Review: "automan: a Python-based, automation framework for numerical computing"

Olivier Mesnard[1]

[1]The George Washington University

April 28, 2020

Review the manuscript CiSESI-2018-02-0013 for *Computing in Science & Engineering*: *"automan: a Python-based, automation framework for numerical computing"* (Ramachandran, 2018).

The manuscript describes an open-source Python-based library, called `automan`, to automate scientific computational workflow. automan is open-source with a permissive license (BSD 3-clause), hosted on GitHub (developing the code in the open under a version-control system). The manuscript was made available on arXiv (https://arxiv.org/abs/1712.04786) before submission to CiSE and serves as API documentation for the Python package.

The main purpose of automan is to automate a computational research manuscript: it provides a Python framework to assemble and run simulations (on local or distributed machines) and to generate figures from the data produced by those simulations. With automan, a user has the possibility to re-create all figures used in a research paper with one command line (as envisioned by (Schwab et al., 2000)). The content of the manuscript perfectly fits into the types of submission sought by the RR track of CiSE ((Barba and Thiruvathukal, 2017)).

The author gives an extensive review of already existing frameworks to automate workflows, leading to a clear explanation on the need for a new tool such as automan.

As mentioned in the manuscript, automan has already been used by the author to automate the generation of the figures in an other manuscript (under review) that compares different Smoothed Particle Hydrodynamics schemes (to simulate incompressible fluid flow problems). The examples provided in the manuscript are well detailed such that it is easy for the users to adopt automan for their own workflow.

As part of the review for the manuscript, I successfully installed automan and created a toy problem to test the features of the package. I used an in-house two-dimensional panel-method code (written in Python) to compute and compare the surface pressure coefficient on a series of NACA airfoils are different angles of attack. automan is modular enough such that I did not have to re-run the simulations if I was only creating/tweaking the post-processing code to create figures. When needed, I was also able to re-run only specific simulations (and not the entire batch). I created a public GitHub repository (https://github.com/mesnardo/automan-example) that contains the input files and Python scripts used to test automan.

Finally, I would like to point out that automan helps users to make their computational workflow reproducible, but does not guarantee numerical reproducibility of the results. As mentioned by the author, automan does not capture the working environment (software versions, dependencies, etc.)

The manuscript should be **accepted after minor revisions**. Following is a list of suggestions/modifications the author should address:

1. I think the author should favor the term "reproducibility" instead of "repeatability". The meanings of "reproducibility" and "replication" are clearly defined in the highly cited paper from Peng (2011).

To add more clarity, the author could recall the definition the "reproducibility" and "reproducible workflow" in the introduction section.

2. As mentioned by the author in the discussion section, automan is a "very young package". Thus, it is very likely that the core of the package and/or the API will change in the future. Therefore, the author should mention the release version of automan on which the manuscript is based.
3. (p.1, l.47) Could the author add in the introduction the license under which the code is being developed and released (BSD 3-clause license)?
4. (p.3, l.31) The term "exactly similar" is too strong to describe a re-implementation of a feature that already exists in luigi. Maybe the author could remove the word "exactly".
5. (p.13, l.12) Small typo ("elgant" -> "elegant").
6. (p.7, l.53; l.26 in Listing 1) I think the name of the method is `make_output_dir`, instead of `make_output_directory`.
7. (p.9, l.5) It was not clear to me which folders or directories I should delete to re-run a completed simulation. It would be interesting to add a command-line instruction as an example.
8. Although this is probably outside the scope for this submission, the developer could use a tool, such as Doxygen or Sphinx, to write the API documentation.

References

Lorena A. Barba and George K. Thiruvathukal. Reproducible Research for Computing in Science & Engineering. *Computing in Science & Engineering*, 19(6):85–87, nov 2017. doi: 10.1109/mcse.2017.3971172. URL https://doi.org/10.1109%2Fmcse.2017.3971172.

R. D. Peng. Reproducible Research in Computational Science. *Science*, 334(6060):1226–1227, dec 2011. doi: 10.1126/science.1213847. URL https://doi.org/10.1126%2Fscience.1213847.

M. Schwab, N. Karrenbach, and J. Claerbout. Making scientific computations reproducible. *Computing in Science & Engineering*, 2(6):61–67, 2000. doi: 10.1109/5992.881708. URL https://doi.org/10.1109%2F5992.881708.