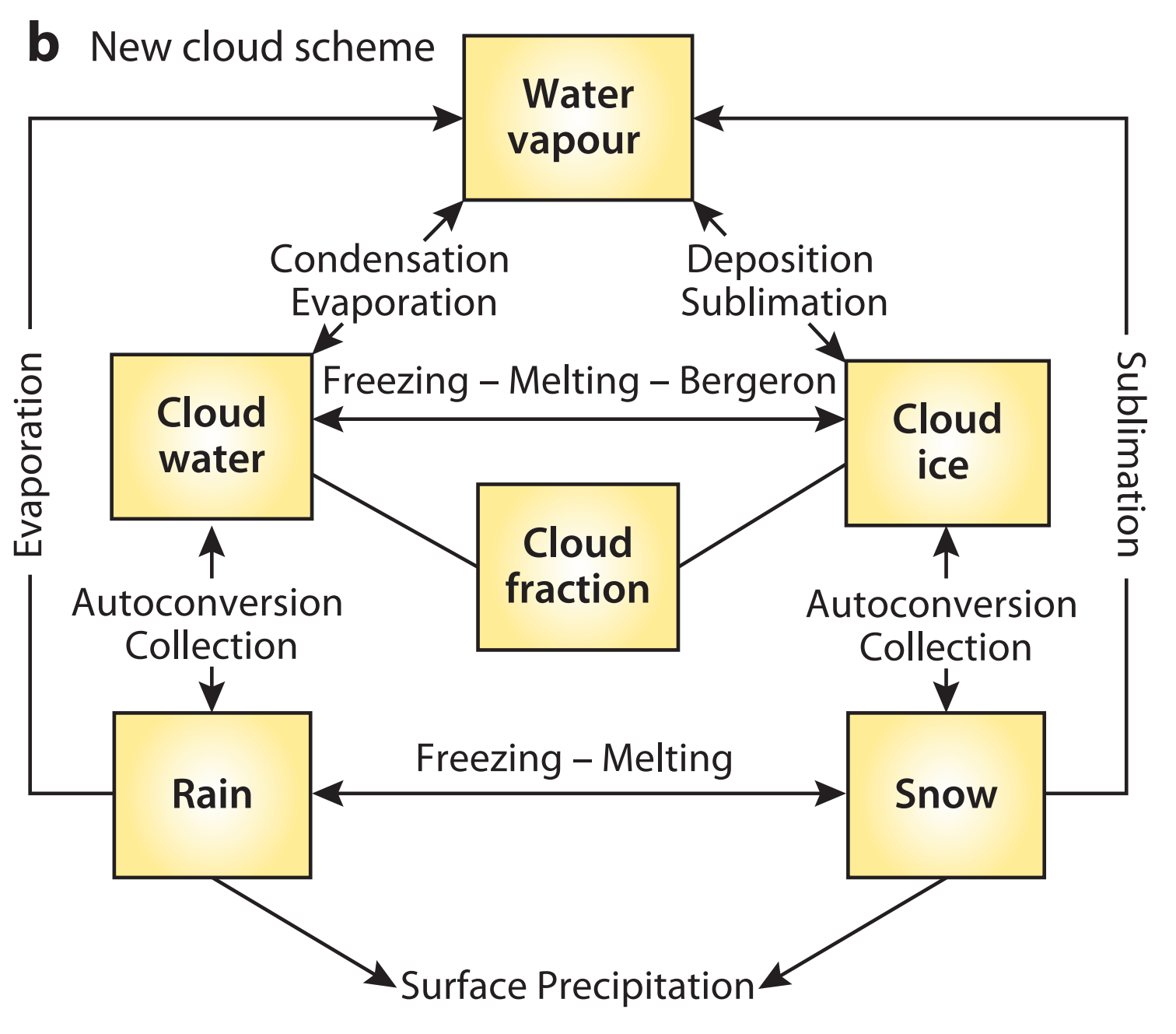
Cloud microphysics dwarf extensions: interfacing to Python and Loki GPU port of the Nonlinear/Tangent Linear/Adjoint variants.

Zbigniew Piotrowski (zbigniew.piotrowski@ecmwf.int), ECMWF, Bonn, Germany

1. Introduction

Porting of the ECMWF operational software stack to modern accelerators is a key for keeping the lead in the medium-range forecasting. Two most common porting approaches focus either on code rewrites using modern hardware-agnostic environments or enabling directive-based accelerator offloading directly or with the use of automated tools. ECMWF undertakes both approaches, yet in different contexts. The first approach is represented by the alternative IFS non-hydrostatic finite-volume dynamical core FVM. FVM is implemented using the Python library GT4Py for generating high-performance implementations of stencil kernels for weather and climate modeling from a domain-specific language (DSL). The second approach is being gradually employed to the Fortran spectral IFS software stack, in particular employing Loki source-to-source translation tool.

For the investigation of the code development and modernization ideas, ECMWF relies in part on the mini-benchmark applications called dwarfs, consisting of simplified, extracted IFS components that are nevertheless at least in part representative for the IFS software stack. A notable example of such dwarf is the CLOUDSC/CLOUDSC2 dwarf embedding IFS cloud microphysics scheme, where CLOUDSC2 is further simplified version of CLOUDSC to facilitate a wider range of experimentation, including tangent linear (TL) and adjoint (AD) versions.



CLOUDSC dwarf physical process scheme

2. Exploring Python-Fortran coupling for IFS-FVM

The process of the development and performance evaluation of the non-hydrostatic dynamical core for global weather usually requires – if any – only fundamental physics modules. However, further steps towards prospective operational use require connecting to a complete set of physical parameterisations. DSL-oriented rewrite of physics is expected to ultimately deliver an optimal performance, however, it is time consuming and leads to tedious-to-maintain dual Python-Fortran codebase. Seeking for an interim solution, a semi-automated solution offered by the f2py/f90wrap was evaluated for the CLOUDSC cloud physics dwarf at ECMWF.

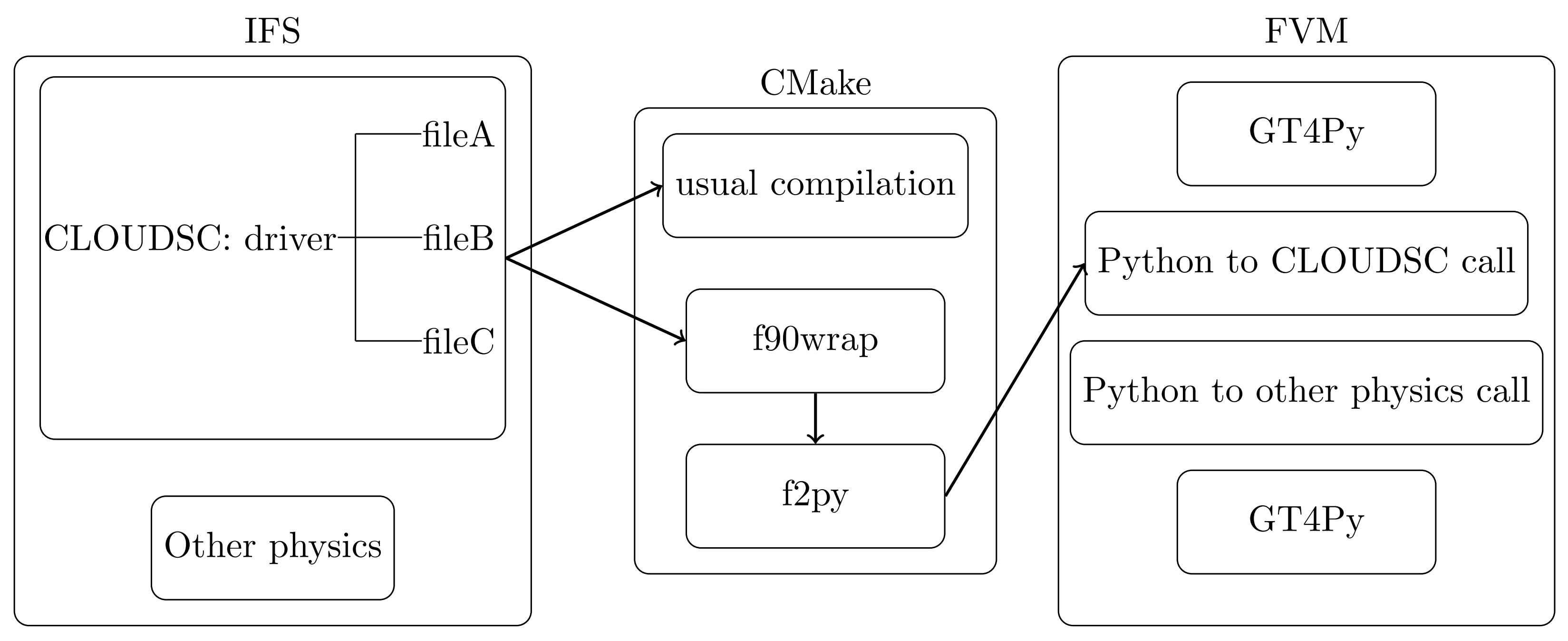


Diagram exposing relation between IFS-spectral and IFS-FVM code stacks. The IFS physics Fortran sources - following the usual build process - are processed further with the third party f90wrap/f2py tools to produce Python module with a corresponding dynamically loaded library.

Steps for connecting Python and Fortran code:

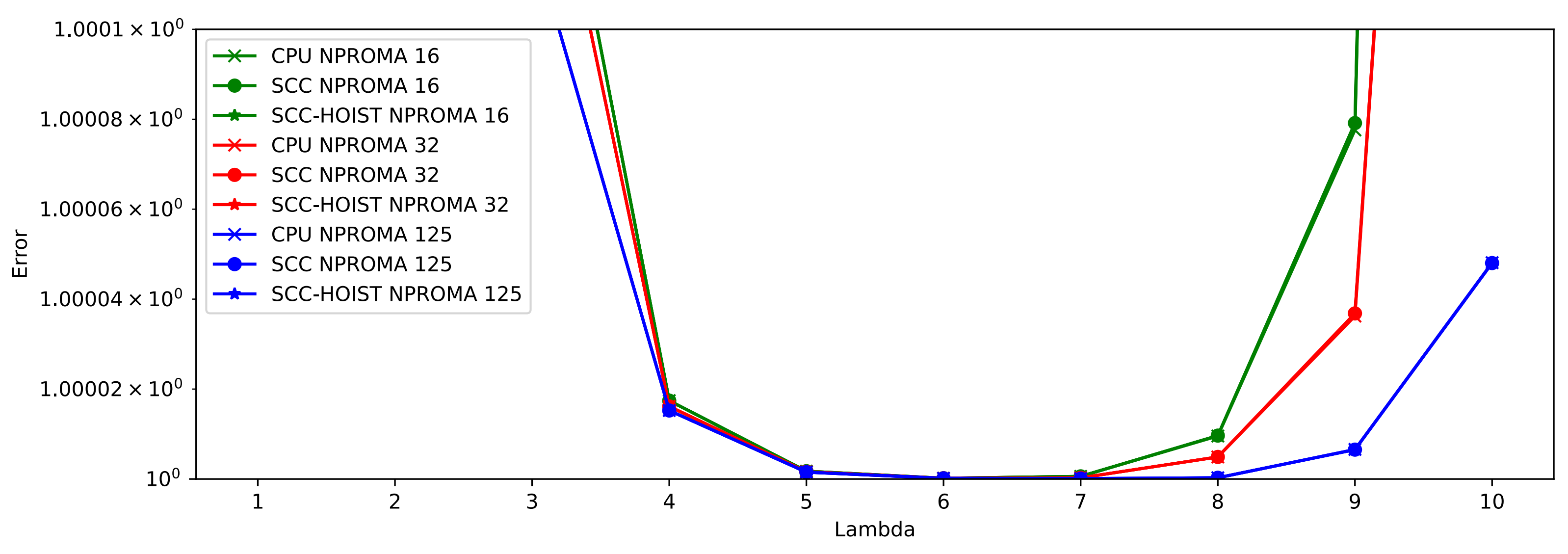
- Identify Fortran memory structures, ensure Python ownership of all non-temporary data (Fortran arrays may be overwritten after leaving the Fortran subprogram).
- Remove dependence on the non-contiguous memory structures, eg. in derived types.
- Equip build process with Python virtual environment, providing the Python dependencies.
- Generate Fortran wrappers using f90wrap.
- Generate Python-loadable .so library using f2py.

Caveats:

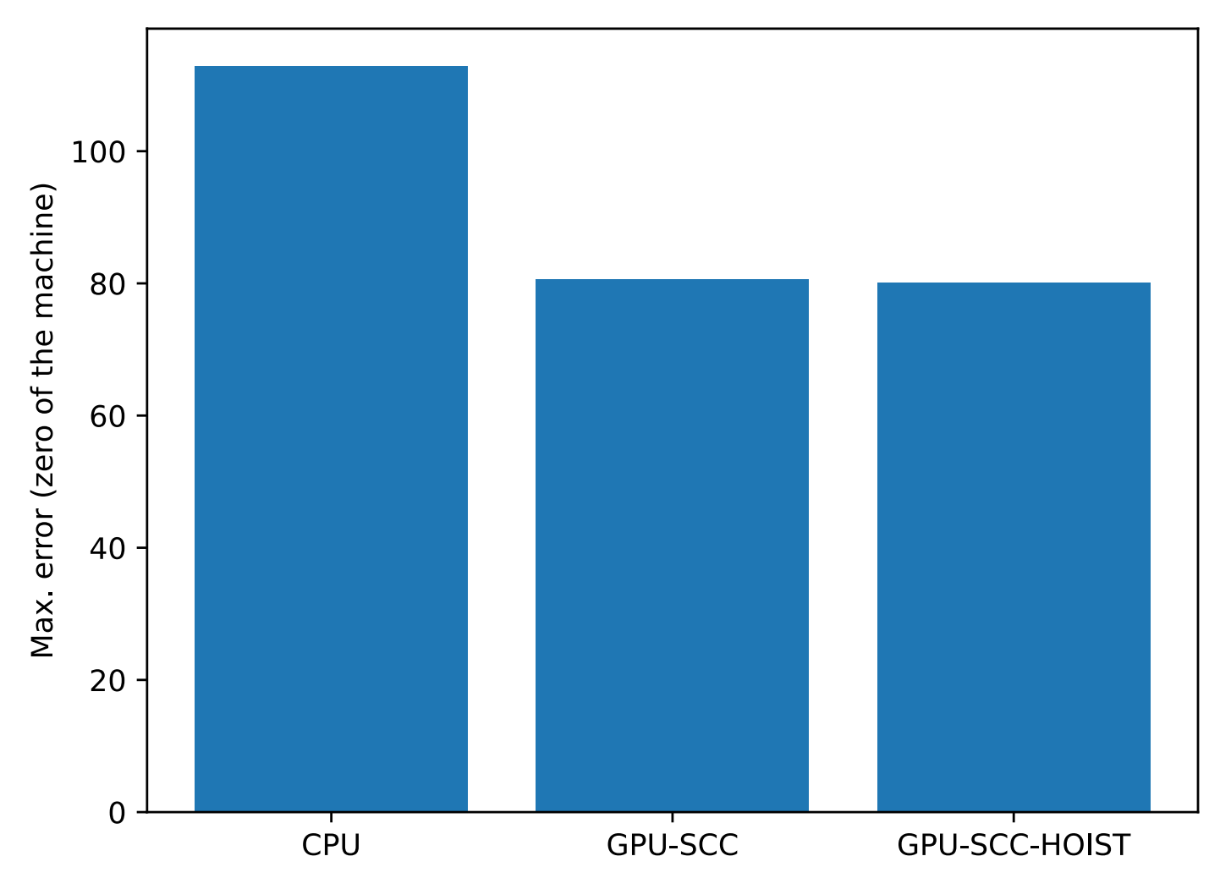
- Working in two (0- and 1-based) indexing regimes of Python and Fortran may easily become an extra effort for the developers.
- f2py relies for the dynamical library packaging step on the numpy.distutils, which is now deprecated. It is unclear what will serve as a replacement for this functionality in the Fortran context.
- Performance of the Python-invoked Fortran code is inferior due to apparent lack of control over the compiler optimizations in the f2py step.

3. Is the GPU accurate enough for the tangent-linear/adjoint computation ?

Using accelerator-based HPC hardware for the weather suite would be difficult without the possibility of running state-of-the-art data assimilation. However, experimental evidence shows that fine numerical details of the realization of computations may impair physical models that rely on satisfying the algebraical identity $AA^{-1}=1$. To investigate if the CLOUDSC2 NL/TL/AD dwarf triad computes properly on GPU, the TL and AD variants were ported to NVIDIA GPUs using Loki source-to-source translation tool (--> see the poster of Reuter et al.). The implementation effort was straightforward and required only a mild adaptation of the LOKI software stack.

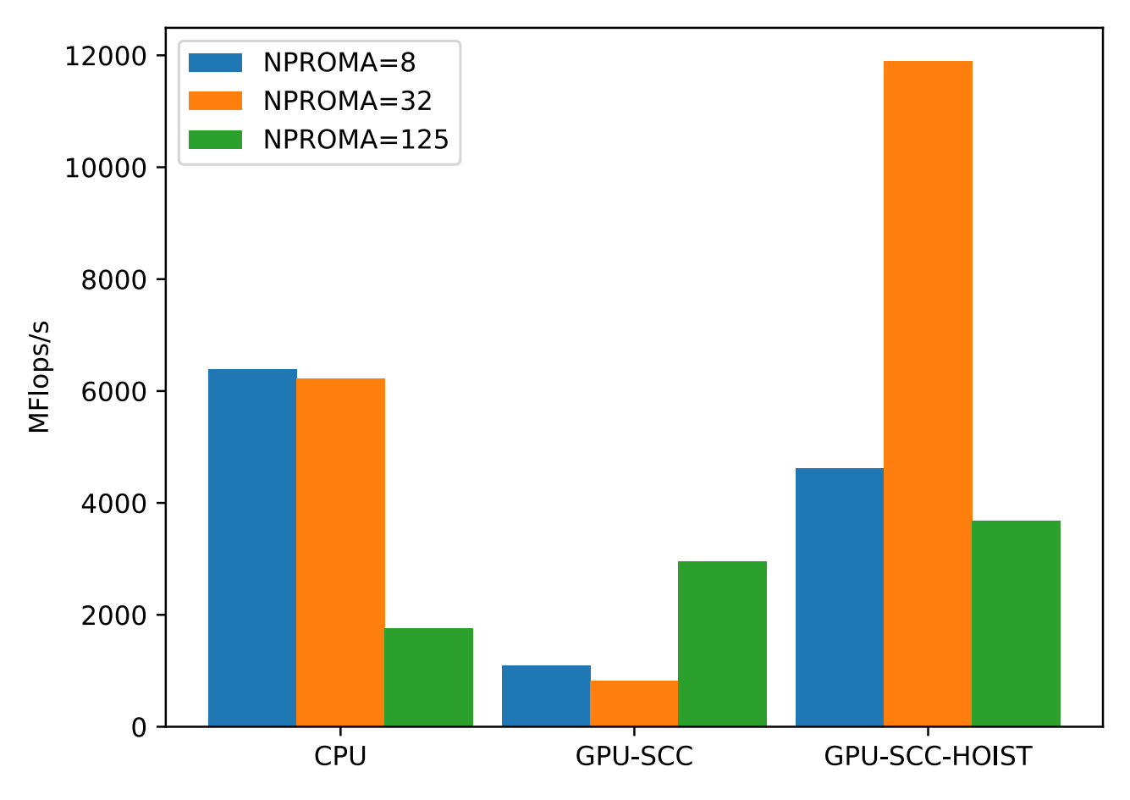


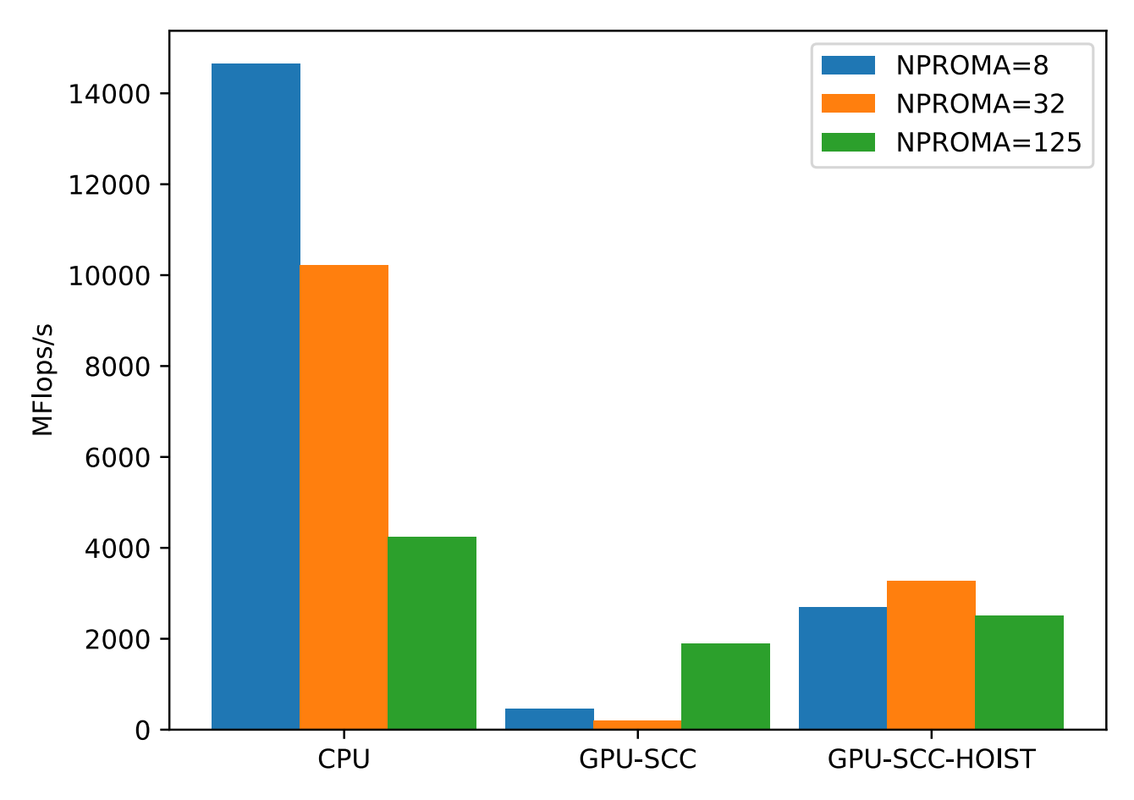
Error norm for CPU and GPU: SCC and SCC-HOIST TL variants.



Maximum error of the AD test for the CPU and GPU: SCC and SCC-HOIST variants.

For the tangent linear model, error norm for CPU and GPU variants are nearly identical, regardless of the NPROMA (i.e. blocking parameter) choice. In turn, the adjoint model test reveals that the GPU-specific maximum error (in terms of zero of the machine) is about 71% of the corresponding value on CPU.





Current computational performance, based on the raw (i.e. unoptimized) TL/AD port, varies on the level of code complication at hand. The AD model, which is the most complex, demands optimization effort, as the 64 CPU threads (run on the 2x EPYC 7742 node) offer significantly more performance. In turn, the TL model with a proper NPROMA choice may easily offer double the CPU performance.

4. Conclusions

- Task of merging Python and Fortran codes is non-trivial and within the f90wrap/f2py-based solution requires further investments. Alternatives, e.g. source-to-source processing or temporary dual codebase should still be considered.
- CLOUDSC2 NL/TL/AD testbed can be computed accurately using the OpenACC GPU port provided by Loki. This suggests that such route to port components of IFS data assimilation is open. Due to the code complexity, a separate and careful insight into the computational performance will be necessary.

The work presented in this poster has been produced in the context of the European Union's Destination Earth Initiative and relates to tasks entrusted by the European Union to the European Centre for Medium-Range Weather Forecasts implementing part of this Initiative with funding by the European Union.

Funded by the European Union Destination Earth implemented by ECMWF esa EUMETSAT