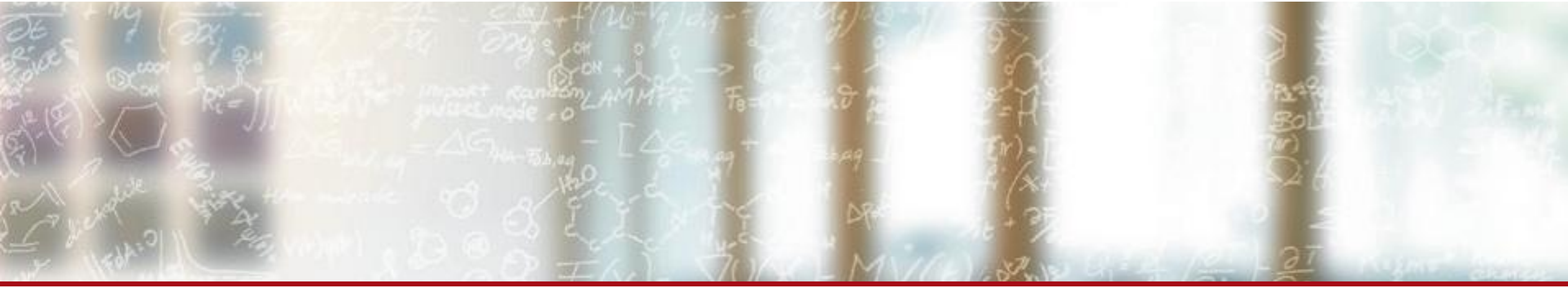
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

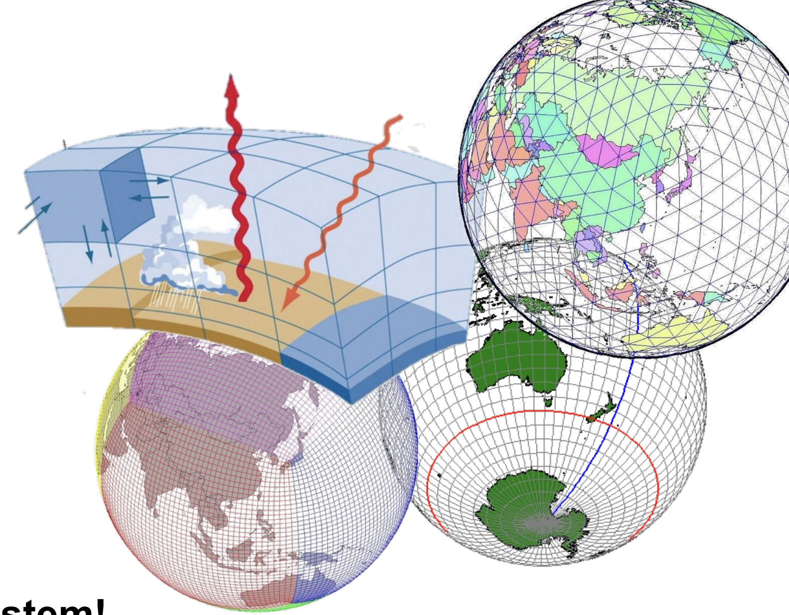


Embedding Weather and Climate Applications in the Python Ecosystem: the case of GT4Py

Presenting: **Mauro Bianco**

Hannes Vogt, Enrique Gonzalez Paredes, Till Ehrengruber, Rico Hauselmann, Linus Groner, Nicoletta Farabullini, Magdalena Luz, Yilu Chen, Felix Thaler, Christoph Müller, Edoardo Paone, Christos Kotsalos, Philip Muller, Ioannis Magkanaris, …

A bit of History and requirements



- 2010: initiative to run COSMO on GPUs
  - STELLA: **Model specific EDSL** library in C++
- 2014: started C++ GridTools
  - **Model agnostic**
  - **Performance portable**
- 2015: Started investigating **GT4Py**
  - A Python rendition of the GridTools concepts to
    - **From prototype to deployment**
  - **Integration with the evergrowing Python ecosystem!**
- **Requirements**
  - **Grid agnostic**
  - **Architecture agnostic**
  - **Model Agnostic**
  - **Different users with different responsibilities**

A C++ GridTools Example: Nabla4

```
struct lap_function {
    using out = st::cartesian::inout_accessor<0>;
    using in = st::cartesian::in_accessor<1, st::extent<-1, 1, -1, 1>>;

    using param_list = st::make_param_list<out, in>;

    template <class Eval>
    GT_FUNCTION static void apply(Eval &&eval) {
        eval(out()) = 4.*eval(in())-(eval(in(1, 0))+eval(in(0, 1))+eval(in(-1, 0))+eval(in(0, -1)));
    }
};
```

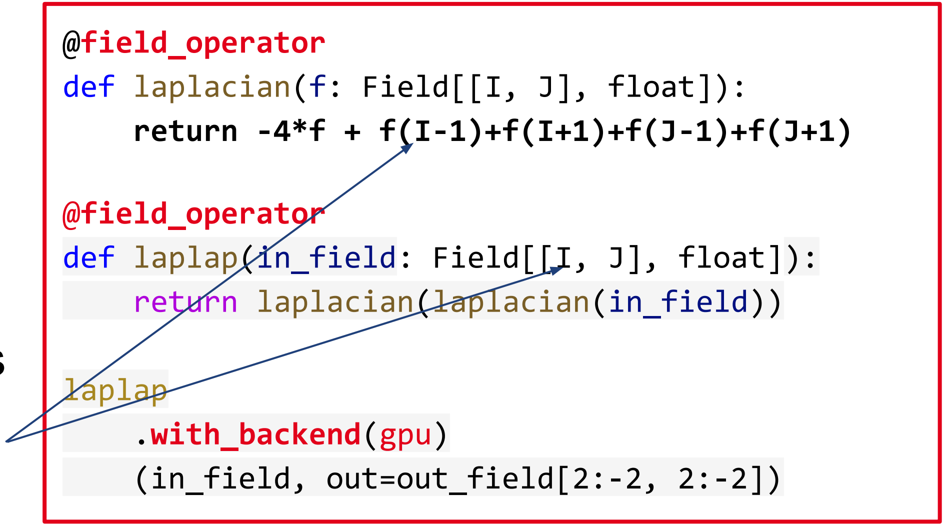
```
auto spec = [](auto in, auto out) {
        GT_DECLARE_TMP(double, lap);
        return st::execute_parallel()
            .ij_cached(lap)
            .stage(lap_function(), lap, in)
            .stage(lap_function(), out, lap);
    };
st::run(spec, stencil_backend_t(), grid, in, out);
```

- Customized function signatures
  - Happy users?
- Temporaries explicit
  - Playground not easy
- Data dependencies explicit by using symbols
  - I like it

GT4Py Computation Specs

- Individual discretized operators
- Operators can be composed
- Point-wise results are returned

- Strong typing makes code safer
  - Relaxing in the future with generics
- Users use the model specific entities
  - No judgement there…
  - Same symbol for axis and index/offset



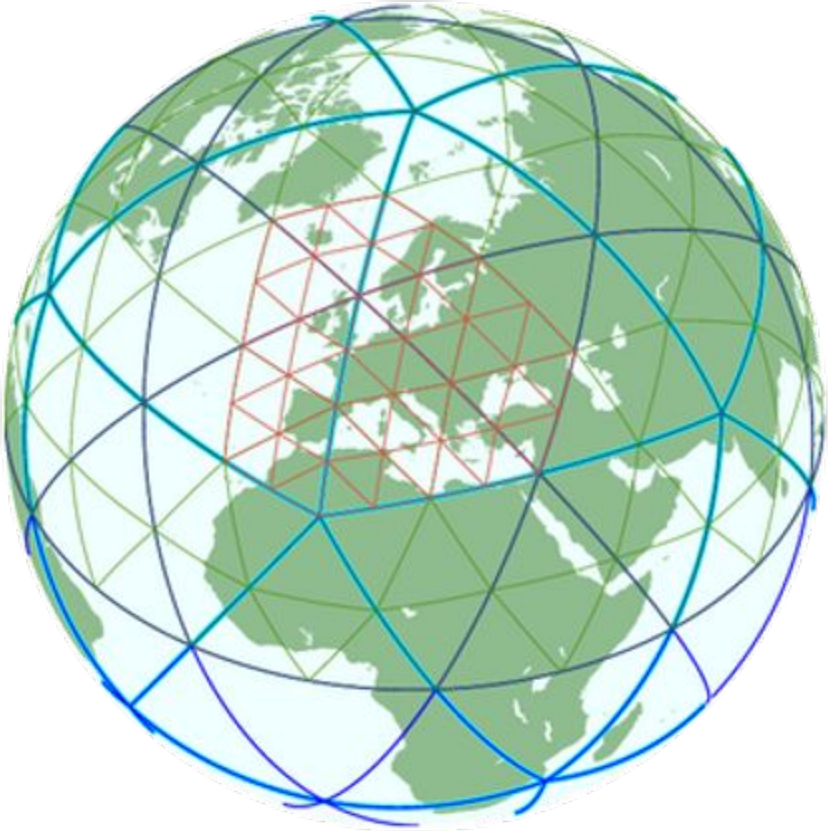
```
@field_operator
def laplacian(f: Field[[I, J], float]):
    return -4*f + f(I-1)+f(I+1)+f(J-1)+f(J+1)

@field_operator
def laplap(in_field: Field[[I, J], float]):
    return laplacian(laplacian(in_field))

laplap
    .with_backend(gpu)
    (in_field, out=out_field[2:-2, 2:-2])
```

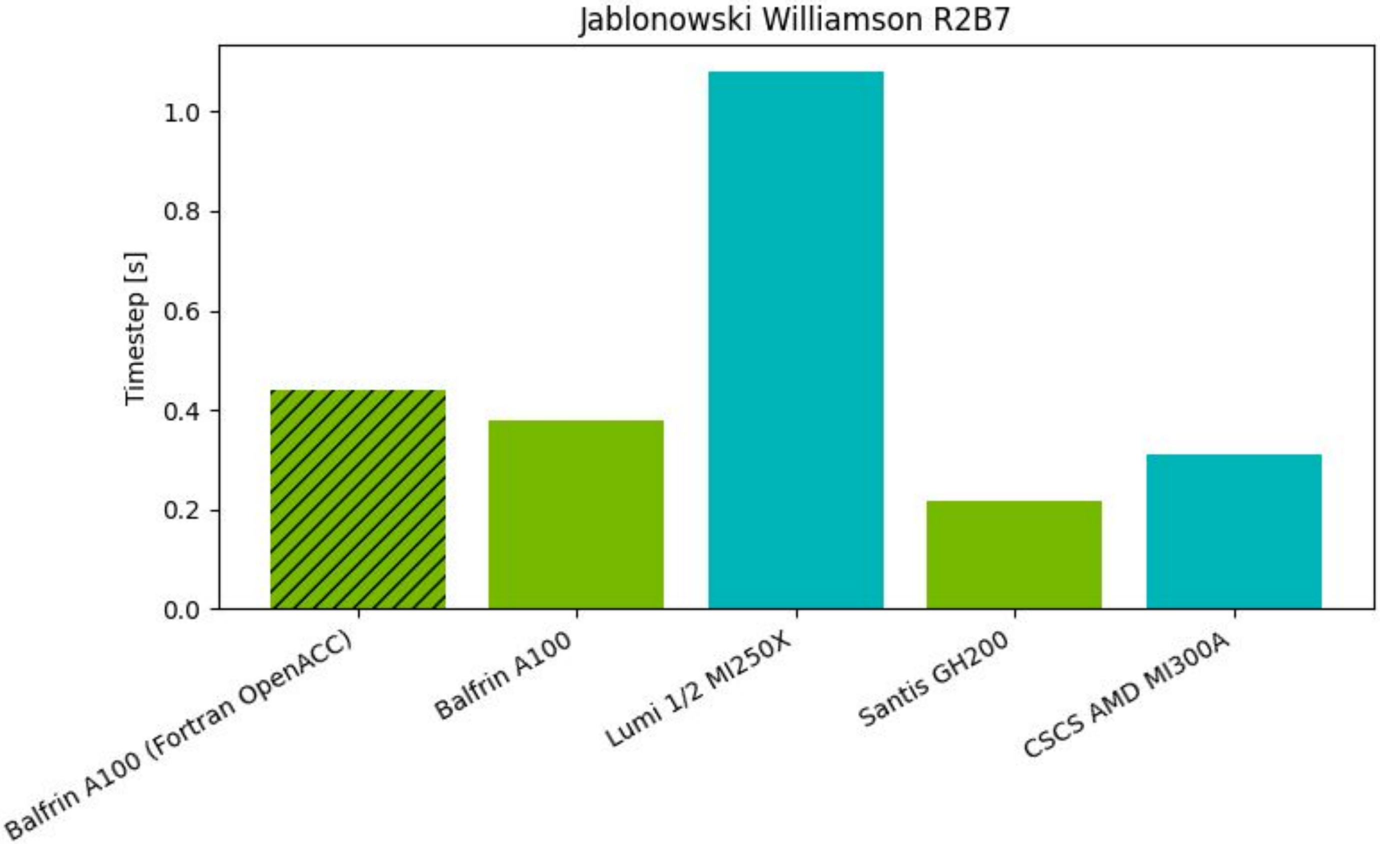
Non-Cartesian co-located grids makes life harder

Icosahedral Grid in ICON

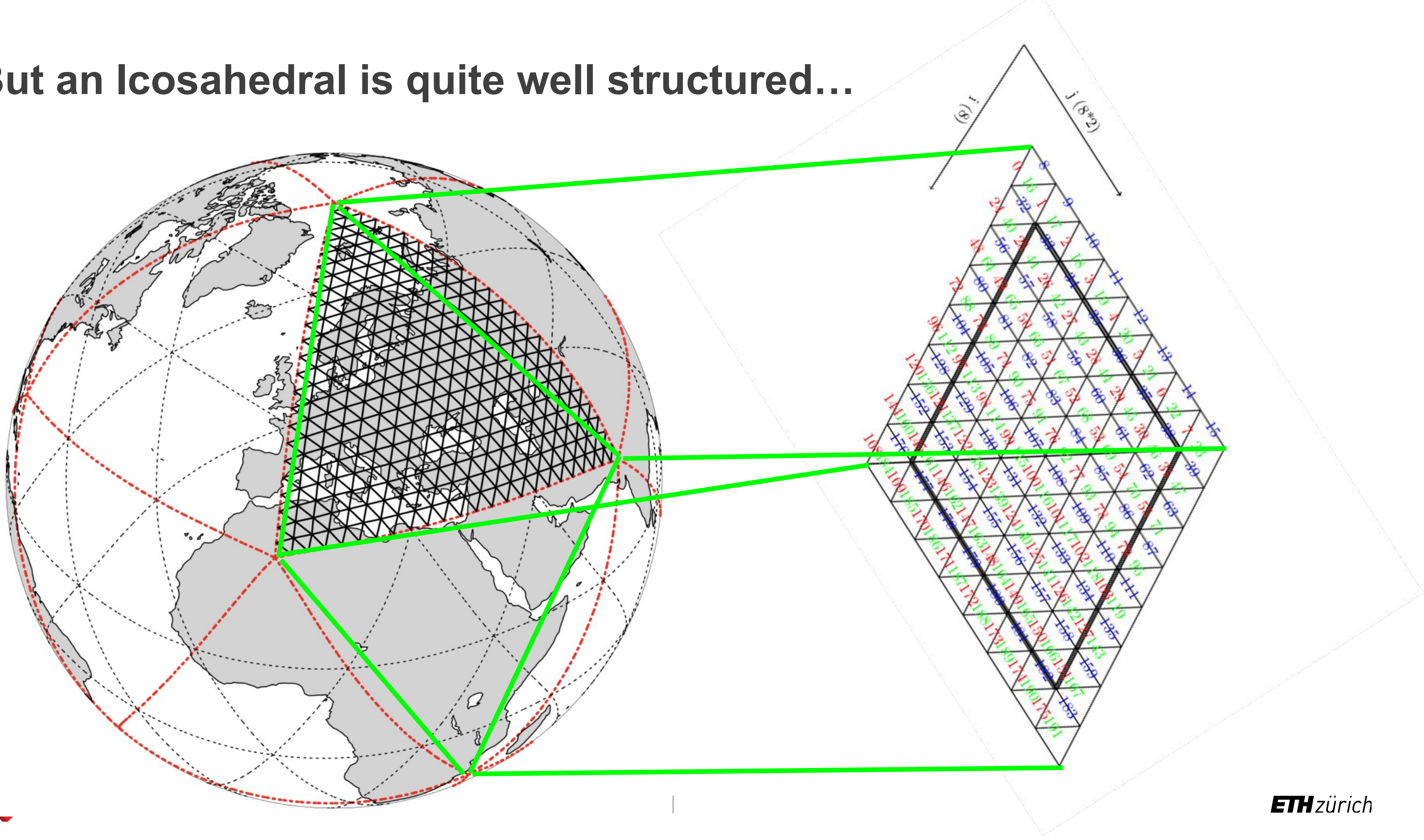


- ICON uses an icosahedral grid
- The implementation does not take advantage of the structure
  - The grid treated as unstructured
  - SFC on decomposed pieces
- Computations on location types:
  - Cells
  - Edges
  - Vertices
    - The final output is here

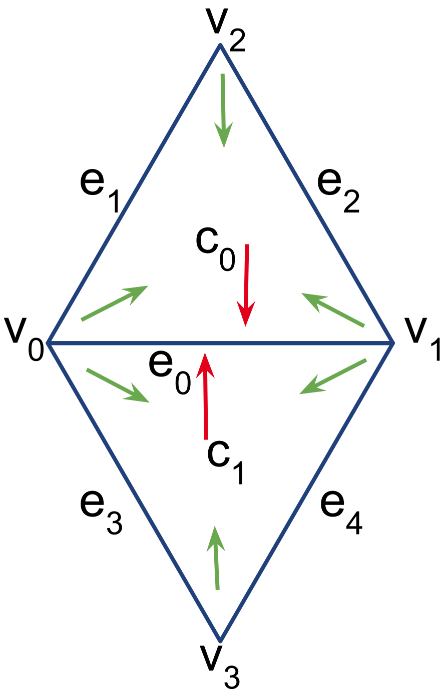
ICON dynamical core in GT4Py (initial results and portability)



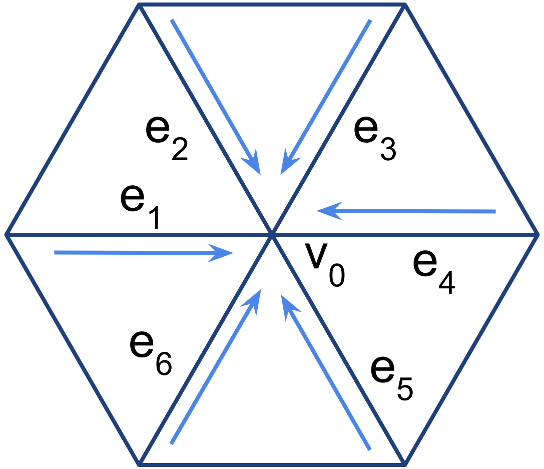
But an Icosahedral is quite well structured…

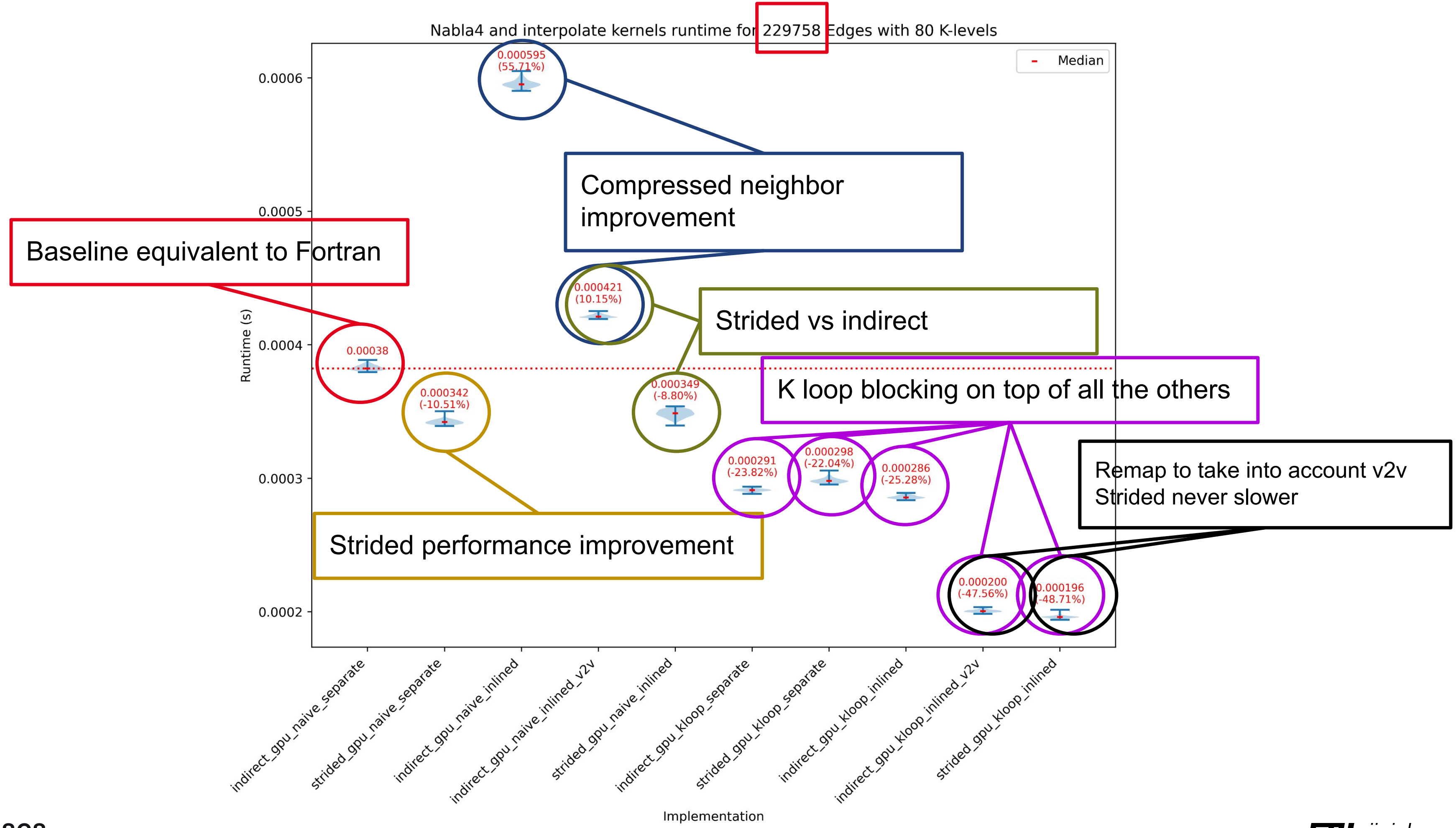


Nabla4 + Interpolation on ICON Grid



- Nabla4
  - Computation on e_0 needs values on c_0 and c_1
  - Values on c_0 needs values on v_0, v_1, and v_2
  - Values on c_1 needs values on v_0, v_1, and v_3
- Interpolation
  - Reduce on edges





Compressed neighbor improvement

Baseline equivalent to Fortran

Strided vs indirect

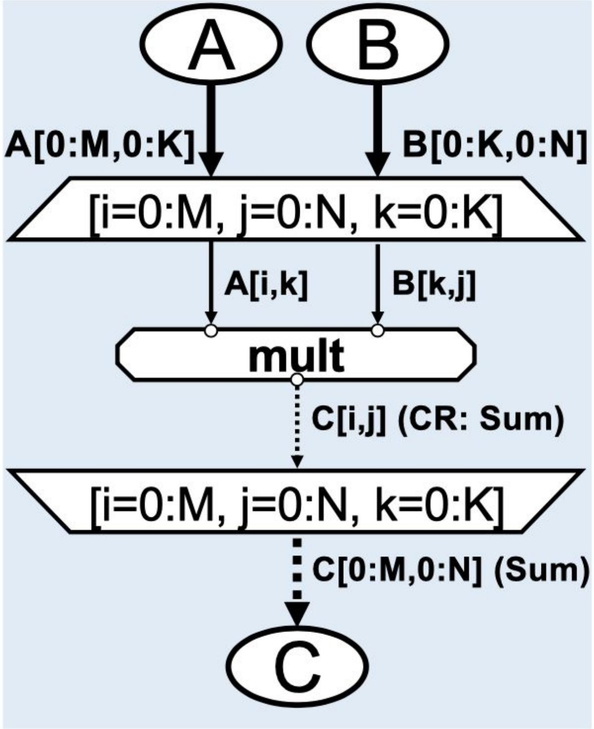
K loop blocking on top of all the others

Remap to take into account v2v
Strided never slower

Strided performance improvement

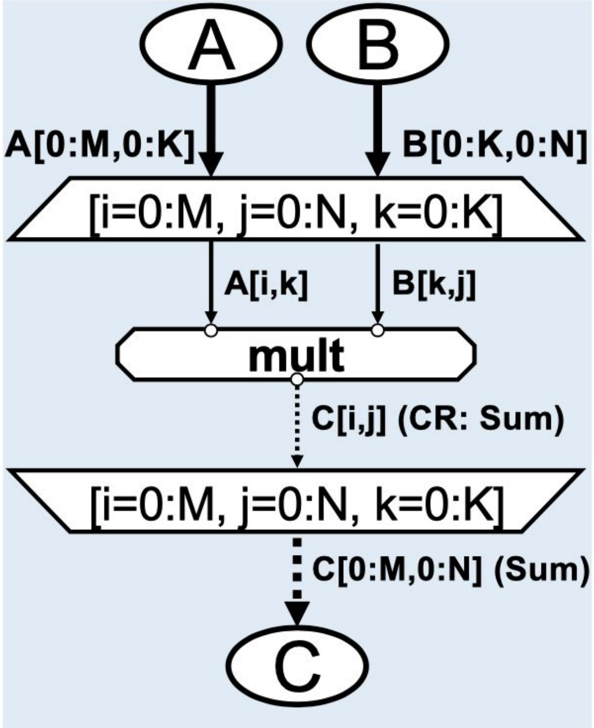
Backend technology

- Using DaCe
  - http://dace.is/fast
- Separation of concerns between
  - Application specialists
  - Performance experts
- Graphic and Programmatic interfaces
  - In IDEs such as vscode
  - Optimization codes are part of the repository



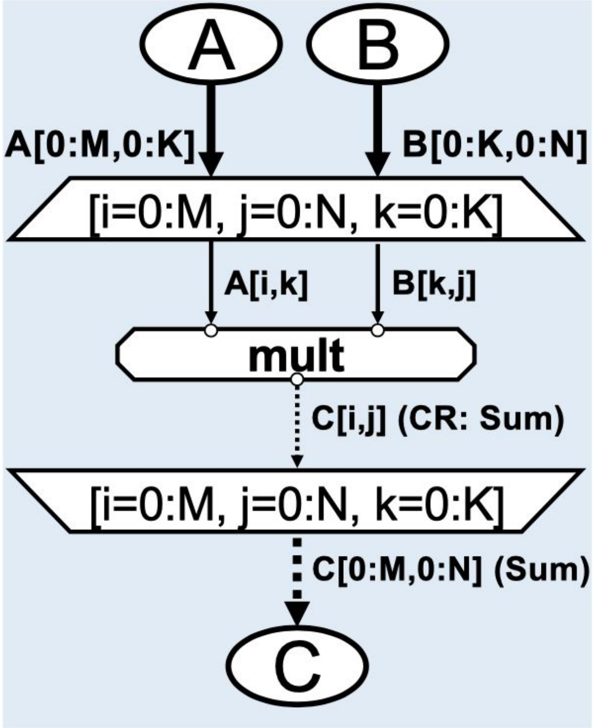
Backend technology

- Using DaCe
  - http://dace.is/fast
- Separation of concerns between
  - Application specialists
  - Performance experts
- Graphic and Programmatic interfaces
  - IDEs such as vscode
  - Optimization codes are part of the repository
- **Separation of concerns != isolation**

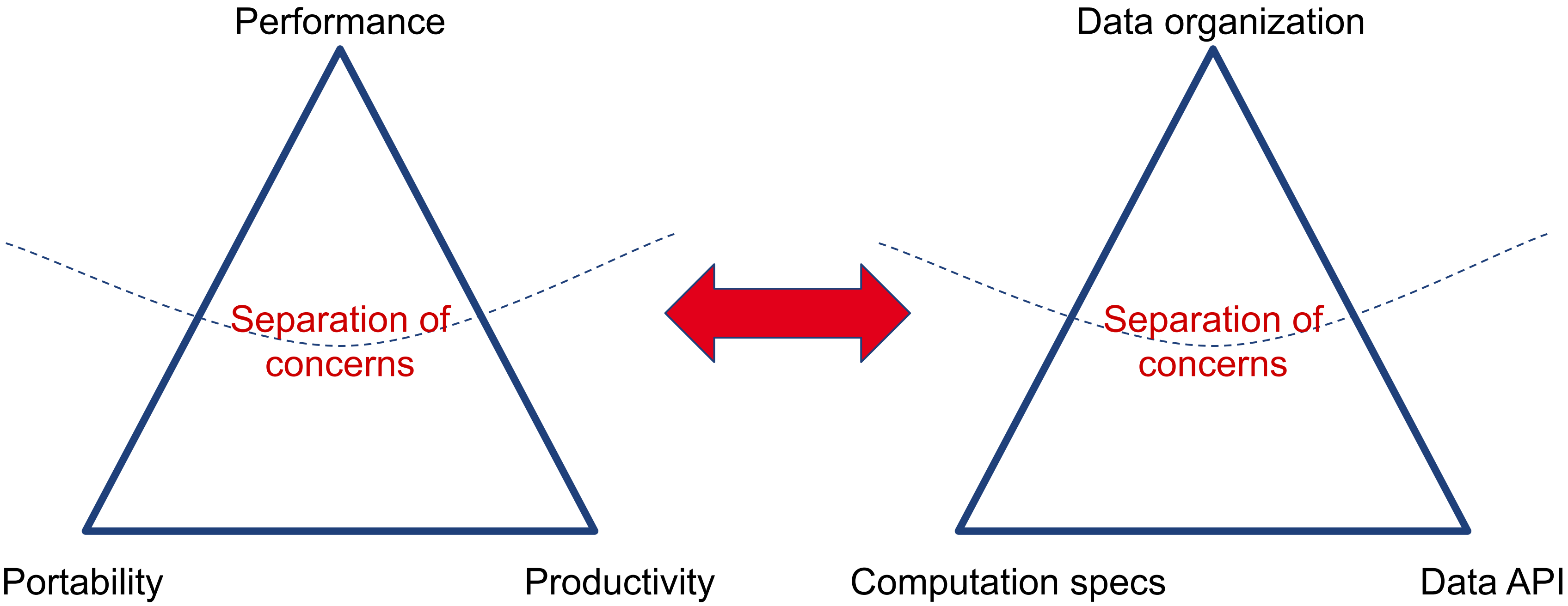


- **Separation of concerns != isolation**

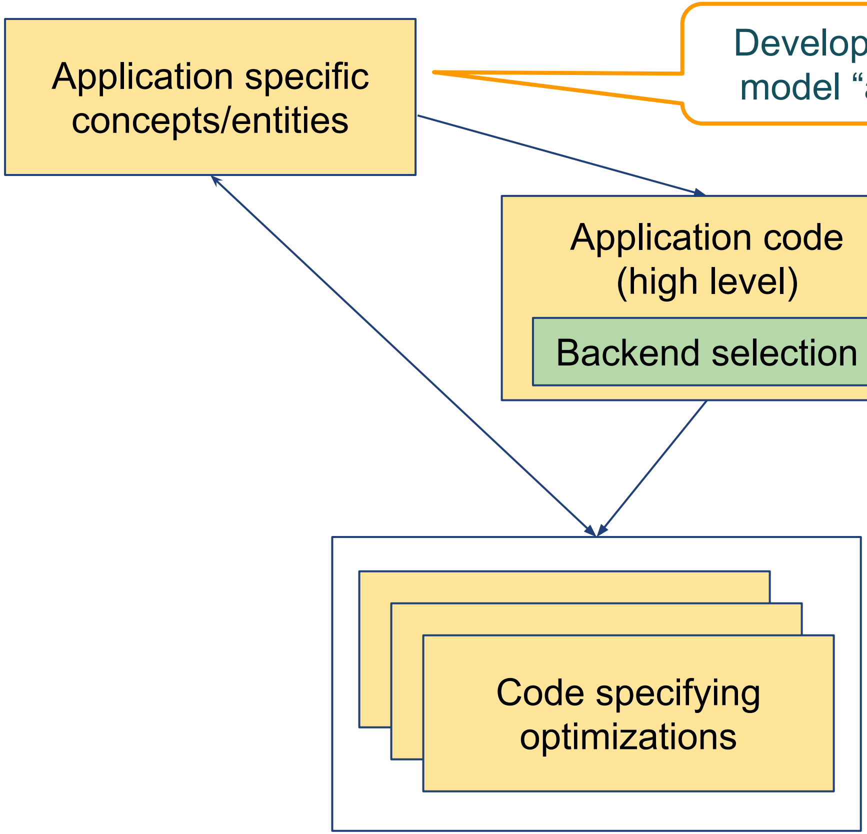
- **Beyond AoS and SoA**
- **Grid API toward the backend** needs to communicate the regions and the mapping
- The backend needs to manage different computations for different regions
- **The user code remains the same**



Secret Sauce



Anatomy of an application



Application specific concepts/entities

Application code (high level)

Backend selection

Code specifying optimizations

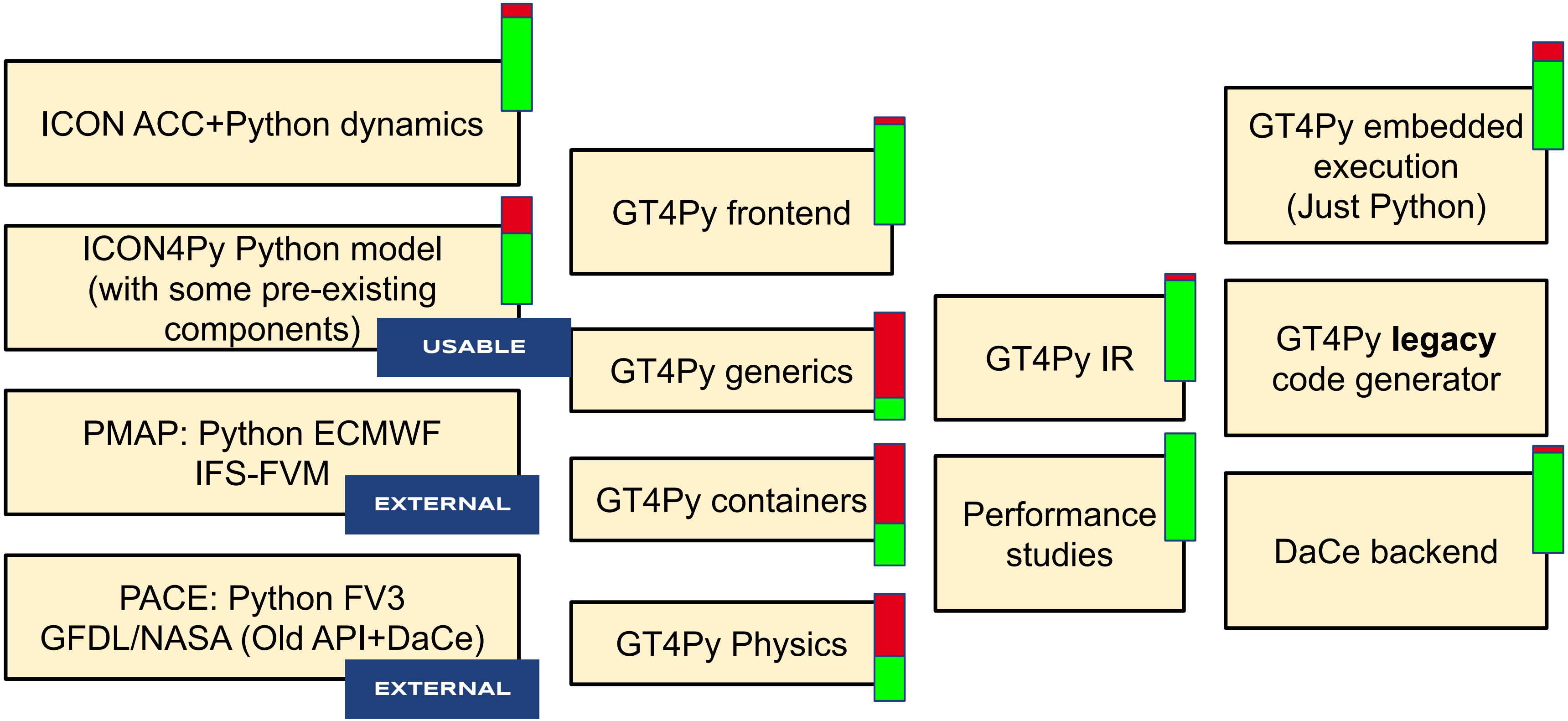
Developed by the model “architect”

Developed by scientists and source of truth

Developed by computer/performance experts

Architecture and possibly input specific

What are we working on



ICON ACC+Python dynamics

ICON4Py Python model (with some pre-existing components)

USABLE

PMAP: Python ECMWF IFS-FVM

EXTERNAL

PACE: Python FV3 GFDL/NASA (Old API+DaCe)

EXTERNAL

GT4Py frontend

GT4Py generics

GT4Py containers

GT4Py Physics

GT4Py IR

Performance studies

GT4Py embedded execution (Just Python)

GT4Py **legacy** code generator

DaCe backend

Thanks to

- Contributors and Partners

CSCS

SPCL

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Federal Department of Home Affairs FDHA
Federal Office of Meteorology and Climatolog
MeteoSwiss

MeteoSwiss

EXCLAIM

ECMWF

Ai2

- Weather and Climate models

CONSORTIUM FOR SMALL SCALE MODELING
COSMO

ICON

FVM for IFS

GFDL FV3

ASK ME ABOUT
SEPARATION
OF
CONCERNS
QWT HOW TO SEPRADIOIN OF COUNCEBNS