Geosci. Model Dev. Discuss.,
doi:10.5194/gmd-2016-318-RC1, 2017

Geoscientific
Model Development
Discussions
Open Access

Interactive comment on "eddy4R: A community-extensible processing, analysis and modeling framework for eddy-covariance data based on R, Git, Docker and HDF5" *by* Stefan Metzger et al.

Anonymous Referee #1

Received and published: 3 March 2017

1 General

- The paper describes the development of an open source framework for the processing of eddy-covariance data. The framework is developed to deal with the wealth of data that will collected within the National Ecological Observatory Network (which is located in the US, a geographical specification not given in the manuscript).
- The paper both addresses the development process as well as some higher level

details of the framework being developed. Little specific information is provided about the actual processing algorithms.

- Three case studies are presented in which the processing framework, in its present state, has been used.
- My general impression is that the work presented in itself is very worthwhile, and -at least in our field- quite novel.

However, I do have some comments:

1. The Development and Operations (DevOps) framework is mentioned as an essential characteristic of the work discussed. However, the DevOps framework is not clearly introduced to the reader (except for one paragraph in the introduction). Even there, most emphasis is placed on the tools, rather than the essential steps or processes that are part of DevOps. I would have expected some sort of a list of steps, or a schematic that shows general characteristics of the workflow in a generic DevOps development process (Wikipedia already shows some colorful examples). Then these generic characteristics could be translated into the specific characteristics of the project that is the subject of the paper. Furthermore, although a number of problems in current practice of EC-data processing are identified (lines 48-62), it remains unclear why DevOps would be the answer.
2. The description of the DevOps framework in section 2 is in fact a description of the collection of tools. It is hard to recognize the DevOps characteristics of it.
3. The description of the tools in section 2 remains rather vague on one hand (2.1, 2.2), and becomes very (too) specific at other points (2.3). For example, to me the message of section 2.1 is that the authors have implemented regular EC-processing software in R. It remains unclear what are the special properties of

this implementation that make it so much better/easier/more flexible/.... than any existing EC-processing toolchain (except that in a number of places terms are used that at least suggest that something special happens in the coding, while it remains unclear what this means in terms of code quality, re-usability etc. e.g. lines 131-134). Similar comments would hold for the other parts of section 2 (see below).

4. The various sections in in section 2 (except 2.1) in my view insufficiently address what the role of the different tools is (on a conceptual level, not an implementation level): who/why are these tools part of the proposed system. Furthermore, some of the sections are not very specific for the proposed application of the tool (section 2.3 could be used in nearly any paper that desribes the use of Docker). The same holds to a large extent for sections 2.2 and 2.4.

5. To summarize my main concern: the paper misses a clear problem statement and as a result it is unclear why the presented software development would be the answer. The paper would be worth reconsidering for publication if the authors would be able to reformulate the paper in such a way that:

- - there is a clear problem statement (which may, or may not be related directly to the NEON network);
  - it is clear that the DevOps methodology is needed to tackle that problem (including a clear general introduction to DevOps, irrespective of the tools used);
  - the this set of proposed tools and methods would enable a DevOps methodology for the task at hand (EC-data processing);
  - the presented cases (section 3) clearly illustrate why a software infrastructure as proposed in the paper is needed.

Below I will provide detailed comments

Note: in the comments below, the comment is preceded by the line number.

2 Detailed comments

1. 56: I do not see why the ultimate goal should be a universal EC processing environment. As long as the software that is used in papers is described in open literature and the source code is openly available, readers will be able to assess the results of the EC-processing used in the described research. Research groups will always have reasons to do things their own way: because of instrument or site specifics, or due to specific requirements in output and analysis. And from software intercomparisons performed in the past it is quite clear in which aspects the main differences between different processing packages occur (e.g. Mauder et al., 2008; Fratini and Mauder, 2014).

2. 71: it is unclear why DevOps is being embraced: why is this methodology so suited for the problem at hand (the problem is not clearly stated, but I assume that the 'strong need' expressed in lines 66-70 is 'the problem')?

3. 83: indeed, I agree that the fact that the proposed work enables an exact reconstruction of the system that was used to construct certain derived data is an important asset (having the data and the software is -in some cases- insufficient to enable exact reproducibility). On the other hand, in the application at hand, bitwise identical results are usually not needed: the statistical errors in the derived quantities are usually orders of magnitude larger than the differences that occur do to differences in details of the computational environment.

4. 96-106: this 'cycle' suggests some regularity and pace of development. Is that what is intended, or does it simply mean that once someone has a suggestion

for a new feature or new implementation, the code has to go through the cycle? My own experience with code development is that many users are using very different versions of the code (from very old to cutting edge) which may yield impractical surprises when these different users supply new code. The advantage of the -apparently well-funded- NEON network is that paid personal is available to oversee new code submissions.
In this section it is not clearly defined who/what is the 'science community' on one hand, and 'NEON science' on the other hand.

5. As explained in my main comment: section 2.1 seems to mainly discuss 'just another EC-processing tool'. Please focus on those aspects that are new and make it particularly suited for NEON, and for use in the DevOps methodology. My impression is that the direct implementation of the option for parallel processing of EC-data is one important aspect (which may not be relevant if a group only operates a limited number of towers, but which *may* be relevant if in 10 years time NEON decides to reprocess all EC-data of all stations according to the latest insights). But if this parallelization is so central, I would like to see some more explanation/example/tests of it (performance, scalability). Another aspect would be indeed the modularity if it allows -as advertised- for the easy adaptation of new hardware. I suppose that you use some sort of instrument abstraction which makes it possible to describe the properties of any thinkable instrument in such a way that the data can be ingested and processed in a correct way. Again, more details on this potentially unique aspect would make the paper worthwhile to read.

6. 168: indeed git is open source and free, but Github is not (unless you use a public repository). That brings me to the question in which way eddy4R will be open-sourced. Will the github repository be opened up for everyone (by now I cannot find it)? Or will you publish certain stable versions to the public and keep the development closed?

Reliance on Github may seem a risk when planning for the processing of data from a project that will run for 30 years. But since git repositories are stored locally as well, no risk exists in case Github would go out of business.

7. 181: who will do the review and testing? Do you have full-time staff for that? Will the testing be automated in the sense that when the new code is included, all prior test cases should still run without errors, while the new code should also provide it's own test case (if it implements a new feature)?

8. 198: The phrase 'minimal context' suggests that Dockers are small. However, in my experience Docker files are usually quite bulky (as they contain a complete operating system + the software needed to run the scripts). Although storage is not an issue nowadays, the wording suggests something different than what readers might expect.

9. 211: it is unclear to what extent hub.docker.com provides versioning (in the sense that I would be able to exactly reproduce a docker that was produced 5 years ago (with the same Debian version and the same R version with the same package versions). In lines 324 and further it becomes clear that indeed you can specify a version of the docker.

10. I *do* understand that a docker image provides a machine that can do the data processing in a way that is independent of the underlying hardware and software (OS). But you do not specify how this machine (which I still would consider 'virtual') talks to the local file system (outside the docker) to read the raw EC data and write the results. Or is the data flow always supposed to pass through the NEON databases (which are accessed over a network connection).

11. 249: does 'uncompressed' mean 'ASCII', or does it mean 4 or 8 byte binary real values. In the first case a 1:10 compression ratio is not unexpected, in the second case I would be surprised.

12. 251: I am surprised that only the variable name and units are added as meta data. Since an important reason for the proposed methodology is reproducibility and traceability, I would expect that also metadata on the processing itself (software versions, tool chain, processing configurations) would be included. In that way a file with processed EC-data, when 'found in the wild' would still tell the story of how it was produced.

13. 253: I would say that text-based data storage of raw EC-data is already something of the distant past. NetCDF has gained significant usage since 15 years. When properly used, NetCDF files are self-descriptive as well (the same reservation would hold for HDF files: the meta data have to be filled). Otherwise many people use the binary file formats produced by their data logging infrastructure. So again, please indicate the real advantages of the HDF format as compared to others, vis-a-vis the requirements of your application, in combination with the DevOps framework. I could think of two aspects: compression of data (which is not possible in NetCDF) and the hierarchical data structure.

14. 258: in the figure it is unclear if this depicts one file, or multiple files. Furthermore, the terminology ('group', 'Level 1,2,3', 'DPL', ...) is not explained. It is insufficient to keep the reader in the dark and just refer to the 'NEON data product naming convention'. To summarize: I do not understand what I am looking at.

15. 268 and related text: in principle, the configuration shown here is not much different from the scripted use of a compiled program (or interpreted script): what is done by the docker images is described in the parameter files. What may be interesting is that the first (left-most) docker-image spawns the second (right-most) images, which are identical in implementation, but serve a different purpose because they are instructed to do so (either the 'turbulence' task or the 'storage' task). Another important asset of the workflow is that apparently meta-data on the processing is included in the HDF files (although line 251 suggests other-

wise). It would be of interest to give more details about that (how self-contained are the HDF-files?).

16. 283: I wonder why apparently a single day is used as the unit of storage when it comes to storage of L1-L4 data. In many applications I would think that longer time series are needed. Or does the data-portal glue these daily datafiles together when the data-request to the portal asks for e.g. a full year of data?

17. 284: As it is shown here, the workflow is tightly integrated with the data portal. It seems to be the only use case. But what if a given researcher wants to (re-)process her/his data on her/his desktop for private use only? Is that possible as well? And how would that alter the use of the presented infrastructure?

18. 294-297: this is the first point where I clearly see a reference to DevOps that links the presented software infrastructure to the advantages that DevOps could provide.

19. 300: again, this figure contains a number of unexplained acronyms and terms. Furthermore, I wonder what the added value of this figure is for the paper as a whole. Finally, a part of the text is very hard to read.

20. 303: section 2.6 reads as a user manual, rather than the description of the logic of operation. To me it is unclear why you would need to login to the docker machine (unless you want to develop and test new R-code. On the other hand, figure 5 suggests that the docker-images can be instructed 'from the outside' using parameter files.

21. 341 and further: to me it is unclear what the function of the three case studies is. In the introductory paragraph details on the computing infrastructure is given. This suggests that some benchmarking in terms of performance will be done. However, this only makes sense to me if there is a standard against which the

new setup can be compared (e.g. current practice of reading ASCII files on a single processor machine?).

22. 349 and further: if indeed the performance of the software (in terms of speed, and perhaps also ease of use, flexibility . . .) is the objective of section 3, I do not see why so much attention needs to be paid to the experimental setup (including photographs!) as well as the interpretation of the results (section 3.1.2). Any EC dataset with a length of 1 to 2 weeks would have sufficed.

23. 381: what kind of additional complexity? It could be that the presented processing infrastructure makes it easy to logically define different processing levels (L1 to L4). In that case it would be of interest to the reader to clarify which are the differences between the various processing levels, and how,conceptually, they can be defined and configured with the current setup. Perhaps this is an important added flexibility? In line 400 it becomes at least clear what is the distinction between L1 and L4.

24. 386: what is the difference between a simple data format and a compound data format. Since the use of the HDF5 file format is such an important aspect, I would expect a clearer description of the way in which the added possibilities of HDF5 files are used (which meta-data, mixes of L1, L2 . . . data . . . .).

25. 389-390: does this mean that the calibration step (to go from L0 to L0p) takes 4.8 PU minutes? Compared to the actual processing this seems long. But perhaps there is a good explanation for it. Please provide the reader with one (or perhaps my interpretation of the numbers is incorrect).

26. 399 and further: only include these results if they show something that only this software can do, and other EC-processing methodologies cannot.

27. 423: the fact that the presented methodology apparently is equally well able to deal with aircraft data as it is can deal with tower data is an interesting added

value, worth advertising! However, I would have appreciated it if the authors would have clearly shown which parts of the processing would work the same for tower and aircraft data, and which steps would be different (e.g. by referring to figure 5). Again, the details of the particular data set (conditions of collection) is less important than the type of data and instruments involved.

28. 442: it is unclear whether the 100Hz → 20Hz reduction is done by the same software or was part of the preparation of the L0 data (i.e. from L(-1) to L0.

29. 445: it is unclear whether the software is able to perform all the corrections that are related to the accelerations of the aircraft (the wind speeds derived from the pressure readings are not the wind speeds). I could imagine that this is part of the L0 → L0p conversion, but this is not specified.

30. 449: it was earlier suggested that the workflow is based on HDF5 files, whereas here gzipped ASCII files are used. I understand that R is able to read all kinds of files, but is reading the ASCII files a standard feature of the methodology or did you first convert the gzipped ASCII files to HDF5 (to produce real L0 data) before ingesting them into the processing software?

31. 451-460: This is an interesting, perhaps non-standard, chain of processing steps. For the reader this would be an interesting example to see how easily this can be configured in your software. Is it just a matter of setting a number of flags in your configuration files? How easily can you change the order of various steps that are applied to high rate data or reduced data?

32. Section 3.2.2: again (like for 3.1.2): presentation of the results is only relevant if your methodology can do something that existing software cannot (or if yours can do it better/faster/easier/more insightful). For instance, if figure 11 would be output that can be produced automatically it could be of added value.
In that respect it would also be interesting to know if the software could be used

in a scenario where certain processing has already been done, and the user decides that he/she wants additional output. Would the software be able to figure out which data (L1... L4) is already there, and which is the minimum set of additional processing steps needed to produce the additional output. Again, in that sense it would be worthwhile for the reader (and potential user) to know which output your software can produce, based on which type of input data (type as in L0, L0p, etc). A well-organized table would be more informative than three case studies. Similarly a clear definition of which types of input data can be ingested would be helpful.

33. 499: this is an interesting application! Does the script automatically select the EddyPro settings that would match the eddy4R settings? If so, providing the difference statistics with your L1-L4 output in the HDF5 file would be useful added value, since users of your L1-L4 data would have an indication how sensitive the resulting aggregated data are to details of the processing (he/she does not know who is wrong and who is right, but a sense of the sensitivity can be helpful).
34. 519: Do you run EddyPro in the same docker as the R-framework? It would be interesting to know what the performance difference (if any) is between EddyPro and your R-based software.
35. 550-560: it is OK for me that in the paper you present the workflow as it is envisaged. But it would be good to already clarify in the main text which steps have already been implemented and which are underway (for the eddy4R part you clearly made this distinction, but for the other parts not). Furthermore, it remains unclear to me to what extent the workflow you propose is strongly tied to the NEON infrastructure. Or could I, as a scientist not working within NEON, be able to adopt your methodology as well. To pose it differently: where does your methodology end and where does the NEON infrastructure start?

3 Very detailed comments

1. 63: please add that NEON is being developed in the US.
2. 68: what do you mean by ‘tight hardware-software integration’? And which code is ‘distributed’ and where ?
3. 189: I think figure 3 is superfluous. The message conveyed by the caption could be included in the text. No illustration needed.
4. 277: for people working in the field of micrometeorology (and particularly when involved in CO2 exchange) the terms “turbulence” and “storage” will ring a bell. But for people somewhat more remote, but interested in the proposed collection of tools, it will be less clear (in the context of a strongly IT-oriented paper “storage” might mean something completely different to the reader). For the clarity of the paper the distinction between the turbulence and storage workflows is in fact irrelevant. The message is simply that based on the same code/machinery you can do different things with the same data Unless you want to show that you can have various parallel workflows which could also interact (but then you have to clarify the potential interactions).
5. 317: I think that figure 7, apparently just a collection of screenshots, is superfluous.
6. 376: it is unclear to me how the current setup would lead to 50 high-rate data streams.
7. 385: the 0.1 to 0.2 Gb: I suppose that this refers to the *input* files (L0 or L0p).
8. 459: the performance is only relevant if it can be compared to something else
9. 460: what do you mean with ‘file-backed objects’?

10. 565: what do you mean by defensible?

References

Mauder, M., Foken, T., Clement, R., Elbers, J. A., Eugster, W., Gr, T., ... & Kolle, O. (2008). Quality control of CarboEurope flux data–Part 2: Inter-comparison of eddy-covariance software. *Biogeosciences*, *5*, 451-462.

Interactive comment on Geosci. Model Dev. Discuss., doi:10.5194/gmd-2016-318, 2017.