gmd-2015-267 comment to referee #2

We would like to thank the reviewer 2 for the careful reading and the constructive review.

General Overview

With this paper we wanted to present YAC with a special focus on a set of selected key aspects that we did differently compared to other coupling solution. The weight generation algorithms we use are more or less the same as in any other coupler and are hence, from our point of view, sufficiently documented. We will try to clarify the focus of the paper.

Overall, I believe there is a shortfall in the number of cited references.

Additional references will be added accordingly.

Are the weights generated on the source or target side, on the union of all tasks or is it up to the user?

The search and the calculation of weights is performed on all source processes (owners of the source grid). Here the data are interpolated onto the target before sending. We will update the revised version of the manuscript accordingly.

Is remapping done on the source or target side or as part of communicating data between models? Is the communication of data between models separate from remapping or part of remapping?

We did not consider our implementation of the remapping step particularly interesting and therefore a detailed description has been omitted.

First, the source processes communicate with each other for a kind of halo exchange in order to provide the necessary data for the processes that have to calculate the stencil. We follow the referees request and will modify the revised version of the manuscript accordingly by adding an additional chapter.

The authors are extremely familiar with Oasis4, but it's unlikely the readers will be. It is fine to compare to Oasis4 and point readers to an Oasis4 reference, but it is also important to make sure the description does not assume the readers are inherently familiar with Oasis4 and that it does not require a study of Oasis4 as a prerequisite.

We will revisit the paper with this in mind.

How does YAC ensure that the weights generated on the fly are of high quality? The text notes problems at poles in various algorithms, issues with different types of grid cells and different edge options, the requirement that one grid have either convex edges or be rectangular. Are those things

> *checked on the fly? Are the weights somehow checked for conservative or gradient properties independently after they are generated? Are the properties at the pole checked? Is this a concern?*

Detailed description of the interpolation algorithms themselves was not supposed to be part of the paper, because from our point of view they do not contain noteworthy differences to other implementations, except for the clipping that has its own paragraph. We will add a sentence to clarify our intention.

> *Page 1, Line 20: "not critical" is not a proper description. ESMs have been carrying out computations on moderate to high resolution grids for several years and the (pre) computation of interpolation weights in terms of both performance and quality has been a critical issue. This issue has taken up not insignificant resources in several projects including OASIS, ESMF, and SCRIP.*

With this paragraph we refer to past generations of ESMs as stated in the beginning of this sentence. “Past generations” we used here as a synonym for coarse resolution models as it has been mainly used in the past CMIP phases. For those coarse resolution models with a low number of horizontal grid points of the order of less than 100.000, the compute time of the neighbourhood search has not been an issue, not in our models nor – to our knowledge – in other coupled models of these generations. We will clarify what we mean by “less critical” in the revised version of the manuscript by shifting the focus onto the horizontal resolution.

> *Section 3.2 describes the communication implementation and is far more important than much of the other material before it. I would like the description here to be expanded a bit, especially as related to lines 10-22. You need to describe "the non-blocking buffered send from Oasis4" for those that are not familiar. Please explain how the callback and data pointer work in a bit more detail. This is not entirely clear. Remember that your audience didn’t work on either the Oasis4 or YAC implementation. Also, maybe there needs to be some further clarification on what aspect is being described. There is communication associated with weights generation, interpolation, and coupling data. Is this communication approach used for all of these?*

We will remove the OASIS4 reference from this section. We will improve the description of the communication scheme and make clear that the communication scheme is used throughout the YAC internal workflow.

> *Section 3.4. The interpolation stack feature is well thought out and something that other weights generation methods are not able to do easily yet but is needed. Well done.*

Thanks.
For us this part is more important than the interpolation methods themselves, because it might give others new ideas for future developments.

gmd-2015-267 comment to referee #2

> *Section 3.7. Does the current calculation of intersections guarantee conservation. The overlapping areas have to be computed in a way that the partial areas add up to the areas of each grid cell in total. I think page 14, line 3 suggests the areas are handled properly, but maybe a sentence stating this would clarify.*

The sum of the partial areas will always add up to the area of the respective grid cell. Everything else is a bug or numerically inaccuracy. We might follow your suggestion and enhance the paragraph in this regard.

> *Section 4. I find the description of the user interface a bit out of place. The article is really focused on the interpolation weights generation. Nothing has been presented about how interpolation is carried out nor how models are coupled. This section, describing how to setup YAC via XML does not seem to fit into the paper.*

Without a user interface the coupler library would be hard to use. Therefore we still think it is worthwhile to mention it, but we will move it to an Appendix.

> *Section 5. The total time and scaling for weights generation is very good. I think it's also important to show similar timing information for patch (1st, 2nd, and 3rd) and nearest neighbor calculations as well as bilinear if it's available. These are very different algorithms and I think providing scaling information for all three is important for this paper. In our experience, the 1st order conservative weights generation is sometimes the easiest and fastest to carry out. (This is also noted in Section 6, so data to back it up would be very useful)*

Additional measurements will be added. The reading and writing of weight files works but is not yet optimised. Therefore we would like to refrain from adding detailed measurements. Furthermore, performance depends on various other external parameters like current workload of the system, the file system, the configuration of the IO library. We consider it far beyond the scope of this paper to analyse this in detail and provide sound interpretations of measured results.

> *Figure 4 title seems incorrect. According to the text, it is the time to generate 1st order conservative mapping weights, not to carry out remapping.*

The title will be adjusted.

> *Section 6. Line 25. The only thing that has been shown in the text is the cost of the weights generation for 1st order conservative. It is a stretch to say "YAC scales reasonably well". Much of the YAC performance is not shown including remapping performance and cost to communicate data back and forth between models, the ability to support concurrency and other issues.*

gmd-2015-267 comment to referee #2

Except for the declaration of the local data on each process which contains no communication, the measurements show the whole initialisation cost.

We do not show measurements for the actual remapping and data exchange, because they typically have no impact on the performance of the model. Nevertheless, we will add respective measurements.

Figure 4 shows measurements with up to 12.288 MPI processes. In our opinion this shows the ability of YAC to support concurrency.

> *Section 6. Line 30. It would be nice to document the actual time needed to write and read mapping files as this is already available in YAC to have a quantitative comparison against the cost of generating the mapping weights. It would not surprise me if the read operation was more expensive than online weights generation for the test case used, and the actual numbers would add to the paper.*

See our remark regarding additional measurements above.

Comments that are mainly on the coupler and not the paper:

> *Section 3.3. The requirement on the user to decompose the grid in a way that also includes a halo region and the rank owner of each halo gridpoint seems to be rather inflexible. That datatype and information will almost certainly have to be computed specifically to support the YAC weights generation interface and is unlikely to be a natural part of any model decomposition.*

It is available in ICON. Furthermore, each model that uses advection and diffusion operators in a domain-decomposed world must have the knowledge about the halo. This includes ocean and atmosphere models. We agree that purely column based models like land components or chemistry (not chemistry transport) may have a problem here. In our case land and biogeochemistry are part of one component which does know about the composition. We do not see any need to modify the manuscript in this respect.

> *The halos are probably only needed in a subset of interpolation methods (like bilinear) and maybe YAC should be computing that connectivity, not the model. In addition, in a decomposition like round robin where each process has a random(ish) set of points, the halo description is going to require that "n" halo points be specified for each grid cell, increasing the memory and complexity. YAC would be much more usable if the connectivity were computed within the coupling layer when needed.*

The halos are used to identify communication partners in the 1st-order conservative, patch recovery, nearest-neighbour and average interpolation. In addition it is used by

the global search. The design is fundamentally based on having the halos. We could identify the owners of the halo points internally, but since ICON already provides us with that information, we did not yet see a need to do that. OASIS4 made an attempt to compute the connectivity within the coupling library. Our personal experience showed us that these attempts failed in the sense that is was not possible to provide a stable and performant algorithm. With every new grid configuration that was introduced to OASIS4 the algorithm needed to be revised. We do not see any need to modify the manuscript in this respect.

We do not mind not supporting round robin like decompositions.

> *Section 3.5. Do you have a bilinear interpolation option? This option is heavily used in ESMs.*

Average with inverse distance weighting basically is linear interpolation. (page 12 line 8). The patch-recovery with a linear polynom fit or a 4-nearest-neighbour interpolation would be another alternative. For triangular grids a bilinear interpolation is not defined.

> *Section 8. The web info is pretty useful. Just FYI, I think it is missing a "getting started" type of documentation for new users to know what calls are needed and how to organize them in their models.*

The source code contains trivial toy models that show how to use YAC. We will take up this suggestion and refer to them on the Doxygen page as well.

> *Technical corrections*

We will consider all technical corrections and revise the text where requested by the reviewer.

Concerning references for both L'Huilier's Theorem and Girard's Theorem in the paper we consider this textbook material like the Pythagorean Theorem for which one usually does not provide citations. For both theorems we are not able to locate the original sources where these were published first by Simon Antoine Jean L'Huilier (1750 – 1840) and Albert Girard (1595 – 1632).