurdik and S. Spaniol. **Iranian customers can't open Confluence pages**. Archived at Archive.today. 2019.
15. D. Smith. **Update on our planned move from Azure to Google Cloud Platform**. Archived at Wayback. 2018.
16. IASGE Team. **U.S. Sanctions Impact on the Git Community**. Archived at Wayback. July 2019.
17. B. F. Roukema. "Replacing dark energy by silent virialisation." In: **Astronomy & Astrophysics** 610, A51 (Feb. 2018), A51. arXiv: 1706.06179.
18. B. F. Roukema and J. J. Ostrowski. "Does spatial flatness forbid the turnaround epoch of collapsing structures?" In: **JCAP** 2019.12, 049 (Dec. 2019), p. 049. arXiv: 1902.09064 [astro-ph.CO].
19. M. Akhlaghi and T. Ichikawa. "Noise-based Detection and Segmentation of Nebulous Objects." In: **The Astrophysical Journal Supplement Series** 220, 1 (Sept. 2015), p. 1. arXiv: 1505.01664 [astro-ph.IM].
20. M. Akhlaghi, R. Infante–Sainz, B. F. Roukema, D. Valls–Gabaud, and R. Baena–Gallé. "Towards long-term archivable reproducibility." In: **ArXiv e-prints** (2020). submitted. arXiv: 2006.03018.
21. GNU. **Fortran 77 Compiler Characteristics**. Archived at Wayback. 2009.
22. J. C. Burley. **Why I'm Stopping My G77 Work**. Archived at Wayback. 2001.
23. S. Bosscher, G. Pfeifer, and T. Burnus. **The other GCC-based Fortran compiler**. Archived at Wayback. 2010.
24. GNU. **7.1 Interoperability with C**. Archived at Wayback. 2010.
25. Anonymous and Oxford University Department of Physics. **Where has g77 gone to? ...or: libg2c considered harmful**. Archived at Wayback. 2020.

26. Riku Nchip et al. **GfortranTransition — Transitioning from g77 to gfortran**. Archived at Wayback. 2016.