Tasmania: A stencil-oriented framework for performance-portable weather and climate applications in Python

S. Ubbiali[1], M. Bianco[2], E. G. Paredes[2], L. Groner[2], W. Sawyer[2], C. Schär[3], L. Schlemmer[4,5], T. C. Schulthess[1,2]

19th Workshop on High Performance Computing in Meteorology
20-24 September 2021

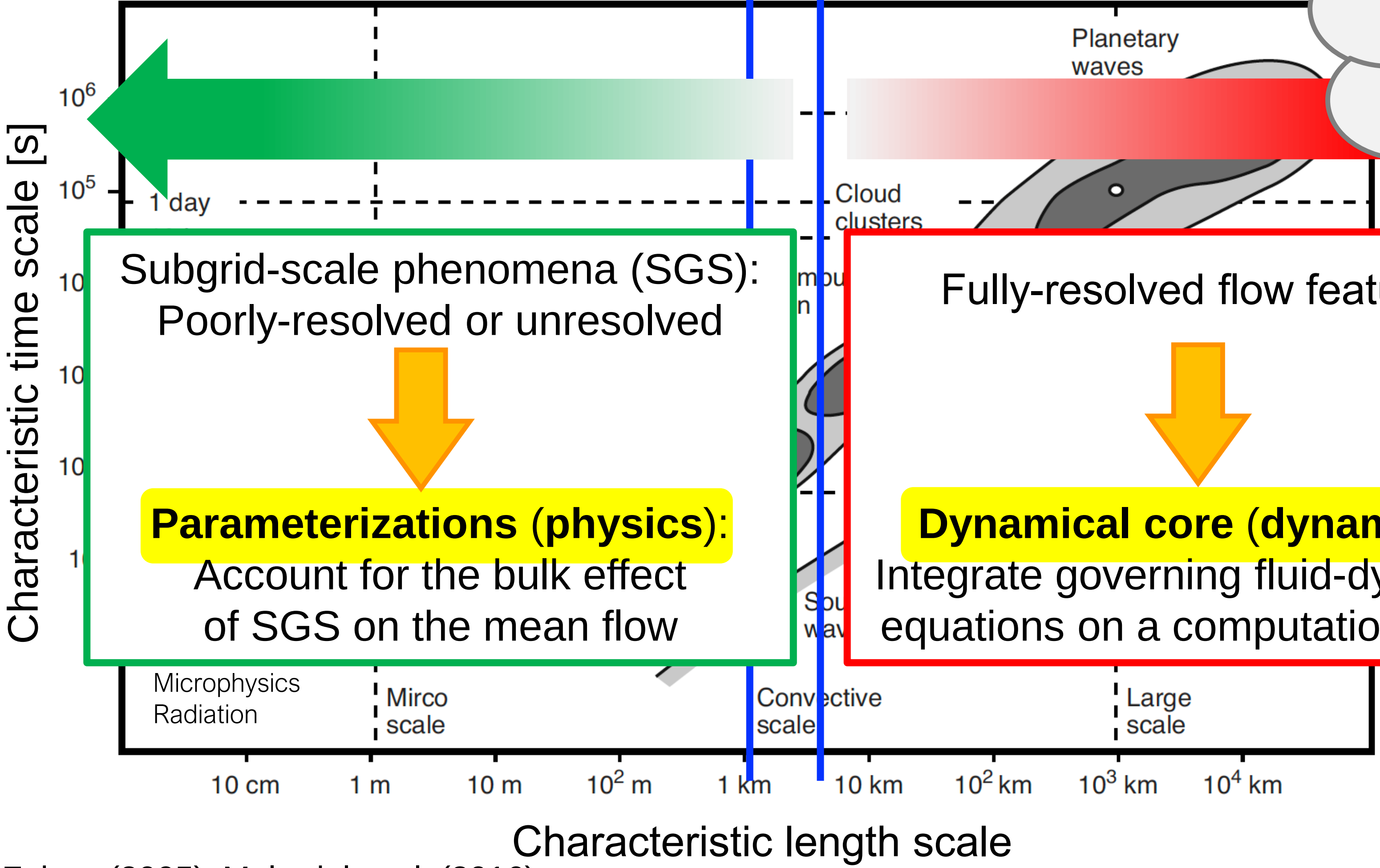
[1] Institute for Theoretical Physics, ETH Zürich, Switzerland, [2] Swiss National Supercomputing Centre, Switzerland
[3] Institute for Atmopsheric and Climate Science, ETH Zürich, Switzerland, [4] Deutscher Wetterdienst, Germany

Atmospheric models: In a nutshell

Nominal resolution
$\Delta x = 1\ \mathrm{km}$

Effective resolution
$\Delta x = 2 - 4\ \mathrm{km}$

Research question:
Which is the impact of the **physics-dynamics coupling** on high-resolution (km-scale) weather and climate models?



Subgrid-scale phenomena (SGS):
Poorly-resolved or unresolved

Parameterizations (physics):
Account for the bulk effect of SGS on the mean flow

Fully-resolved flow features

Dynamical core (dynamics):
Integrate governing fluid-dynamics equations on a computational grid

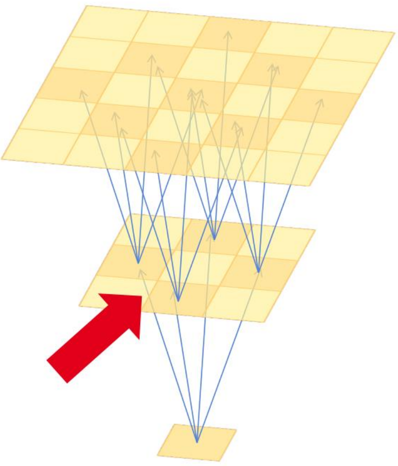
Model components

Fuhrer (2005), Malardel et al. (2016)

Introduction

We present the Python library Tasmania which serves three main goals.

- Extending the **Sympl** framework to write **inter-operable** and **self-documenting** components to build **flexible**, **modular** and **maintainable** models.
  - Breeding ground to assess the process coupling and the physics time-stepping.
- Exposing facilities to write (**performance-portable**) stencil codes.
- Probing the feasibility of Python as the host programming language for the next-generation of models.



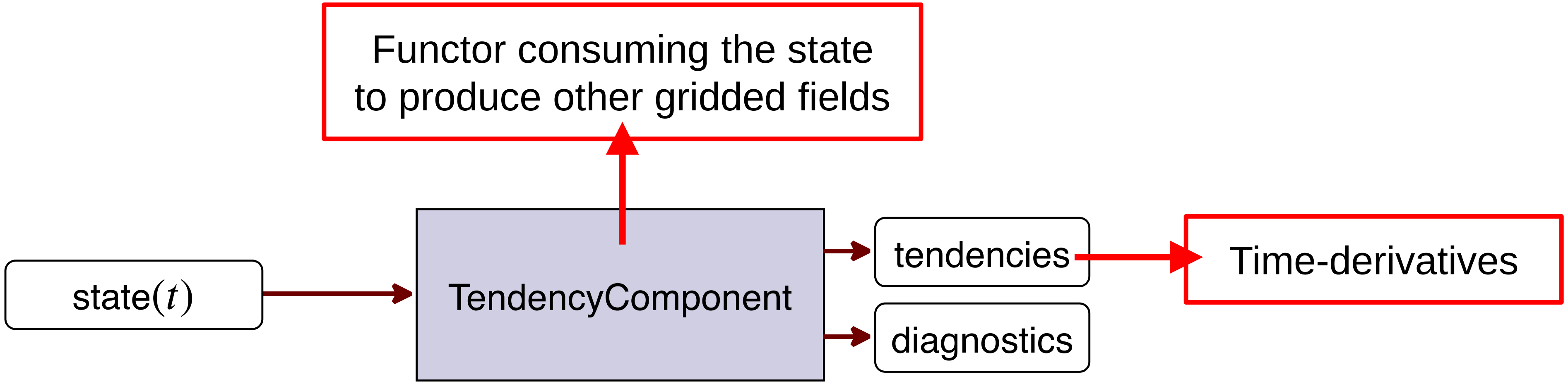
Monteiro et al. (2018), Schär et al. (2019); E. G. Paredes

Components: Physics

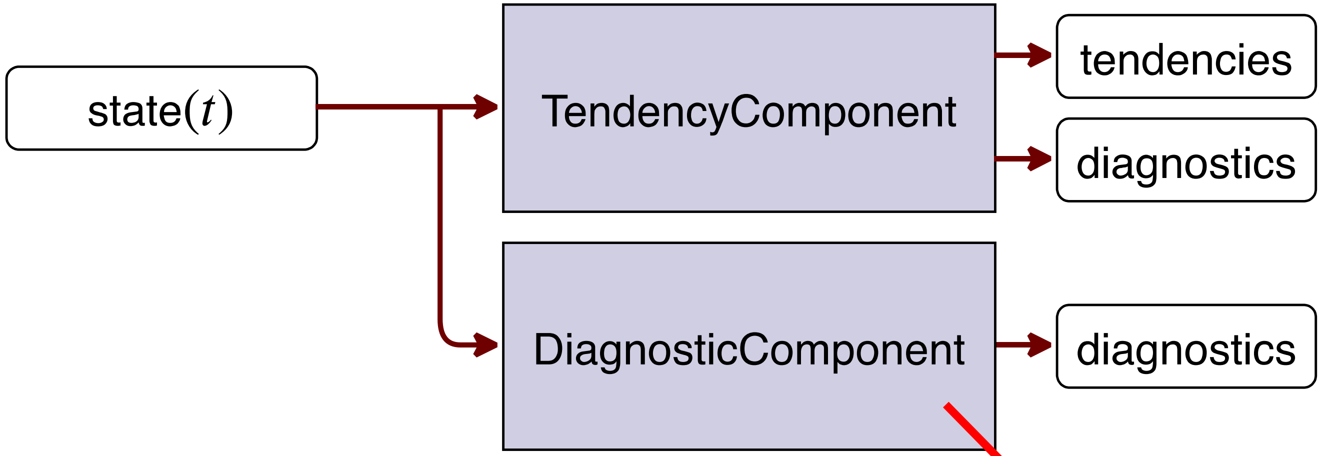
Dictionary of **DataArrays** storing grid-point values and metadata for atmospheric variables

state(t)

Components: Physics

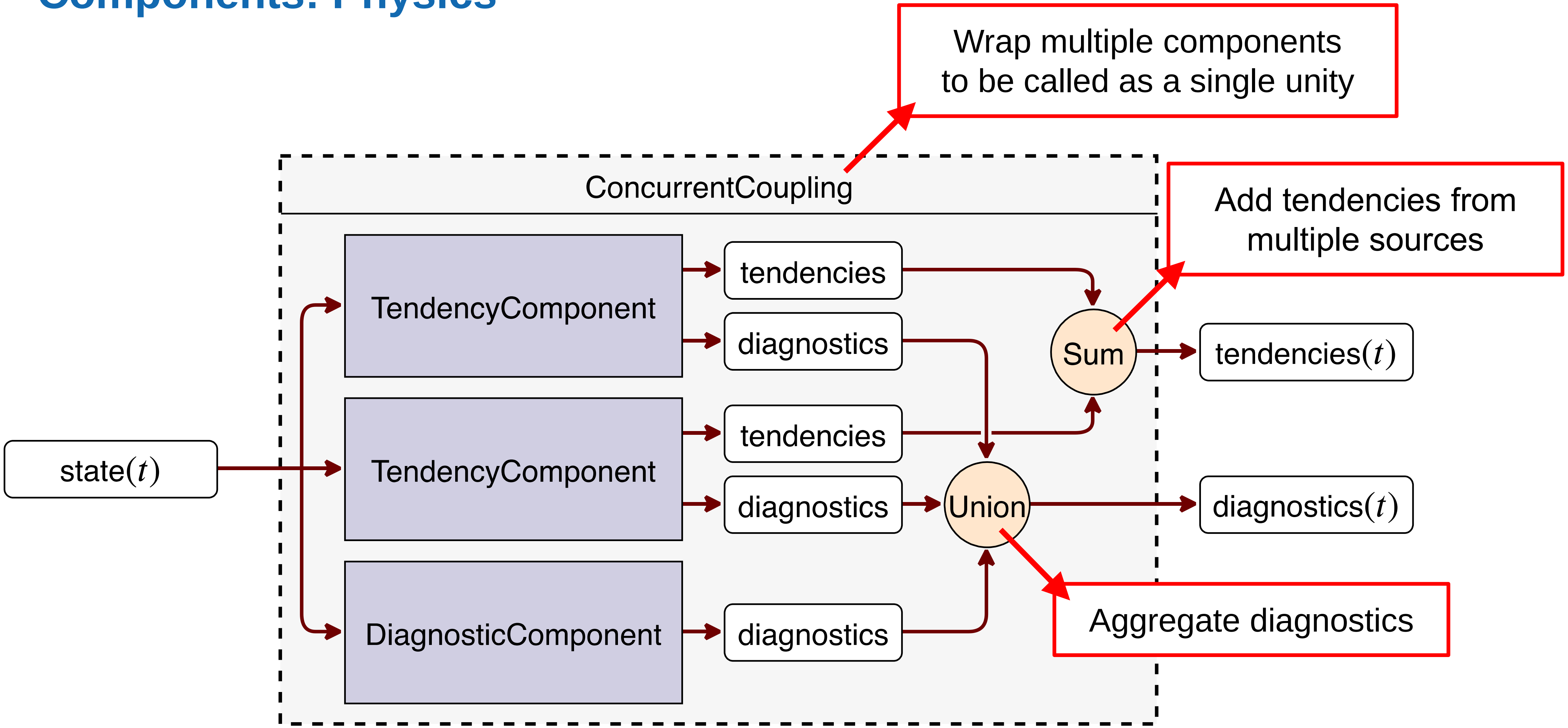


Components: Physics



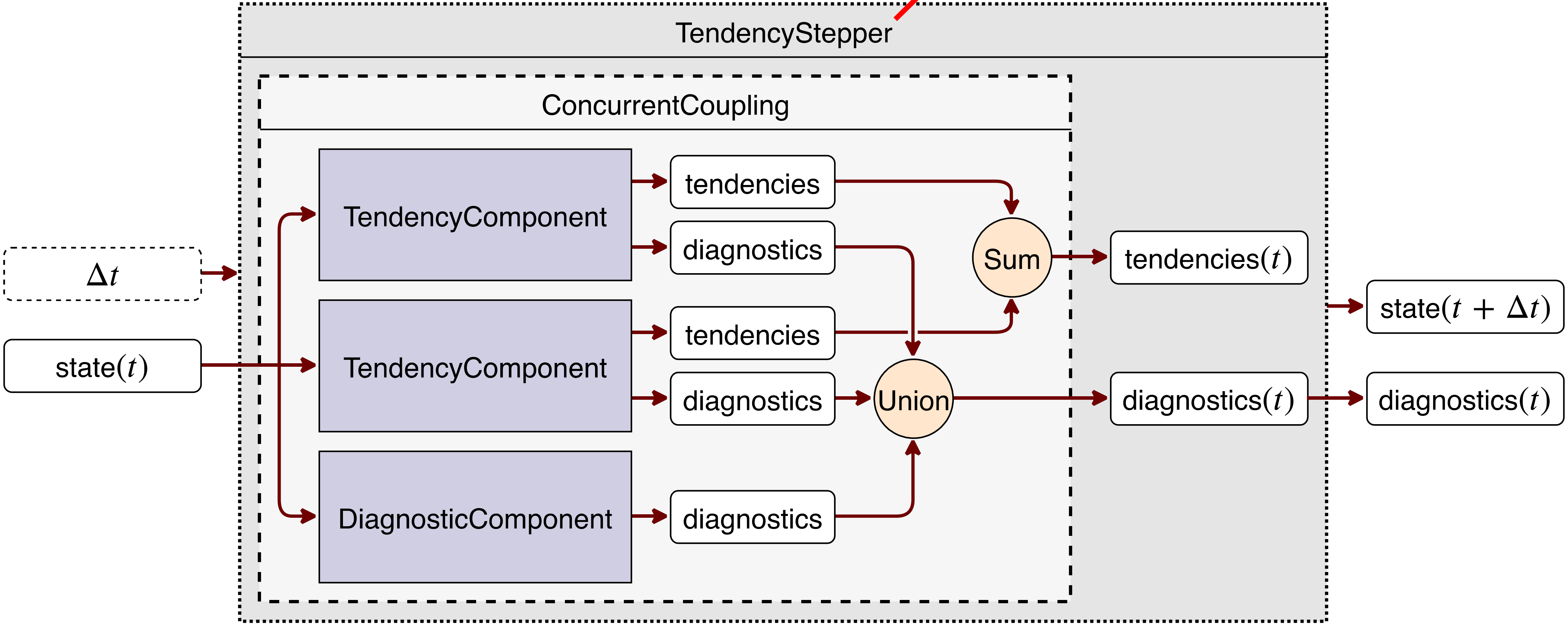
Stripped-down version of TendencyComponent

Components: Physics



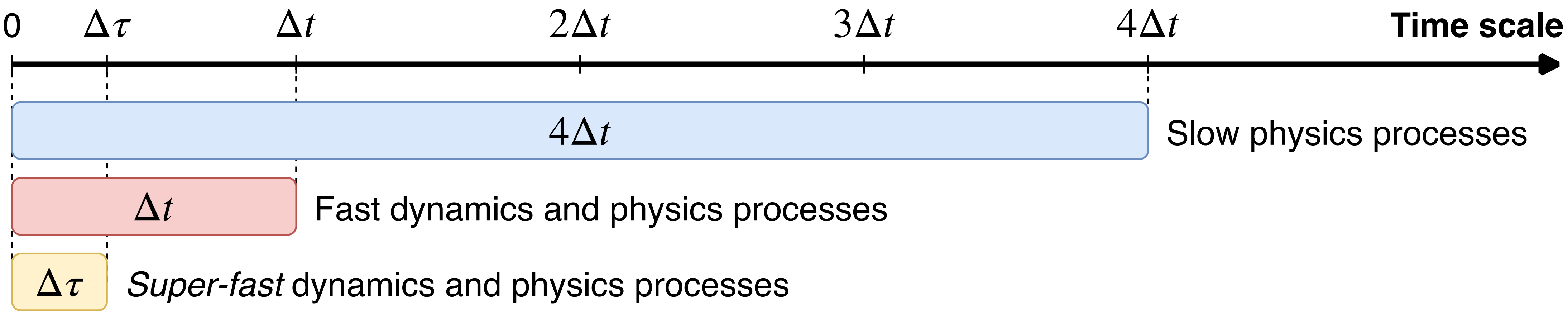
Components: Physics

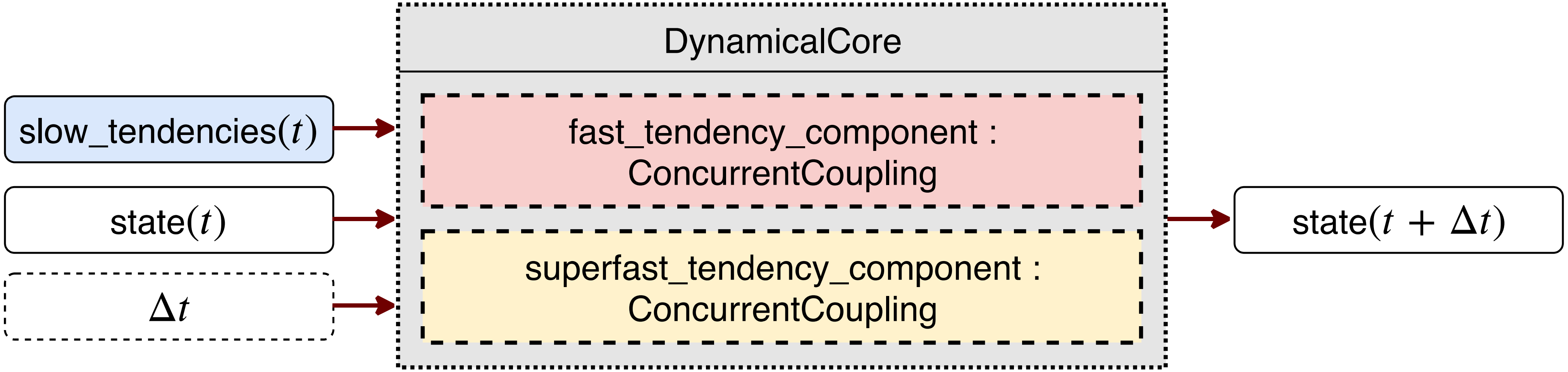
Use the tendencies to advance the state to the next time-level



Components: Dynamics

The class **DynamicalCore** supports **partial-time splitting**: Treating processes with different time-steps.

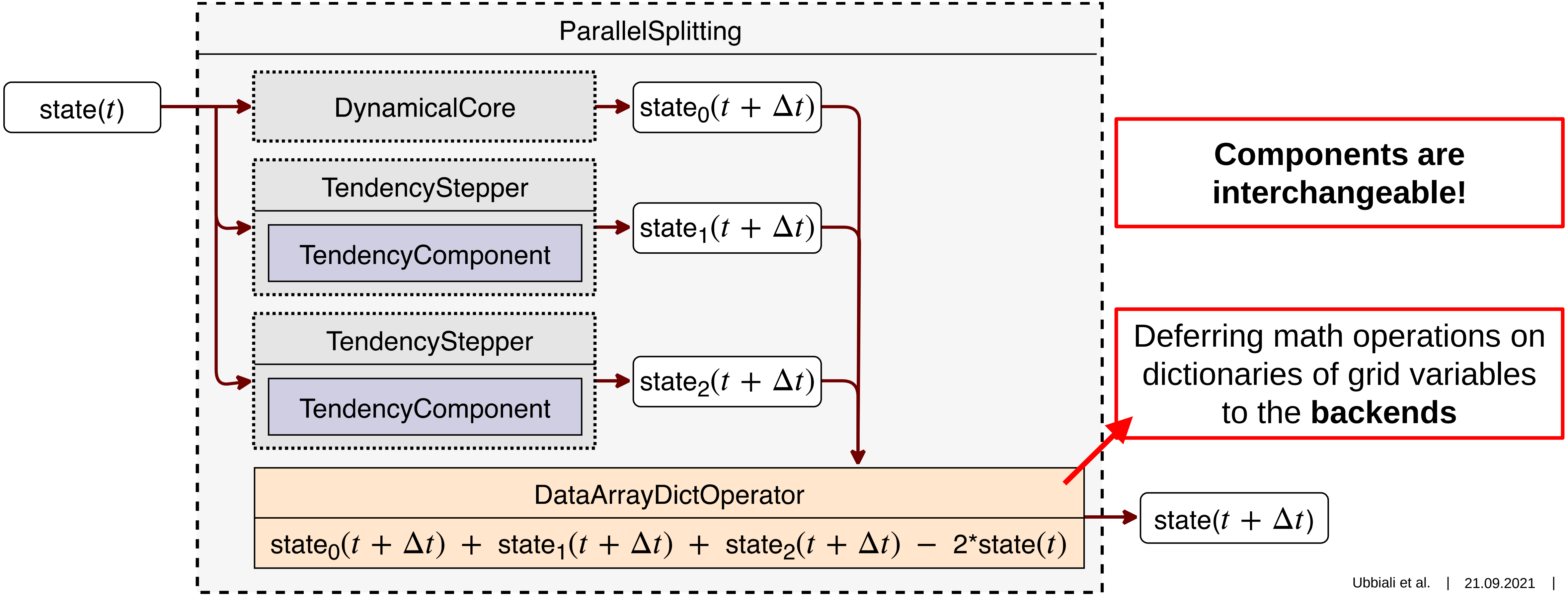




Prill et al. (2020)

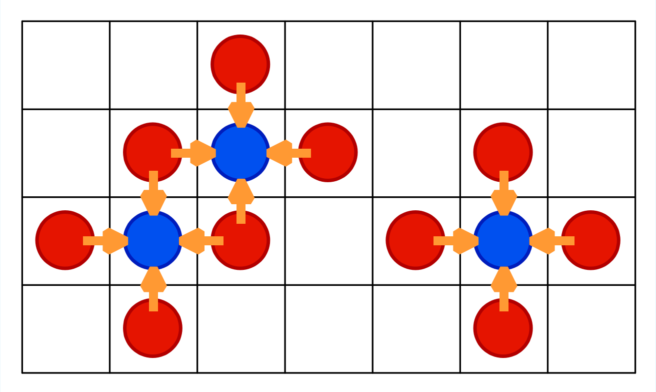
Components: Federation

- The federation classes (or **couplers**) glue dynamics and physics components to form a cohesive model.
- Four couplers currently provided which support six **physics-dynamics coupling** strategies.



Towards performance-portable stencil codes

A **stencil** computes grid point values by accessing a fixed pattern of neighboring points



Ubiquitous in grid-based collocation methods (e.g. FD, FV, FE, single-column physics)

Performance-portability
Running **optimally** on a variety of hardware architectures

Manual
Diversifying implementation based on target hardware

Code versions explosion
Maintenance nightmare

Automatic
Exploit code-generation techniques to synthetize hardware-specific code

Single application code

Tasmania provides a registration mechanism to define, organize and manage **multiple backends**

Infrastructure code ensures proper dispatch of **(i)** memory allocation, **(ii)** kernel compilation, **(iii)** stencil launch

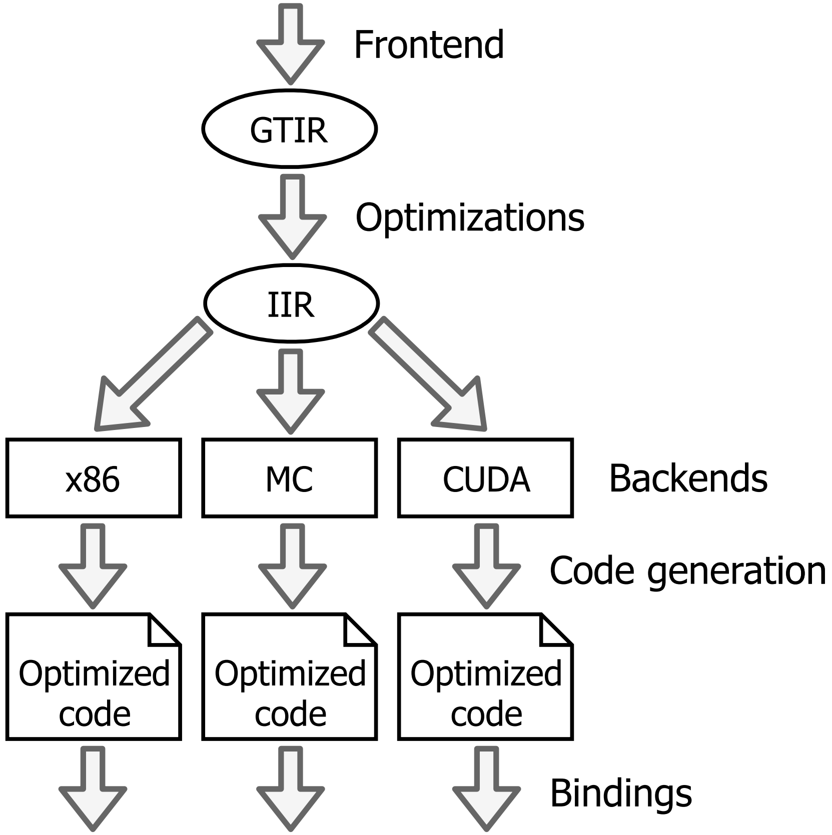
The following backends are available: **(a)** NumPy, **(b)** CuPy, **(c)** **GT4Py**

GT4Py: A library for stencil applications

- Generates high-performance implementations of stencil kernels from **descriptive** and **hardware-agnostic** definition in a Domain Specific Language (**DSL**).
- **Frontend** and **optimization** passes translate definition into hierarchy of tree-like Intermediate Representations (**IR**).
- **Backends** harness the **GridTools** framework to produce optimized code for target computer architectures.
  - Support for multi- and many-core CPU, and GPU.
  - Python backends available for fast prototyping.
- Generated code is compiled into binary and accessible from Python via bindings.

3 P's: Productivity, Portability, Performance

```
@gtscript.stencil(backend="...")
def laplacian(
    in_phi: gtscript.Field[float], out_lap: gtscript.Field[float]
):
    with computation(PARALLEL), interval(...):
        out_lap = - 4 * in_phi[0, 0, 0] \
            + in_phi[-1, 0, 0] + in_phi[1, 0, 0] \
            + in_phi[0, -1, 0] + in_phi[0, 1, 0]
```

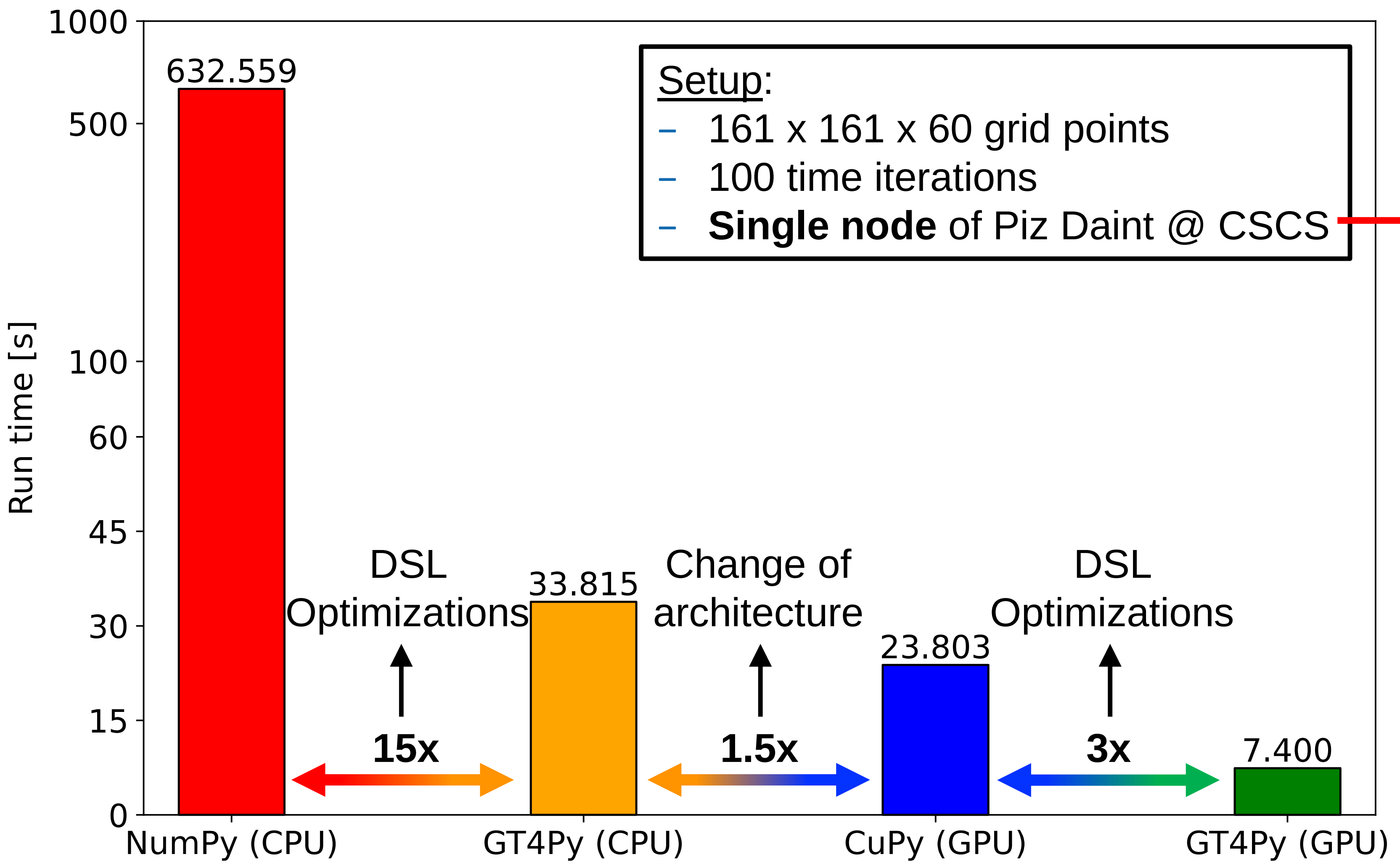



```
laplacian(phi, lap, origin=(1, 1, 0), domain=(nx-2, ny-2, nz))
```

Afanasyev et al. (2021); https://github.com/GridTools/gt4py/

Performance

Proof-of-concepts: Three-dimensional hydrostatic mountain flow model using isentropic coordinates.



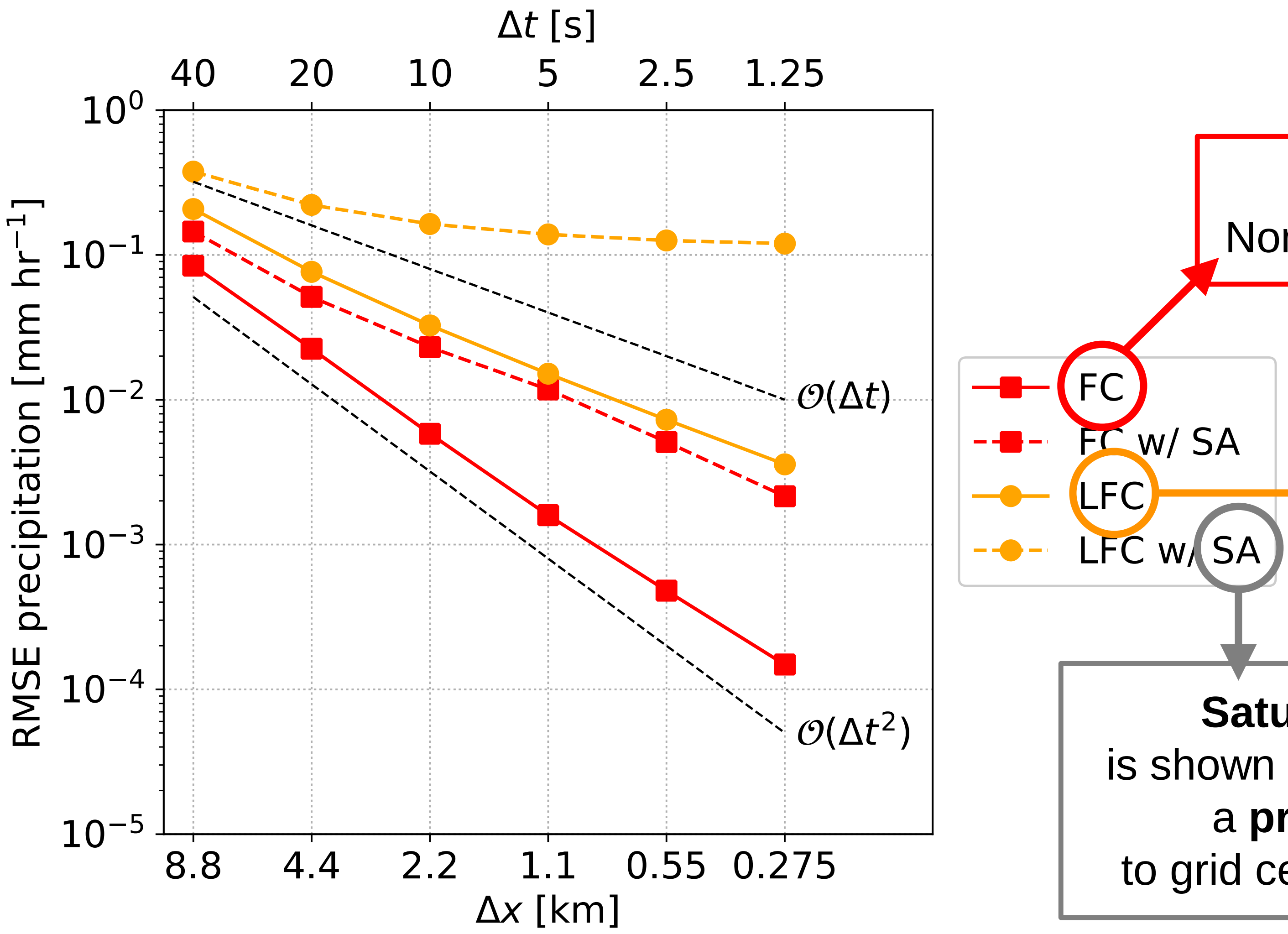
Distributed computing not supported yet

Tasmania + GT4Py
Effective solution to overcome slowness of Python interpreter

Ubbiali et al., submitted

Research application

Self-convergence tests on vertical slice simulations of moist precipitating airflow past isolated mountain.



Full coupling
Nominal rate of convergence: $\mathcal{O}(\Delta t^2)$

***Lazy* full coupling**
Nominal rate of convergence: $\mathcal{O}(\Delta t)$

Saturation adjustment (dotted)
is shown detrimental for convergence w.r.t.
a **prognostic approach (solid)**
to grid cell condensation and evaporation

Conclusions and outlook

- Process coupling seems to play a central role at kilometer-scale resolutions and beyond.
- **Warning!** The full toolchain (**Tasmania** + **GT4Py**) is still under development and so **feature-incomplete**.
- However, early results are promising: Python is a strong candidate for next-generation of models.
- Ongoing efforts to port ICON (EXCLAIM), IFS-FVM (KILOS) and FV3 (O. Fuhrer’s & J. McGibbon’s talks) to Python.

Contact details

Stefano Ubbiali – subbiali@phys.ethz.ch

https://exclaim.ethz.ch/, https://www.pasc-ch.org/projects/2021-2024/kilos/

Main references

Fuhrer, O. (2005). *From advection to convection: Dynamical and numerical issues in high-resolution modeling of atmospheric flows past topography* (Doctoral dissertation). ETH Zurich, Switzerland. Retrieved from https://www.research-collection.ethz.ch/handle/20.500.11850/54203

Malardel, S., Wedi, N., Deconinck, W., Diamantakis, M., Kühnlein, C., Mozdzynski, G., ... & Smolarkiewicz, P. (2016). A new grid for the IFS. *ECMWF Newsletter*, *146*, 23-28.

Monteiro, J. M., McGibbon, J., & Caballero, R. (2018). sympl (v. 0.4. 0) and climt (v. 0.15. 3)–Towards a flexible framework for building model hierarchies in python. *Geoscientific Model Development*, *11*(9), 3781-3794.

Prill, F., Reinert, D., Rieger, D., Zaengl, G., Schroeter, J., Foerstner, J., ... & Vogel, B. (2019). Working with the ICON model practical exercises for NWP mode and ICON-ART. *ICON Model Tutorial, Deutscher Wetterdienst, Offenbach, Germany*.

Schär, C., Fuhrer, O., Arteaga, A., Ban, N., Charpilloz, C., Di Girolamo, S., ... & Wernli, H. (2020). Kilometer-scale climate models: Prospects and challenges. *Bulletin of the American Meteorological Society*, *101*(5), E567-E587.

Ubbiali, S, Schär, C., Schlemmer, L., Schulthess, T. C. (2021). A numerical analysis of six physics-dynamics coupling schemes for atmospheric models. *Journal of Advances in Earth System Modeling*, submitted.