Hydrol. Earth Syst. Sci. Discuss., 9, C6405–C6409, 2013
www.hydrol-earth-syst-sci-discuss.net/9/C6405/2013/

Hydrology and Earth System Sciences Discussions

Interactive comment on "Technical Note: An open source library for processing weather radar data (*wradlib*)" *by* M. Heistermann et al.

M. Heistermann et al.

heisterm@uni-potsdam.de

Received and published: 18 January 2013

Dear Referee,

Thanks a lot for your response to our *"Technical Note: An open source library for processing weather radar data (wradlib)"* and for the many helpful comments. On the following pages, we would like to reply to the major comments (all comments related to typos and other minor issues have been dealt with in the revised manuscript).

Before, though, we would like to give one general response on some of the referee comments: Both referees asked, for a couple of aspects, for more detailed information, e.g. in terms of example code, technical guidance, or with respect to data formats.

Although the paper is submitted as a Technical Note, we are hesitant to overload it with technical detail, as it still should be brief and concise, and, most important, not be mistaken as a tutorial or quick start manual. The example code shown in the manuscript is rather intended to give a “feel” of the library usage, not to provide complete workflows. On the other hand, we agree that the information requested by the reviewers could be helpful to the reader and to potential users of the wradlib library. Thus, for some of the issues raised by the referees, we included the requested information on the wradlib documentation page (http://wradlib.bitbucket.org), mostly as topical tutorials or recipes. Please consider this only as a general remark. The specific suggestions will be found within the replies to the comments below. Please note that the referee comments are shown in **bold format.**

We believe that the revised manuscript, together with the enhanced wradlib documentation pages, is improved as compared to the first version and we are very grateful for the advice given. We hope that the present quality of the manuscript, together with the following replies, meets the standards of HESS and can now be considered for publication.

Sincerely Yours,

Maik Heistermann (on behalf of the co-authors)

In order to understand which specific routines have been implemented in wradlib and what specific implementation was used, would it be an option to provide a bit more details in Table 1. More specifically, it would be nice to add the publications of the original processing algorithms, so far implemented in wradlib. For the reader used to work with volumetric weather radar data at a regular basis, this will give insight in the capabilities but also limitation of wradlib, while for new users further reference is provided.

Our original idea was to provide that kind of detail – e.g. references to implemented algorithms – in the sub-sections of section 3. However, we agree that an additional column in table 1 could be helpful to provide a quick overview about the current state of development. The manuscript was revised accordingly.

In Chapter 3 the authors provide a number of features to process radar data, including the numerical code used to call the specific library routine. In the current manuscript, for each of these features, a different example/figure is provided. However, except for Figure 5 and 6 for a synthetic case, these are stand alone examples. It would be nice if the authors could provide one specific case study, using actual weather radar and rain gauge observations to show the processing capabilities of wradlib, in a step by step manner. Especially for readers, not used to the processing of radar data at a regular basis, such an example could show the benefit/usability of wradlib.

We agree that a full worked example would be interesting to show. However, we suspect that this is beyond the scope of this technical note which should be rather concise and brief. Instead, we suggest publishing more elaborated examples under the tutorials section of the wradlib homepage and refer the interested reader to these sources. This way, we are also able to keep the content of the actual manuscript more generic, while the wradlib documentation page can be easily updated with new examples or updated/improved algorithms. We already published a tutorial illustrating a typical processing workflow under http://wradlib.bitbucket.org/tutorial_workflow.html. A recipe for the workflow underlying Fig. 4 was published in the recipes section of the wradlib documentation page (http://wradlib.bitbucket.org/recipes.html).

Since, the processing of radar data comprises the analysis of large three dimensional arrays, which are updated every 5-10 minutes, processing these data can take considerable amount of time. To gain a better understanding on its hydrological potential, it would be good to provide some specific details on the

computational speed of wradlib, especially with respect to analyzing volumetric data for one or multiple radars.

On the one hand, we agree that providing information about the computational requirements of wradlib would be interesting. On the other hand, the informative value of selected performance tests might be limited due to a number of reasons: (i) the computation time for specific tasks strongly depends on the spatial characteristics of the underlying radar data (e.g. the angular and range resolution or the number of available elevation angles). Of course, this information must be provided with the test results; however, the results might be difficult to transfer due to non-linearity; (ii) the performance depends on both CPU and memory and potentially other hardware settings. Again, transferability might be an issue; (iii) wradlib is still young and not tuned towards performance, yet (as is pointed out in the manuscript, too). However, if users will find selected functionalities to limit computational performance, it might be possible to address these issues with reasonable effort; (iv) the scope and organization of a processing workflow highly depends on the user's preferences. It is thus trivial to say that the actual application of the library determines the required computation time; (v) computational benchmark experiments are particularly useful if a benchmark was available. Such a benchmark could be the processing time required by another radar software or library, together with standardized processing tasks. However, such a set-up is well beyond the scope of the paper.

Still, we agree that it would be good to get an idea about the order of magnitude required for typical applications. We selected the example of Figure 4 to provide such an estimate and provide the information in the revised manuscript within the caption of Figure 4: Processing two hours of 5-minute resolution scans for two radars with 360 azimuth angles and 128 range bins including reading raw data, clutter correction, iterative attenuation correction, gridding and quality-based composition takes about 20 seconds (on a 32bit Windows machine, using one out of eight threads of an Intel 2.93 GHz Quad-Core processor).

The authors mention that wradlib is an open source library, which enables users to add numerical routines themselves. However, in order to avoid that within a few years many versions of wradlib are used all with different capabilities, depending on the users capabilities to add additional routines, how are the authors going to add the user provided additions, to the latest version of wradlib.

In the revised manuscript, we give more information on how to integrate new developments into wradlib in section 2.4. First, the wradlib lead developers offer the service to integrate new developments or algorithms in the wradlib main branch. Second, Bitbucket/Hg offers a powerful and convenient mechanism to integrate your own developments into the main branch: the so-called Pull Request. This is the standard procedure to integrate new developments in wradlib: Basically, you “fork” the main wradlib repository, implement your own changes, and then apply for merging your fork back into the main repository (provided that the change is approved by the lead development team). We added a section to http://wradlib.bitbucket.org/community which guides potential developers step-by-step in applying this mechanism.