Review Veris: Fast & Efficient Sea-Ice Modeling in Python woth GPU acceleration

Gärtner et al.

This manuscript evaluates “Veris” a sea ice model written in Python optimized for GPU processors. Veris is written based on the implementation from MITGCM. This is the main innovation and a good step forward.

This is the second review and this version of the manuscript is more focused, however I would still recommend some changes.

Within the model description I still find that there is a discrepancy between the broad model description and the description of the EVP solver that is mostly in focus here. This weight should somehow be reflected in the model description and more focus should be on the EVP. Maybe because the manuscript try to be both a general model introduction and at the same time focus on a specific element.

The manuscript mostly focuses on the dynamic solver and how fast this is solved.

The study generalizes a comparisons of CPU’s and GPU’s based on 1 example of each. In reality this is a bit more complex as one for instance can buy both low end and high-end hardware. Therefore, I think that the conclusions of CPU vs GPU are generalized more than the result allow. It is good that the power consumption is brought into the discussion. It should be mentioned that price will also be a part of the decision when buying hardware. A comment about the price in the discussion is sufficient as these will vary.

For the ocean model Veros I find that the discussion of its ability to couple is good but not particular important for this manuscript, thus it should not be mentioned in the abstract as it is not important for the conclusion of this manuscript. I would move this into the discussion with a reference to future development.

For the result/discussion I would like to have a larger focus on how the time to solution/scalability should be improved and if this is possible on the provided hardware. There is short reference to saturation (bandwidth I guess), which is likely the cause, however I miss an analysis of this. Is it feasible to have an effective EVP solver with the high number of MPI calls? Should OMP parallelization be used?

You have already cited Rasmussen et al., there is a discussion of the bottlenecks in a traditional FORTRAN EVP solver. Is it the same bottlenecks in the Python code? It would be nice if the authors could show what the bottlenecks are.

Specific comments

Line 25: Especially for the case here focused on the EVP solver is it really the case that the number of floating-point calculations is important for the run time? Hardware has changed with more and more cpu’s per node/GPU’s and often it is the bandwidth that is the issue.

Line 45: An important “if” is mentioned here. Many climate models are not built to utilize the hardware of today.

Line 59-62: What is needed here. Do you need to change Veros? This means that the coupled system cannot be run in parallel?

Line 140: A2 shows that the bias is localized. Any idea why (besides the lower concentration/higher sea ice gradient)?

Sharded arrays are used as the solution but communication is still needed (If I understand this part correct). It would be beneficial to describe this a bit more as it is central to why it works, how efficient is it?

Line 190 to 195. How is the ocean initialized? Is it realistic? The run starts on the 1/1 and the plot shows the 1/7. Is that after 6 months? The text states that a one year solution is carried out? I am not entirely sure what figure 2 demonstrates except that the system runs and that it produces sea ice.

Line 202: The just in time compilation is a reason for the performance improvement. I think that it is important to state how significant this overhead is. Is it only timings of the dynamics that are used or is it the full model that runs only dynamics (including I/O)?

Section 3.2.2:

It is not entirely clear for me how one GPU is specified. How is this comparable to one CPU?

How do you specify different number of GPU processes? A GPU processor normally comes with “many” cores/threads that share the workload. Are you sure it is the communication? GPU’s likes to have a high throughput of data. The larger grid likely facilitates this.

Section 3.2.3: The CPU nodes do not utilize the full number of processors in any of the settings. This requires running more nodes, hence energy. Is this due to lower performance? This also means that a lot of the processers idle. Besides the point that the GPU is more energy efficient in this case it also indicates that the CPU hardware is not suited for the problem at hand or the code is not suited for the CPU hardware. The most efficient utilizes 1/8 of the cpu cores allocated.

Table 1: The CPU is listed as both a cpu and a gpu. Is that a mistake? Are both used?

Figure 2: Yes, it creates ice but is it realistic? I don’t think that the discussion of integration with Veros belongs in the result section as there is no real result (except formation of sea ice).

Figure 3: I cannot see the color difference between the two Veris cpu runs. Please modify