A unified and Python-based approach to the physics-dynamics coupling in atmospheric models

One-year Ph.D. interview

Candidate:
S. Ubbiali

Supervisors:
Prof. Dr. T. Schulthess
Prof. Dr. C. Schär
Dr. L. Schlemmer

December 3, 2018

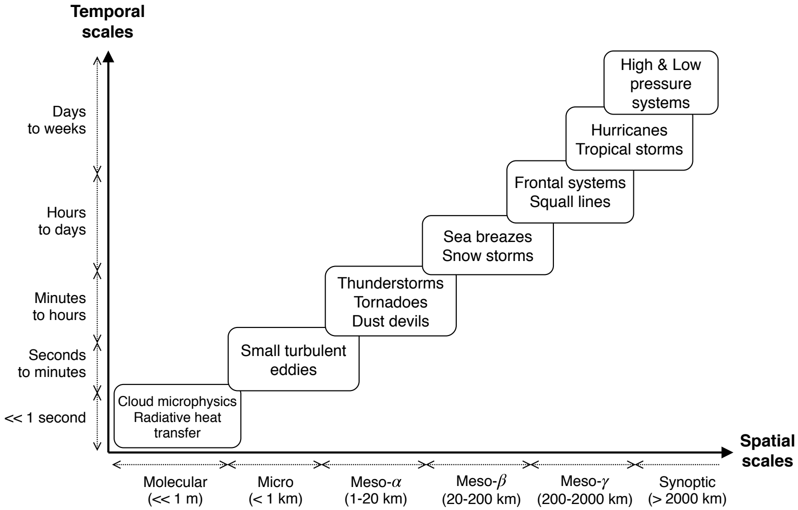
ETH zürich
IAC Institute for Atmospheric and Climate Science
CSCS

Background

Processes occurring in the atmosphere span a wide range of scales.

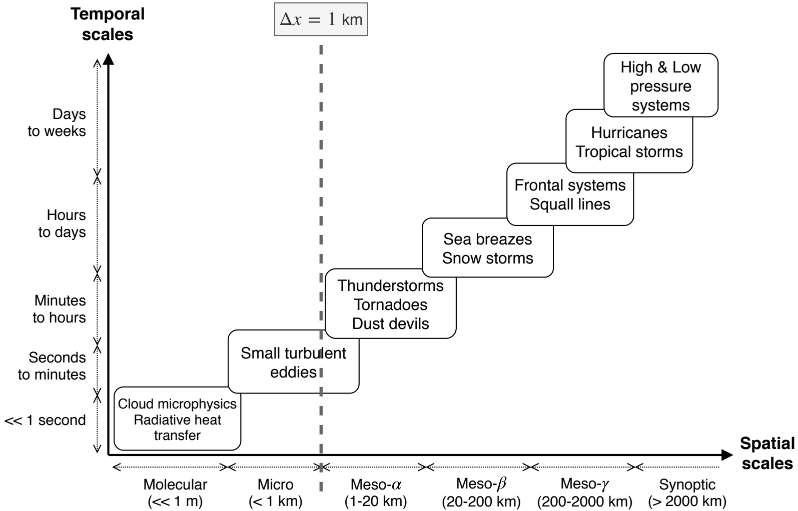
Background

Processes occurring in the atmosphere span a wide range of scales.



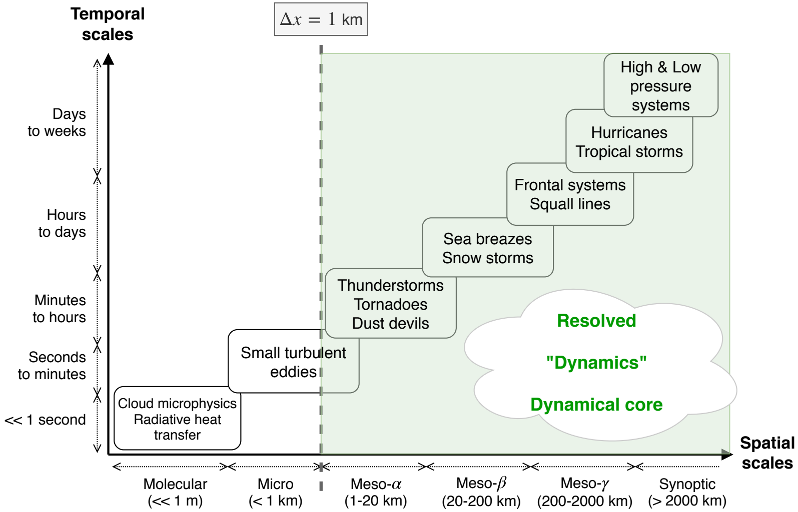
Background

Processes occurring in the atmosphere span a wide range of scales.



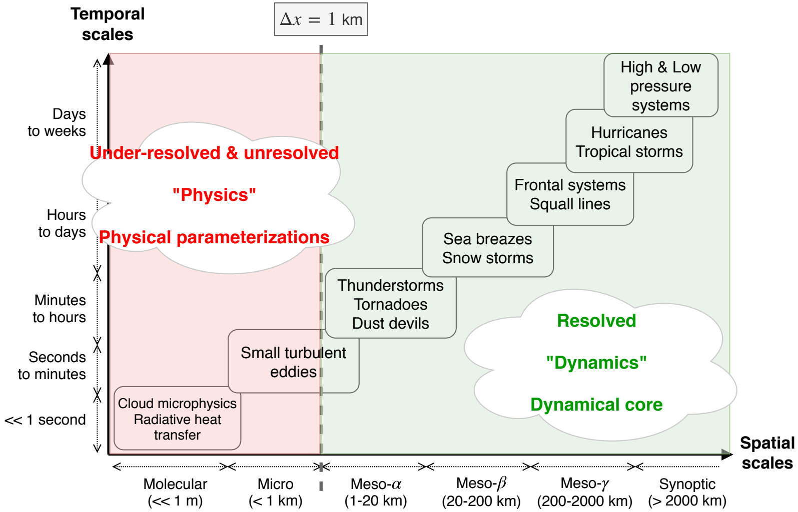
Background

Processes occurring in the atmosphere span a wide range of scales.



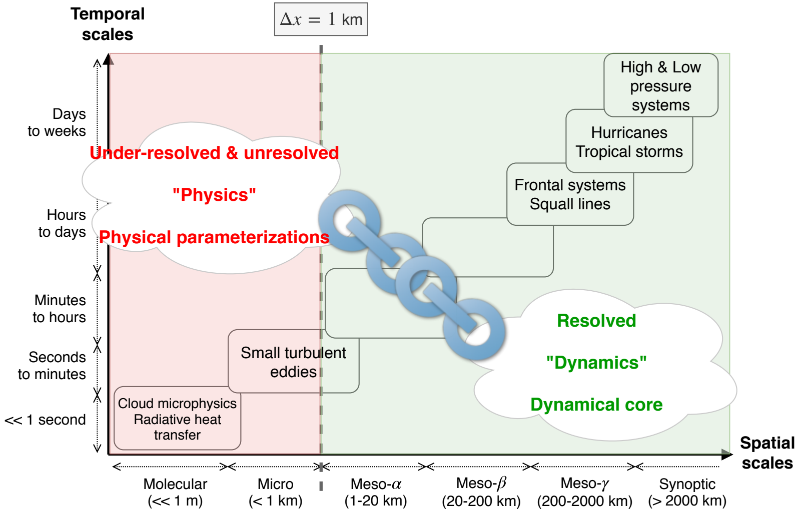
Background

Processes occurring in the atmosphere span a wide range of scales.



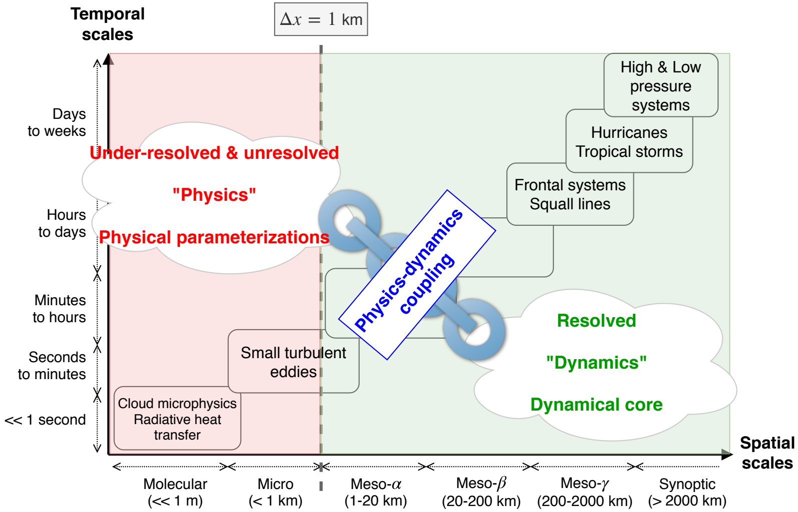
Background

Processes occurring in the atmosphere span a wide range of scales.



Background

Processes occurring in the atmosphere span a wide range of scales.



Motivation

- Physical parameterizations express the mutual interaction between subgrid-scale phenomena and large-scale dynamics in terms of the resolved fields.
- Typically devised in a *single-column* fashion.
- For ease of model development, parameterizations studied in isolation.
  - ✓ Consolidated knowledge of single physical and chemical processes.
  - ✓ Error introduced by any parameterization increasingly reduced.
  - ✗ Poor understanding of the impact of physics time-stepping.
  - ✗ Physics-dynamics coupling performed in a crude fashion.
- Arguably, these deficiencies root in the lack of *interoperability*, *usability* and *software reuse* of legacy (Fortran) codes.

Kalnay (2003), Dickinson et al. (2002), Donahue & Caldwell (2018)

Objectives

At the theoretical level:

- Develop a unified framework to analyze different coupling strategies.
- Extend previous studies carried out in more simplified contexts.
- Determine nominal order of convergence.

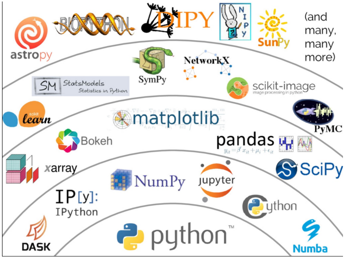
On the software side:

- Build a flexible and modular Python framework (`tasmania`) to ease composition, configuration and simulation of Earth models.

- Each process represented as a Python class.
  - User is given fine-grained control on the execution flow.
  - Wrapping existing Fortran codes is an option. Alternately...
- Inherent execution slowness of the Python interpreter overcome by leveraging `GridTools4Py` (`GT4Py`) - a domain-specific language (DSL) for stencil-based operations.

GridTools4Py: overview

- Developed at CSCS, with `tasmania` serving as major driving application.
- High-level, expressive, declarative definitions of the stencil.
- Internally, `GT4Py` builds an *intermediate representation* (IR) of the stencil as a graph of operations.
  - Allows visual inspection of the operations tree.
- Flexible execution model allowing heterogeneous executors:
  - Pythonic backends for debugging and early testing purposes;
  - Bindings to high-performance `GridTools` backend (in progress).
- Seamless integration with the Python scientific stack enables development of end-to-end applications.
  - Supports standard NumPy arrays.
  - Works in both IPython and Jupyter environments.



E. Paredes

tasmania: Taxonomy of components

```
# Simply retrieve diagnostics
diagnostics = DiagnosticComponent* (state)

# Calculate tendencies, i.e., time-derivatives,
# and retrieve diagnostics
tendencies, diagnostics = TendencyComponent* (state)

# Step the state based on one or more TendencyComponents,
new_state = TendencyStepper* (state, timestep)

# Dynamical core
new_state = DynamicalCore(timestep, state, tendencies)
```

* Abstract base class provided by `sympl` – a package offering functions and objects which could be used by any Earth system model.

Monteiro et al. (2018)

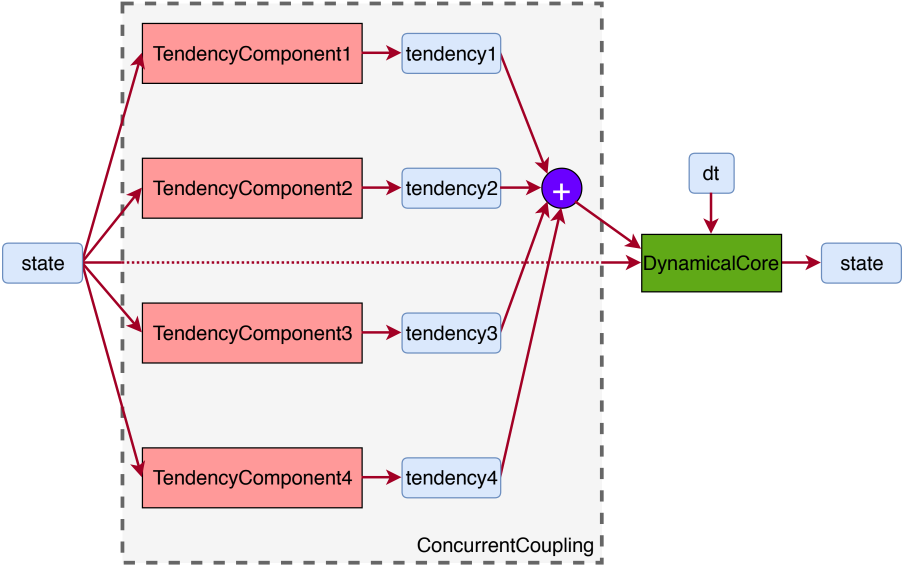
`tasmania`: Couplers

- Couplers automate the execution of a set of physical packages, pursuing a well-defined coupling strategy.
- Three couplers are currently available:
  - `ConcurrentCoupling`,
  - `ParallelSplitting` (not working...),
  - `SequentialUpdateSplitting`,

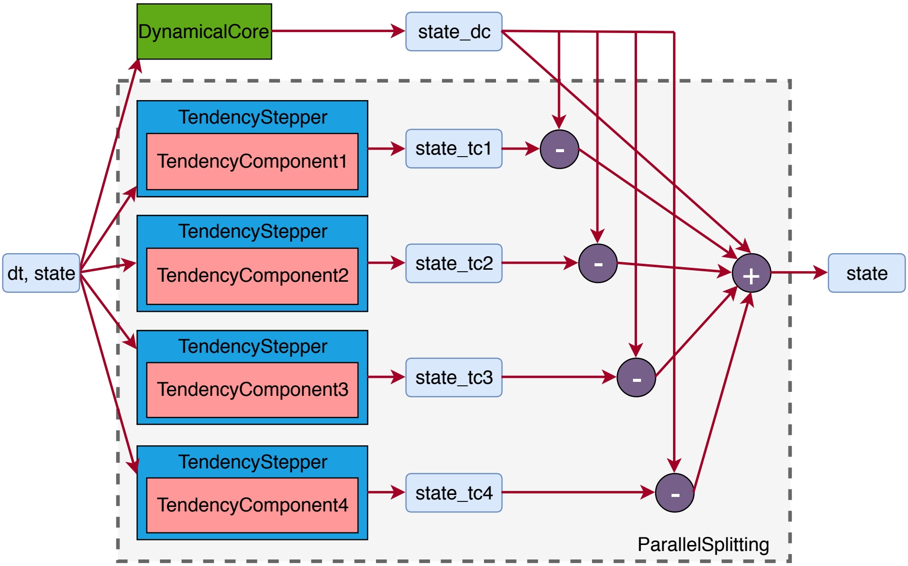
  which support four coupling mechanisms:
  - concurrent coupling (CC) [high order],
  - parallel splitting (PS) [1^{st}-order],
  - sequential-update splitting (SUS) [1^{st}-order],
  - symmetrized sequential-update splitting (SSUS) [2^{nd}-order].
- Couplers assert compatibility between encompassed components.

Staniforth (2002a,b), Dubal et al. (2004, 2005, 2006), Strang (1968)

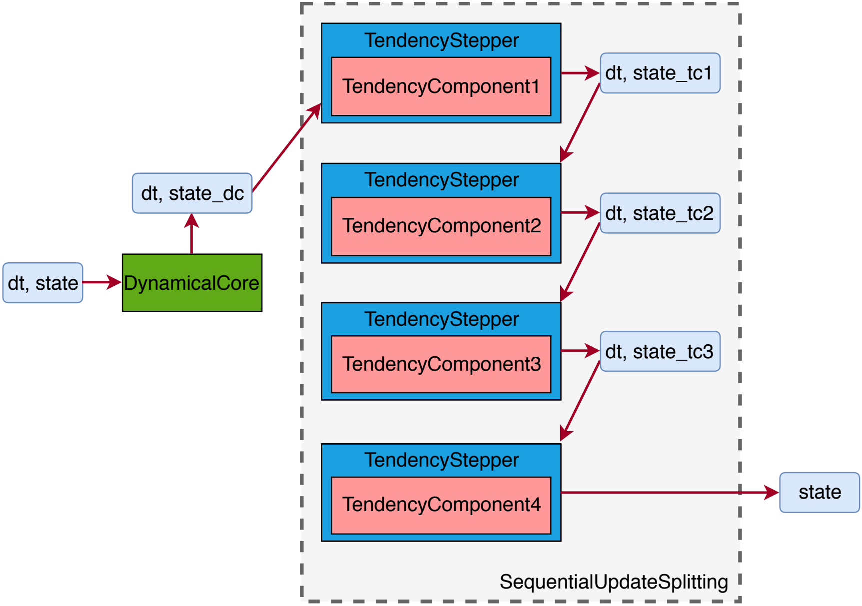
Concurrent coupling (CC)



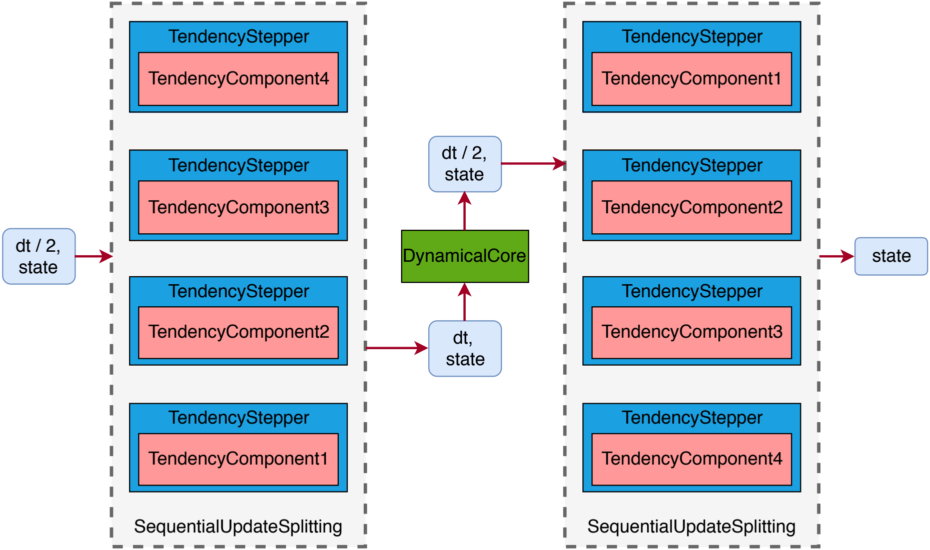
Parallel splitting (PS)



Sequential-update splitting (SUS)



Symmetrized sequential-update splitting (SSUS)



The isentropic system

Isentropic coordinates consist of:

- horizontal, Cartesian coordinates x and y;
- potential temperature θ,

$$\theta := T \left(\frac{p}{p_{\text{ref}}} \right)^{R/c_p} \in [\theta_s, \theta_t] ,$$

with

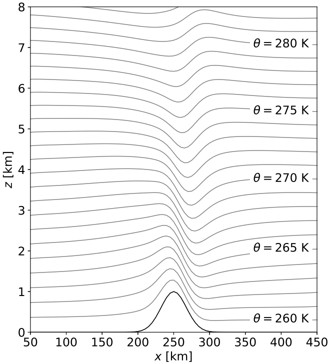
$T =$ temperature,

$p =$ pressure,

$p_{\text{ref}} = 1000$ hPa,

$c_p =$ specific heat for dry at constant pressure,

$R =$ gas constant for dry air.



Isentropic surfaces in a stable atmosphere.

Isentropic model: Hydrostatic dynamics in isentropic coordinates, where the diabatic heat acts as the vertical velocity.

Zeman (2016)

The isentropic model: Prognostic equations

$\sigma := -(1/g)\,\partial p/\partial\theta$ = *isentropic density*, with g = gravitational constant

$\mathbf{v} = [u,\, v]^T$ = horizontal velocity vector

$U := \sigma\, u$ = x-momentum, $V := \sigma\, v$ = y-momentum

$M := c_p\, T + g\, z$ = Montgomery potential, with z = geometric height

f = Coriolis parameter

q = non-precipitating tracer, S_q = physical source-sink rates for q

$$\frac{\partial \sigma}{\partial t} + \frac{\partial u\sigma}{\partial x} + \frac{\partial v\sigma}{\partial y} = -\frac{\partial \dot{\theta}\sigma}{\partial \theta}$$

$$\frac{\partial U}{\partial t} + \frac{\partial uU}{\partial x} + \frac{\partial vU}{\partial y} = -\frac{\partial \dot{\theta}U}{\partial \theta} - \sigma\frac{\partial M}{\partial x} + fV$$

$$\frac{\partial V}{\partial t} + \frac{\partial uV}{\partial x} + \frac{\partial vV}{\partial y} = -\frac{\partial \dot{\theta}V}{\partial \theta} - \sigma\frac{\partial M}{\partial y} - fU$$

$$\frac{\partial \sigma q}{\partial t} + \frac{\partial u\sigma q}{\partial x} + \frac{\partial v\sigma q}{\partial y} = -\frac{\partial \dot{\theta}\sigma q}{\partial \theta} + \sigma S_q$$

The isentropic model: Prognostic equations

$\sigma := -(1/g)\,\partial p/\partial\theta$ = *isentropic density*, with g = gravitational constant

$\mathbf{v} = [u,\, v]^T$ = horizontal velocity vector

$U := \sigma\, u$ = x-momentum, $V := \sigma\, v$ = y-momentum

$M := c_p\, T + g\, z$ = Montgomery potential, with z = geometric height

f = Coriolis parameter

q = non-precipitating tracer, S_q = physical source-sink rates for q

---

$$\frac{\partial \sigma}{\partial t} + \boxed{\frac{\partial u\sigma}{\partial x} + \frac{\partial v\sigma}{\partial y}} = 0$$

$$\frac{\partial U}{\partial t} + \boxed{\frac{\partial uU}{\partial x} + \frac{\partial vU}{\partial y}} = \boxed{-\,\sigma\frac{\partial M}{\partial x}}$$

$$\frac{\partial V}{\partial t} + \boxed{\frac{\partial uV}{\partial x} + \frac{\partial vV}{\partial y}} = \boxed{-\,\sigma\frac{\partial M}{\partial y}}$$

Dry adiabatic configuration:

- $\mathscr{D}$
- $\mathscr{P}_1$

The isentropic model: Prognostic equations

$\sigma := -(1/g)\,\partial p/\partial\theta$ = *isentropic density*, with g = gravitational constant

$\mathbf{v} = [u,\, v]^T$ = horizontal velocity vector

$U := \sigma\, u$ = x-momentum, $V := \sigma\, v$ = y-momentum

$M := c_p\, T + g\, z$ = Montgomery potential, with z = geometric height

f = Coriolis parameter

q = non-precipitating tracer, S_q = physical source-sink rates for q

---

$$\frac{\partial\sigma}{\partial t} + \boxed{\frac{\partial u\sigma}{\partial x} + \frac{\partial v\sigma}{\partial y}} = \boxed{-\frac{\partial\dot{\theta}\sigma}{\partial\theta}}$$

$$\frac{\partial U}{\partial t} + \boxed{\frac{\partial uU}{\partial x} + \frac{\partial vU}{\partial y}} = \boxed{-\frac{\partial\dot{\theta}U}{\partial\theta}}\ \boxed{-\,\sigma\frac{\partial M}{\partial x}}\ \boxed{+\,fV}$$

$$\frac{\partial V}{\partial t} + \boxed{\frac{\partial uV}{\partial x} + \frac{\partial vV}{\partial y}} = \boxed{-\frac{\partial\dot{\theta}V}{\partial\theta}}\ \boxed{-\,\sigma\frac{\partial M}{\partial y}}\ \boxed{-fU}$$

Dry non-adiabatic configuration:

- $\mathscr{D}$
- $\mathscr{P}_1$
- $\mathscr{P}_2$
- $\mathscr{P}_3$

The isentropic model: Prognostic equations

$\sigma := -(1/g)\,\partial p/\partial\theta$ = *isentropic density*, with g = gravitational constant

$\mathbf{v} = [u, v]^T$ = horizontal velocity vector

$U := \sigma\, u$ = x-momentum, $V := \sigma\, v$ = y-momentum

$M := c_p\, T + g\, z$ = Montgomery potential, with z = geometric height

f = Coriolis parameter

q = non-precipitating tracer, S_q = physical source-sink rates for q

$$\frac{\partial\sigma}{\partial t} + \boxed{\frac{\partial u\sigma}{\partial x} + \frac{\partial v\sigma}{\partial y}} = \boxed{-\frac{\partial\dot{\theta}\sigma}{\partial\theta}}$$

$$\frac{\partial U}{\partial t} + \boxed{\frac{\partial uU}{\partial x} + \frac{\partial vU}{\partial y}} = \boxed{-\frac{\partial\dot{\theta}U}{\partial\theta}}\ \boxed{-\,\sigma\frac{\partial M}{\partial x}}\ \boxed{+\,fV}$$

$$\frac{\partial V}{\partial t} + \boxed{\frac{\partial uV}{\partial x} + \frac{\partial vV}{\partial y}} = \boxed{-\frac{\partial\dot{\theta}V}{\partial\theta}}\ \boxed{-\,\sigma\frac{\partial M}{\partial y}}\ \boxed{-fU}$$

$$\frac{\partial\sigma q}{\partial t} + \boxed{\frac{\partial u\sigma q}{\partial x} + \frac{\partial v\sigma q}{\partial y}} = \boxed{-\frac{\partial\dot{\theta}\sigma q}{\partial\theta}}\ \boxed{+\,\sigma S_q}$$

Moist non-adiabatic configuration*:

- $\mathscr{D}$
- $\mathscr{P}_1$
- $\mathscr{P}_2$
- $\mathscr{P}_3$
- $\mathscr{P}_4$

*Not tested

The isentropic model: Diagnostic equations

$\Pi := c_p \left(\frac{p}{p_{\text{ref}}} \right)^{R/c_p} = \frac{c_p \, T}{\theta}$ = Exner function

z_s = topography height

$$\begin{cases} \dfrac{\partial p}{\partial \theta} = -g \, \sigma \\ p(\theta = \theta_t) = p_t \end{cases} \quad \text{and} \quad \begin{cases} \dfrac{\partial M}{\partial \theta} = \Pi \\ M(\theta = \theta_s) = \Pi(\theta = \theta_s) \, \theta_s + g \, z_s \end{cases}$$

Adiabatic flow: Convergence analysis (1)

Two-dimensional, dry flow over an isolated *Witch of Agnesi* mountain,

$$z_s(x) = \frac{h\,a^2}{x^2 + a^2}\,,$$

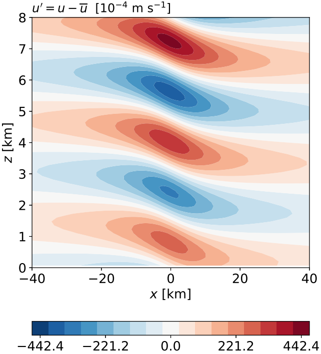
with $h = 1$ m the mountain height and $a = 10$ km its half-width at half-height.

If the flow is isothermal and the mean wind $\overline{u}$ is constant with height, the displacement of a streamline from its height far upstream satisfies

$$\delta(x, z) = \left(\frac{\overline{\rho}}{\rho_0}\right)^{-1/2} h\,a\,\frac{a\cos(l\,z) - x\sin(l\,z)}{x^2 + a^2}\,,$$

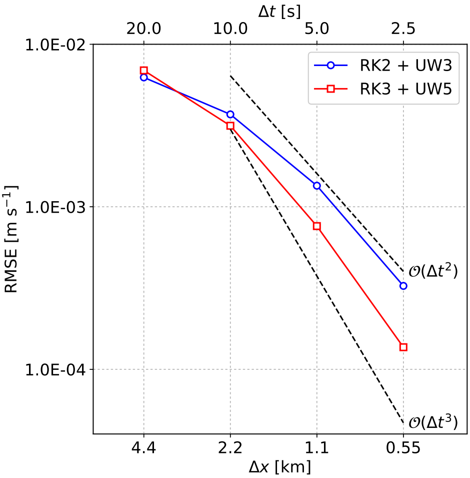
where $\overline{\rho}$ and ρ_0 are the mean and base state density, respectively, and l is the Scorer parameter. Then, the velocity components can be derived as

$$u = \overline{u}\left[1 - \frac{1}{\overline{\rho}}\left(\overline{\rho}\,\delta\right)_z\right] \quad \text{and} \quad w = \overline{u}\,\delta_x\,.$$



Langhans et al. (2012)

Adiabatic flow: Convergence analysis (2)



- Blue: 2-stages Runge-Kutta (RK2) + 3rd-order upwind scheme (UW3) + 2nd-order centered formula for pressure gradient.
- Red: 3-stages Runge-Kutta (RK3) + 5th-order upwind scheme (UW5) + 4th-order centered formula for pressure gradient.
- Root-mean-squared error (RMSE) w.r.t. a high-resolution simulation at $\Delta x = 275$ m.
- Only grid points within 40 km from mountain center and below 8 km considered.
- Similar results with any coupling.

Non-adiabatic flow: Description

- Consider stratified flows past isolated mountains with heated/cooled surface.
- The *modified* thermal forcing is given by

$$\dot{\theta} = \begin{cases} \dfrac{\theta\, R_d\, \alpha}{p\, c_p}\, F_0 \exp\left[-\alpha(z - z_s)\right]\left[\sin(2\pi\omega\, t) + \sin(2\pi\omega_{\mathtt{FW}}\, t)\right] & \text{if } r < L^*, \\ 0 & \text{otherwise}, \end{cases}$$

where

$$F_0 = F_0(t) = \begin{cases} 800 \text{ W m}^{-2} & \text{at day} \\ -75 \text{ W m}^{-2} & \text{at night} \end{cases}, \quad \alpha = \alpha(t) = \begin{cases} 1/600 \text{ m}^{-1} & \text{at day} \\ 1/75 \text{ m}^{-1} & \text{at night} \end{cases},$$

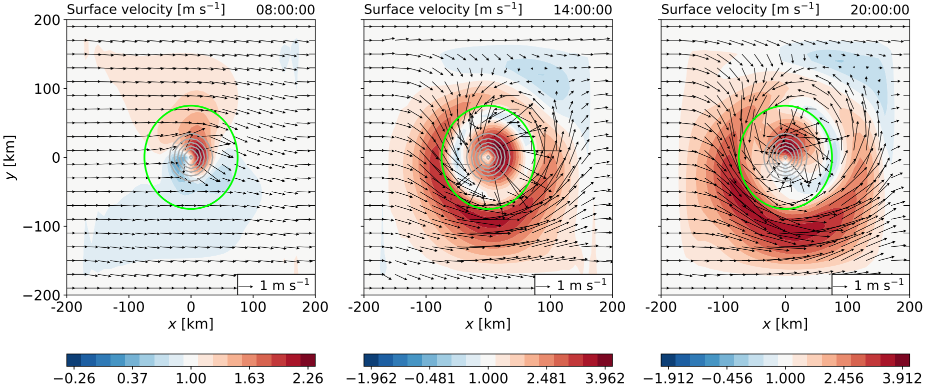
$$z_s = H[1 + (r/L)^2]^{-3/2}, \quad r = \sqrt{x^2 + y^2}, \quad L^* = 3L,$$

$$\omega = 1/24 \text{ h}^{-1} \text{ (diurnal cycle)}, \quad \omega_{\mathtt{FW}} \in \{0, 1/2, 2\} \text{ h}^{-1} \text{ (high-frequency forcing)}.$$

- Configuration: $H = 500$ m, $L = 25$ km, $u(t=0) = 1$ m s^{-1}, $v(t=0) = 0$ m s^{-1}.
- Numerical solver: RK3 + UW5 (horizontal adv.) + UW3 (vertical adv.)
- Resolution: $\Delta x = \Delta y = 10$ km, 50 vertical levels, $\Delta t = 10$ s.

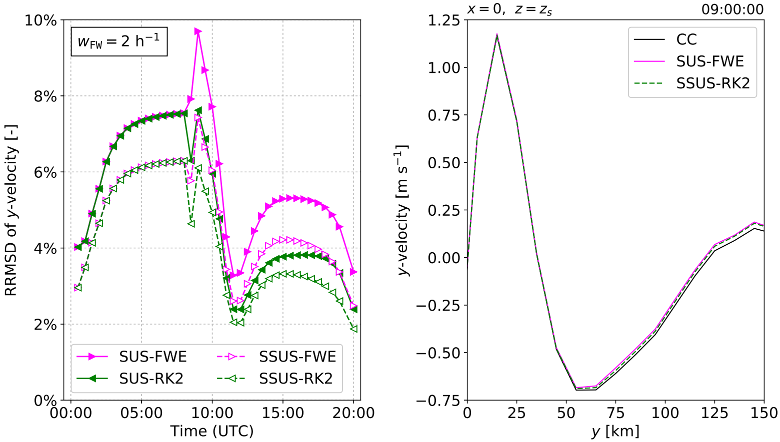
Reisner & Smolarkiewicz (1994)

Non-adiabatic flow: Early results (1)



Near-surface velocity field produced by CC for $\omega_{\mathtt{FW}} = 0$ at 0800 UTC (*left*), 1400 UTC (*center*) and 2000 UTC (*right*). The simulation starts at 0000 UTC and the heating source is switched off until 0800 UTC; the simulation is then conducted for further 12 hours. The green circle encloses the heated/cooled area.

Non-adiabatic flow: Early results (2)



Left: Time series of relative root-mean-squared deviation (RRMSD) of y-velocity yielded by SUS and SSUS, calculated with respect to the CC solution for $\omega_{\mathtt{FW}} = 2$ h^{-1}. Physics time stepping performed via forward Euler (FWE) or two-stages Runge-Kutta (RK2). *Right*: Horizontal profile of y-velocity at 0900 UTC.

Tentative schedule

Tasks	**Time**
Start of the Ph.D. project.	November 1, 2017
Familiarize with the weather and climate research fields.	Winter 2018/2019
Develop `tasmania` leveraging `GT4Py-v3`; implement different solvers for the isentropic model, a few parameterizations and various numerical dwarfs within the framework.	January 2018 - November 2018
Shape the theoretical framework and analyze state-of-the-art coupling mechanisms.	Summer 2018
Upgrade `tasmania` to `GT4Py-v4`; report feedbacks to the `GT4Py` development team.	November 2018 - February 2019
Run extensive convergence tests with the isentropic model on CSCS machines.	Spring 2019
Implement a non-hydrostatic model and further parameterizations; perform experiments.	Winter-Summer 2019
Investigate alternative ways to couple physics with dynamics.	Fall 2019 - Winter 2020
Integrate `tasmania` with a domain decomposition library.	Winter-Summer 2020
Writing of the thesis.	Summer-Fall 2020

References I

Caya, A., Laprise, R., and Zwack, P. (1998). Consequences of using the splitting method for implementing physical forcings in a semi-implicit semi-Lagrangian model. *Mon. Wea. Rev.*, 126(6):1707–1713.

Dickinson, R. E., Zebiak, S. E., Anderson, J. L., Blackmon, M. L., De Luca, C., Hogan, T. F., Iredell, M., Ji, M., Rood, R. B., Suarez, M. J., et al. (2002). How can we advance our weather and climate models as a community?. *Bull. Amer. Meteor. Soc.*, 83(3):431–436.

Donahue, A. S. and Caldwell, P. M. (2018). Impact of physics parameterization ordering in a global atmosphere model. *Journal of Advances in Modeling Earth Systems*, 10(2):481–499.

Dubal, M., Wood, N., and Staniforth, A. (2004). Analysis of parallel versus sequential splittings for time-stepping physical parameterizations. *Mon. Wea. Rev.*, 132(1):121–132.

Dubal, M., Wood, N., and Staniforth, A. (2005). Mixed parallel–sequential-split schemes for time-stepping multiple physical parameterizations. *Mon. Wea. Rev.*, 133(4):989–1002.

References II

Dubal, M., Wood, N., and Staniforth, A. (2006). Some numerical properties of approaches to physics–dynamics coupling for NWP. *Quart. J. Royal Meteorol. Soc.*, 132(614):27–42.

Durran, D. R. (2013). *Numerical methods for wave equations in geophysical fluid dynamics*, volume 32. Springer Science & Business Media.

Gross, M., Wan, H., Rasch, P. J., Caldwell, P. M., Williamson, D. L., Klocke, D., Jablonowski, C., Thatcher, D. R., Wood, N., Cullen, M., et al. (2018). Physics–dynamics coupling in weather, climate, and earth system models: Challenges and recent progress. *Mon. Wea. Rev.*, 146(11):3505–3544.

Kalnay, E. (2003). *Atmospheric modeling, data assimilation and predictability*. Cambridge university press.

Langhans, W., Schmidli, J., and Schär, C. (2012). Bulk convergence of cloud-resolving simulations of moist convection over complex terrain. *J. Atmos. Sci.*, 69(7):2207–2228.

Monteiro, J. M., McGibbon, J., and Caballero, R. (2018). `sympl` (v. 0.3. 2) and `climt` (v. 0.11. 0) – towards a flexible framework for building model hierarchies in Python. *Geosci. Model Dev. Disc.* In review.

References III

Reisner, J. M. and Smolarkiewicz, P. K. (1994). Thermally forced low Froude number flow past three-dimensional obstacles. *J. Atmos. Sci.*, 51(1):117–133.

Schulthess, T. C., Bauer, P., Fuhrer, O., Hoefler, T., Schär, C., and Wedi, N. (2019). Reflecting on the goal and baseline for exascale computing: a roadmap based on weather and climate simulation. *IEEE Computing in Science and Engineering*. Accepted.

Staniforth, A., Wood, N., and Côté, J. (2002a). Analysis of the numerics of physics–dynamics coupling. *Quart. J. Royal Meteorol. Soc.*, 128(586):2779–2799.

Staniforth, A., Wood, N., and Côté, J. (2002b). A simple comparison of four physics–dynamics coupling schemes. *Mon. Wea. Rev.*, 130(12):3129–3135.

Strang, G. (1968). On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5(3):506–517.

Williamson, D. L. (2002). Time-split versus process-split coupling of parameterizations and dynamical core. *Mon. Wea. Rev.*, 130(8):2024–2041.

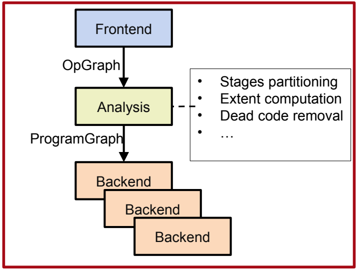
Zeman, C. (2016). *An isentropic mountain flow model with iterative synchronous flux correction*. Master thesis, ETH Zürich.

Why Python?

- New generations of domain scientists are more familiar and proficient with high-level programming languages, e.g., Python, than low-level languages, e.g., Fortran, C/C++.
- Python for scientific computing:
  - Python language has clean syntax and great expressiveness;
  - Python integration with other languages is great;
  - Python scientific stack includes an impressive collection of open source software for scientific computing (SciPy ecosystem).
- Main reason behind this success is Python's Buffer Protocol.
  - Low-level API for direct manipulation of memory buffers.
  - Multi-dimensional data structures stored in contiguous memory.
  - It eliminates the need to copy data.

E. Paredes

- The **frontend** builds a intermediate representation of the stencil as a graph of operations.
- During the analysis, the graph is processed and transformed to collect backend-independent information.
  - E.g., stages, temporaries, field accesses.
- The **backend** translates IR in specific code for target hardware.
- Modular design.
  - Ease addition of new components.



E. Paredes, Schulthess et al. (2019)

GridTools4Py: An example

```python
import gridtools as gt

@gt.stencil_function(backend="numpy")
def horizontal_diffusion(data, weight, *, alpha=4.0)):
    laplacian = - alpha * data[0, 0] + \
                (data[-1, 0] + data[1, 0] +
                 data[0, -1] + data[0, 1])
    flux_i = laplacian[1, 0] - laplacian[0, 0]
    flux_j = laplacian[0, 1] - laplacian[0, 0]
    diffusion = weight[0, 0] * (flux_i[-1, 0] - flux_i[0, 0] +
                                 flux_j[0, -1] - flux_j[0, 0])
    return diffusion
```

```python
import numpy as np

in_data  = np.random.rand(64, 64)
weight   = np.random.rand(64, 64)

out_data = horizontal_diffusion(in_data, weight)
```

GridTools4Py: Sketchy roadmap

- First GPL-licensed release of `GT4Py` by Q1 2019.
  - Directly directed towards academics.
  - Life span: $\approx$ 2 years.
  - Thinking about injecting `GT4Py` is some MSc courses to get students involved.
- `GT` backend available by Q2 2019.
  - Still under GPL license.
- Going multinode: integration of `GT4Py` with a domain partition library (cf. L. Strebel's MSc thesis).

`tasmania`: Design principles & requirements

(i) Model conceived as a chain of components, with each component representing a physical or dynamical process.

(ii) User is given fine-grained control on which components to include in the model, and in which order they should be executed.

(iii) Components are highly inter-operable.
   - In principle, any order of execution is supported.

(iv) Components can be instantiated independently, and run stand-alone.
   - E.g., a SCM does not need an underlying dycore to be executed.
   - Particularly useful for early testing of parameterization schemes.

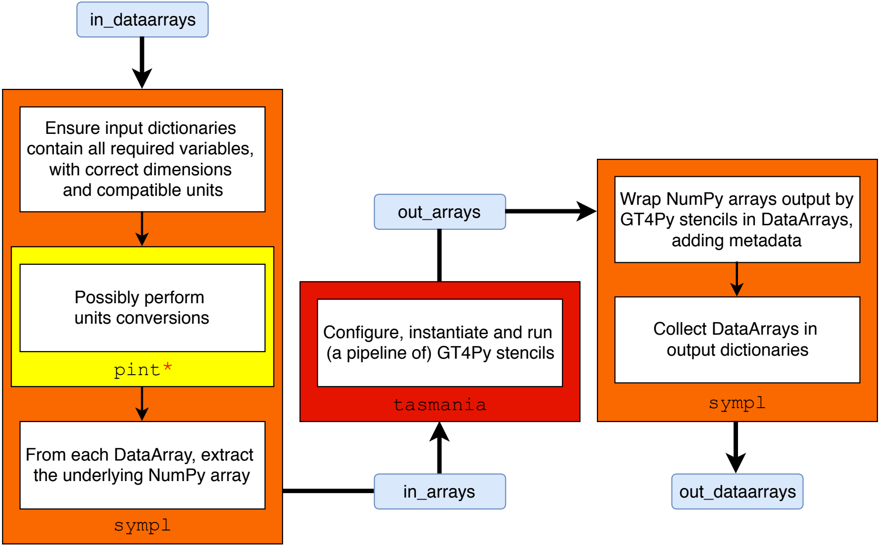
(v) Support for 2D, 3D, and single-column simulations.

(vi) User-interface abstracts away details of the computing architecture.
   - Separation of concerns between domain and computer scientists.

tasmania: Some (technical) details

- state, tendencies and diagnostics are plain dictionary where:
  - *keys* are strings denoting quantity names, which should be compliant with the CF Conventions;
  - *values* are xarray's DataArrays storing values and metadata (dimensions, coordinates, units) for those quantities.
- DataArray's API similar to that for pandas' Series.
- Each component specifies fundamental properties (name, dimensions, units) of both the variables which it requires in input, and the quantities which it calculates and outputs.
- This allows for effective runtime checks on input arguments.

tasmania: Intra-component workflow



* Package to define, operate and manipulate physical quantities.

Unified theoretical framework (1)

Canonical problem

$\psi = \psi(\boldsymbol{x}, t)$ = prognostic variable

$\mathscr{D}$ = spatial operator modeling the dynamics

$\mathscr{P}_m$ = $m-$th physical process, with $\mathscr{P} = \sum_{m=1}^{M} \mathscr{P}_m$

$R = R(\boldsymbol{x}, t)$ = forcing term independent of ψ (e.g., orographic forcing)

$$\frac{\partial \psi}{\partial t} - \mathscr{D}\psi = \sum_{m=1}^{M} \mathscr{P}_m \psi + R(\boldsymbol{x}, t) = \mathscr{P}\psi + R(\boldsymbol{x}, t)$$

Exact solution

If $\mathscr{D}$ and $\mathscr{P}$ do not depend on time, and $R \equiv 0$:

$$\begin{aligned} \psi(t) &= \psi(0) \exp\left[t\big(\mathscr{D} + \mathscr{P}\big)\right] \\ &= \psi(0) \left[I + t\,(\mathscr{D} + \mathscr{P}) + (1/2)\, t^2\, (\mathscr{D} + \mathscr{P})^2 + (1/6)\, t^3\, (\mathscr{D} + \mathscr{P})^3 + \ldots \right] \end{aligned}$$

$$\Rightarrow \psi(t+\Delta t) = \underbrace{\left[I + \Delta t\,(\mathscr{D} + \mathscr{P}) + (1/2)\Delta t^2\, (\mathscr{D} + \mathscr{P})^2 + (1/6)\Delta t^3\, (\mathscr{D} + \mathscr{P})^3 + \ldots \right]}_{\textbf{Exact update operator}} \psi(t)$$

Unified theoretical framework (2)

Semi-discretized form

$\psi_j = \psi_j(t) \approx \psi(\boldsymbol{x}_j, t)$

$\mathbb{D}$, $\mathbb{P}_m$ = numerical approximations of $\mathscr{D}$ and $\mathscr{P}_m$, respectively

$R_j(t) = R(\boldsymbol{x}_j, t)$

$$\frac{d\psi_j}{dt} - \mathbb{D}\psi_j = \mathbb{P}\psi_j + R_j(t) \qquad \text{(Set of ODEs)}$$

Note. In what follows, subscripts denoting the spatial location are omitted.

Fully-discretized version

- For the sake of tractability, we limit ourselves to two-time-level multi-step time integration schemes.
- The discretization of an ODE $\dot{\psi} = f(t,\ \psi)$ can be cast into the form

  $$\psi^{n+1} = \mathbf{L}\psi^n,$$

  with $\mathbf{L} = \mathbf{L}(t,\ \Delta t,\ f)$ the (nonlinear) numerical update operator.

Unified theoretical framework: Coupling

SUS

(i) $\psi^{n+1,0} = \mathbf{L}_D(\Delta t,\ \mathbb{D})\,\psi^n$

(ii) $\psi^{n+1,m} = \mathbf{L}_m(\Delta t,\ \mathbb{P}_m)\,\psi^{n+1,m-1}$,
for $m = 1,\ldots,M$

(iii) $\psi^{n+1} = \mathbf{L}_R(\Delta t,\ R)\,\psi^{n+1,M}$

PS

(i) $\psi^{n+1,D} = \mathbf{L}_D(\Delta t,\ \mathbb{D})\psi^n$

(ii) $\psi^{n+1,m} = \mathbf{L}_m(\Delta t,\ \mathbb{P}_m)\psi^n$,
for $m = 1,\ldots,M$

(iii) $\psi^{n+1,R} = \mathbf{L}_R(\Delta t,\ R)\psi^n$

(iv) $$\frac{\psi^{n+1}-\psi^n}{\Delta t} = \frac{\psi^{n+1,D}-\psi^n}{\Delta t} + \sum_{m=1}^{M}\frac{\psi^{n+1,m}-\psi^n}{\Delta t} + \frac{\psi^{n+1,R}-\psi^n}{\Delta t}$$

CC

$\psi^{n+1} = \mathbf{L}(\Delta t,\ \mathbb{D}+\mathbb{P}+R)\ \psi^n$

SSUS

$0 < \eta < 1$

(i) $\widetilde{\psi}^0 = \mathbf{L}_R(\eta\Delta t,\ R)\,\psi^n$

(ii) $\widetilde{\psi}^m = \mathbf{L}_m(\eta\Delta t,\ \mathbb{P}_m)\,\widetilde{\psi}^{m-1}$,
for $m = 1,\ldots,M$

(iii) $\psi^{n+1,M+1} = \mathbf{L}_D(\Delta t,\ \mathbb{D})\,\widetilde{\psi}^M$

(iv) $\psi^{n+1,m} = \mathbf{L}_m((1-\eta)\Delta t,\ \mathbb{P}_m)\,\psi^{n+1,m+1}$,
for $m = M,\ldots,1$

(iv) $\psi^{n+1} = \mathbf{L}_R((1-\eta)\Delta t,\ R)\,\psi^{n+1,1}$