UK-based use of SCM and LES cases

Leif Denby, University of Leeds

HIGHTUNE/DEPHY workshop on SCM & LES case standardisation

Monday 15th June 2020

Two models

Met Office UM (Unified Model)

- Operational NWP and GCM
- Will be replaced by LFRic

SCM used for:

- first stage of testing in the development of new schemes (or adjustments to existing ones)
- debugging physics implementations
- detecting bugs through (automated) testing

Fast build-run-debug cycle makes fixing errors easy

- Input: namelist case definitions (initial and forcing profiles), HIGHTUNE/DEPHY reading unlikely in UM (LFRic possible)

MONC (Met Office-NERC Cloud model)

- Large-Eddy Simulations Model

LES used for:

- Process studies
- Constraining process schemes in GCM/NWP

Detailed process studies not possible in UM

- Input: namelist case definitions (initial and forcing profiles), currently implementing HIGHTUNE/DEPHY format reading

Aims for standardisation of cases

1. netCDF file-content specification for cases (SCM and LES)
   - with validation tool (in Python)
2. Catalogue of cases
   - stored on community git repository, with all cases automatically validated on pull-request/commit
   - **Every case defined in same way from python `class BaseCase`**, with which netCDF forcing files are created
   - written and importable in python (e.g. `from moistconvection.cases import RICO`), and follows PEP conventions (e.g. code cleaned with `black` https://github.com/psf/black)

Aims for standardisation of cases

- Case implementation in python becomes documentation for case
- Python implementation of case used to generate netCDF forcing file, plots (or directly called from models)

```
class RICO(BaseCase):
    """
    Based on KNMI's synthesis of the RICO field compaign for a LES intercomparison stu

    http://projects.knmi.nl/rico/setup3d.html
    """
    SCITATION_REF = "vanZanten et al 2011"
    FORCING_IS_TRANSIENT = False

    def __init__(self):
        # surface conditions
        self.ps = 101540.  # [Pa], surface pressure
        self.p0 = 100000. # [Pa], reference pressure
        self.Ts = 299.8  # [K], sea surface temperature

        # constants
        self.Lv = 2.5e6  # [J/kg]
        self.c_p = 1005. # [J/kg/K]
        self.g = 9.81    # [m/s^2]
        self.R_d = 287.  # [J/kg/K]

        self.z_max = 4e3

    # forcing
    @np.vectorize
    def ddt_theta_l__ls(z):
        """
        Large Scale Horizontal Liq. Water Pot. Temperature Advection combined
        with Radiative Cooling [K/s]
        """
        if z > 0:
            return -2.5 / 86400
        else:
            return 0.0
```

Ongoing relevant work

- `lagtraj` (https://github.com/EUREC4A-UK/lagtraj)
  - Extract from model data (currently ECMWF ERA5) along calculated trajectories the relevant atmospheric forcings to carry out Lagrangian study of cloud development
  - Will use HIGHTUNE/DEPHY output format
- `moistconvection` (https://github.com/leifdenby/moistconvection/)
  - Python-based moist convection cases catalogue, with 7 cases currently. Development slow, but could merge with HIGHTUNE/DEPHY repository.