CORDEX-WRF: creation of a module for WRF to provide the extra CORDEX output

L. Fita[1], R. Pennel[2], J. Polcher[2], K. Béranger[2,5], S. Bastin[3], T. Arsouze[2,4], E. Katragkou[6], G. Sofiadis[6], T. M. Giannaros[7], T. Lorenz[8], and J. Milovac[9]

[1]Centro de Investigaciones del Mar y la Atmósfera (CIMA), CONICET-UBA, CNRS UMI-IFAECI, C. A. Buenos Aires, Argentina
[2]Laboratoire de Météorologie Dynamique (LMD), IPSL, CNRS, École Polytechnique, Palaisseau, France
[3]Laboratoire Atmosphères, Milieux, Observations Spatiales (LATMOS), IPSL, CNRS, Guyancourt, France
[4]École Nationale Supérieure de Techniques Avancées (ENSTA ParisTech) , 828, Boulevard des Maréchaux , 91762 Palaiseau Cedex , France
[4]Université Grenoble Alpes (UGA), CNRS, IRD, IGE, Grenoble, France
[6]Dpt. of Meteorology and Climatology, School of Geology, Aristotle University of Thessaloniki (AUTH), Thessaloniki, Greece
[7]National Observatory of Athens (NOA) - Institute for Environmental Research and Sustainable Development (IERSD), Penteli, Greece
[8]Uni Research, the Bjerknes Centre for Climate Research, Bergen, Norway
[9]Institute of Physics and Meteorology, University of Hohenheim, Stuttgart, Germany

July 10, 2018

Abstract

CORDEX requirements of data for stake holders and decision making community, push the output of the atmospheric models, which demands that usually require time consuming post-process of the standard model output. In order to avoid this time and effort consuming post-processing task, here is presented the implementation of a new module into the Weather and Forecasting Model (WRF, http://www.mmm.ucar.edu/wrf/users/ Skamarock et al., 2008) module called `module_diag_cordex` with which is expected to substantially limit the need of post-processing.

A WIKI version of this document can be found at:

http://wiki.cima.fcen.uba.ar/mediawiki/index.php/CDXWRF

In order to get the code send an email to: lluis.fita[a]cima.fcen.uba.ar in order to keep a track and being able to inform of new versions/corrections.

Disclaimer
`Authors decline any responsibility of the possible unexpected consequence of the use of this software. This piece of code is provided following the scientific spirit of sharing knowledge and technical advances. Please, use it with the same intention and willingness.`

There are three working versions of the code for WRFV3.7.1, WRFV3.8.1 and WRFV3.9.1.1. Different tests seem to show that the module slows model performance by a 40% (depending on namelist, compilation, ...)

Be aware that certain surface variables and their statistics (clWRF Fita et al., 2010) are retrieved from namelist configuration (from WRF users web page[1])

```
4.  output_diagnostics = 1 in &time_control.  Climate diagnostics.  This option
outputs 36 surface diagnostic variables:  maximum and minimum, times when max and
min occur, mean value, standard deviation of the mean for T2, Q2, TSK, U10, V10,
10 m wind speed, RAINCV, RAINNCV (the last two are time-step rain).  The output
goes to auxiliary output stream 3, and hence it needs the following:

auxhist3_outname = "wrfxtrm_d<domain>_<date>"
auxhist3_interval = 1440, 1440,
frames_per_auxhist3 = 100, 100,
```

[1] http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_V3/users_guide_chap5.htm#_Description_of_Namelist

Contents

1 CORDEX requirements

CORDEX requirements of data must cover all the possible needs of stake holders, and scientists working on the adaptation and mitigation strategies. They are grouped in different levels of frequency and priority. A working copy of this list is available here: `https://www.hymex.org/cordexfps-convection/wiki/doku.php?id=protocol` from the CORDEX convection permitting Flag Ship Pilot study.

Some of the variables are not directly computed in the WRF model which require to extend the model output in order to provide enough variables to post-process the variables.

The implementation of the `module_diag_cordex` module should allow to avoid the post-processing by computing the CORDEX-required (Core & Tier) variables during model integration

NOTE: Be aware that any systematic checking process has been applied to this module. Use it and the variables therein at your own risk !! It has been tested on a 2-nested domain configuration with the inner domain at cloud resolving resolution ($< 5\ km$, without cumulus scheme), making use of restarts and on serial, pure distributed memory and hybrid distributed/shared parallel environment

2 module_diag_cordex

The module is basically based on two modules:

- `phys/module_diag_cordex.F`: Main module which manages the calls to the variables and the accumulations for the means, ...
- `phys/module_diagvar_cordex.F`: Module with the computation of all the variables

This module is accompanied with a new `Registry/registry.cordex` where the variables and a new section in the `namelist.inpt` labeled `cordex` are defined. There are other necessary complementary modifications on `phys/module_diagnostics_driver.F` encompassed by the pre-compilaton flag `CORDEXDIAG`, as well some modifications in the `main/depend.common` and `phys/Makefile`. Output is provided by the auxiliary history output #9 with a provisional file name: `wrfcordex_d<domain>_<date>` All that variables which are only required at output time step, are computed only at that exact time.

2.1 Additional: pressure levels interpolation

At the same time, WRF can output on pressure levels while integration. However, initial version of the module does not include required CORDEX variables:

- `wa` vertical wind speed [ms^{-1}]
- `hus` specific humidity [1]
- `uer` Earth-rotated x-component wind speed [ms^{-1}]
- `ver` Earth-rotated y-component wind speed [ms^{-1}]
- `ws` wind speed [ms^{-1}]

It has been accomplished after modifying the codes: `Registry/registry.diags`, `phys/module_diagnostics_driver.F`, `phys/module_diag_pld.F` and `dyn_em/start_em.F`. The three latest modifications are also encapsulated within pre-compilation flag `CORDEXDIAG`.

See more details in how to activate this option in WRF users web-page over the namelist section `diags&`.

2.2 Additional: water budget

It has been added also the components of the atmospheric water budget. They are accumulated internally and vertically integrated allover the column. In order to provide this capability, a series of modifications have been introduced in `dyn_em/solve_em.F`

3 Installation

These steps must be followed prior the re-compilation of the WRF model and assuming that the process is started where the code resides (`WRFV3`). **NOTE:** make sure that the already compiled version of WRF and the version of the module are the same!

1. Untar the file

```
$ tar xvfz WRF_CORDEX.tar.gz
```

2. It deflates all the required files and the modified orignal WRF files

```
main/depend.common
dyn_em/solve_em.F
dyn_em/start_em.F
phys/module_diagnostics_driver.F
phys/module_diag_cordex.F
phys/module_diagvar_cordex.F
phys/module_diag_pld.F
phys/Makefile
README.cordex
Registry/registry.cordex
Registry/registry.diags
```

3. On the `Registry/Registry.EM` add the following line (after the line with include `registry.bdy_perturb` (on WRFV3.8.1), `registry.new3d_wif` (on WRFV3.9.1.1))

```
include registry.cordex
```

4. Clean the code (in order to avoid to run again configure one can make a copy of the `'configure.wrf'` and recover it after the clean, otherwise it is erased)

```
$ cp configure.wrf configure.cordex.wrf
$ ./clean -a
$ cp configure.cordex.wrf configure.wrf
```

5. edit the `` `configure.wrf' `` and add the line (after the line `-DNETCDF` and/or `CLWRFGHG`)

```
-DCORDEXDIAG
```

6. Set up (or not) the pre-compilation variable `CDXWRF` (after the line `-DCORDEXDIAG`)

```
-DCDXWRF=[value]
```

Accordingly to the value given to the pre-compilation variable `CDXWRF` one obtains:

- Without adding the variable: all CORDEX 'Core' variables
- `CDXWRF=1` CORDEX 'Tier' variables: clivg, clivh, zmla, [cape/cin/zlfc/plfc/lidx]{min/max/mean}
- `CDXWRF=2` The same as with `CDXWRF=1` and additional variables: ua, va, ws, ta, press, zg, hur, hus, tfog, fogvisblty{min/max/mean}, tds{min/max/mean} and the Water-Budget relarted ones: wbacdiabh, wbacpw, wbacpw[c/r/s/i/g/h], wbacf, wbacf[c/r/s/i/g/h], wbacz, wbacz[c/r/s/i/g/h], wbacdiabh{l/m/h}, wbacpw{l/m/h}, wbacpw[c/r/s/i/g/h]{l/m/h}, wbacf{l/m/h}, wbacf[c/r/s/i/g/h]{l/m/h}, wbacz{l/m/h}, wbacz[c/r/s/i/g/h]{l/m/h}

Simultanesouly, one needs to modify the `Registry/registry.cordex` accordingly to the value of `CDXWRF`:

- Without adding `CDXWRF`, nothing needs to be changed
- Adding `CDXWRF=1`, one needs to remove the comment `##CDXWRF1##` at the beginning of the line of the definition of certain variables
- Adding `CDXWRF=2`, one needs to remove the comment `##CDXWRF1##` and `##CDXWRF2##` at the beginning of the line of the definition of certain variables

7. Additionally, one can also get the instantaneous values for the variables which only certain statistics (accumulation, minimum, mean, ...) are provided. In order to get them, one need to:

   (a) Search in `phys/module_diagnostics_driver.F` and `phys/module_diag_cordex.F` the lines of code marked with `'INSTVALS'` and change accordingly.

   (b) Modify `Registry/registry.cordex` accordingly (removing `##INST##` at the beginning of the line of the definition of certain variables, and adding `'h9'` to certain others)

8. compile as always

```
$ ./compile em_real >& compile.log
```

4 Usage

These are the steps to use the module

1. One need to add to the 'namelist.input' the auxiliar output number 9 (e.g. for output every 3 hours and 1-day files) at the `&history' section:

```
 auxhist9_outname = "wrfcdx_d<domain>_<date>"
auxhist9_interval = 180, 180,
frames_per_auxhist9 = 8, 8,
io_form_auxhist9 = 2
```

2. Also a new section should be added (assuming it will get complex and different implementations of the diagnostics might be necessary...)

```
&cordex
output_cordex              = 1
psl_diag                   = 1:  sea-level pressure diagnostic following
hydrostatic Shuell correction
                           = 2:  psl diagnostic following a target pressure
                           = 3:  psl diagnostic following ECMWF method (default)
psmooth                    = 5:  passes of neighborgh filtering (3x3-grid point
                                 mean) of psfc for psl_diag=2 (default 5)
ptarget                    = 70000.:  pressure [Pa] target to be used by
                                 psl_diag=2 (default 70000.)
wsgs_diag                  = 1:  wind-gust diagnostic following Brasseur, 2001,
                                 MWR (default)
                           = 2:  wsgs folllowing heavy precipitation method
output_wb                  = 1:  whether water-budget variables have to
                                 computed (1) or not (0, default)
wsz100_diag                = 1:  wind extraoplation at z100m_wind using
                                 power-law method (default)
                           = 2:  wind extraoplation at z100m_wind using
                                 Monin-Obukhov theory
z100m_wind                 = 100.:  height to extraplate winds (100.  default)
zmlagen_dqv                = 0.1:  percentage of variation of mixing ratio to
                                 determine mixed layer depth used in zmla
                                 computation (0.1 default)
zmlagen_dtheta             = 1.5:  increment in K of potential temperature
                                 from its minimum within the MLD used in zmla
                                 computation (1.5 default)
potevap_diag               = 1:  potential evapotranspiration following
                                 Penman-Monteith formulation after ORCHIDEE
                                 implementation
convxtrm_diag              = 0:  diagnostic of extremes from convection
                                 indices:  0:  No (default); 1:  yes
fogvisibility_diag         = 1:  diagnostic of visibility inside fog
                                 following Kunkel (1984)
                           = 2:  RUC method (Smirnova et al., 2000)
                           = 3:  FRAML 50% prob Gultepe and Milbrandt,
                                 (2010) (default)
fogvars                    = 1:  variables to use to diagnose fog using 3D [hur]
(default)
                           = 2:  sfc [hus] (not available for Kunkel,
                                 1984)
/
```

4.1 Pressure interpolation

Remember to activate section `&diags` in order to get pressure-level vertical interpolation of state variables (g.e.: assuming 6 levels only and output every 3 hours)

```
&time_control
(...)
auxhist23_outname="wrfpress_d<domain>_<date>"
io_form_auxhist23 = 2,
auxhist23_interval = 180, 180,
frames_per_auxhist23 = 100, 100,

(...)
/
(...)

&diags
p_lev_diags = 1,
num_press_levels = 6,
press_levels = 100000, 92500, 85000, 70000, 50000, 20000
use_tot_or_hyd_p = 1
p_lev_missing = -999.
/
```

5 Variables

These are the different variables added and their implementations from the WRF point of view. There might be necessary to revise some of them, or even decide which version to use.

In case of accumulation/mean they are also be included

These variables are:

- Instantaneous diagnostics (only computed on output times)
  - Core
    - `prw`: Total water path
    - `clwvi`: Total liquid water path (QCLOUD + QRAIN)
    - `clivi`: Total ice water path (QSNOW+QICE+GRAUPEL+QHAIL)
    - `uas`: 10m earth-rotated eastward wind [ms-1]
    - `vas`: 10m earth-rotated northward wind [ms-1]
    - `wss`: 10m wind speed [ms-1]
    - `hurs`: 2m relative humidty [1]
    - `huss`: 2m specific humidty [1]
    - `psl`: sea level pressure [Pa] (three different ways)
    - `mrso`: total soil moisture content [kgm-2]
    - `slw`: total liquid water content [kgm-2]
    - `ws100`: 100m wind speed [ms-1]
    - `uz100`: 100m wind x-direction [ms-1]
    - `vz100`: 100m wind y-direction [ms-1]
    - `tauu`, `tauuv`: components of the downward wind stress at 10 m [m2s-2] (might be zero if `sf_sfclay_physics /= 1, 5`)
    - `cdcdx`: drag coefficient [-] (might be zero if `sf_sfclay_physics /= 1, 5`)
    - `tauugen`, `tauuvgen`: generic components of the downward wind stress at 10 m [m2s-2]
    - `cdgen`: generic drag coefficient [-]
    - `ps`: surface pressure [Pa]
    - `ts`: skin temperature [K]
  - Tier1
    - `zmla`: pbl height following a generic method [m]
    - `clgvi`: Total graupel path (QGRAUPEL)
    - `clhvi`: Total hail path (QHAIL)
  - Additional
    - `ua`: 3D earth-rotated eastward wind [ms-1]
    - `va`: 3D earth-rotated northward wind [ms-1]
    - `ws`: 3D wind speed [ms-1]
    - `ta`: 3D air-temperature [K]
    - `press`: 3D air pressure [Pa]
    - `zg`: 3D geopotential height [m]
    - `hur`: 3D relative humidty [1]
    - `hus`: 3D specific humidty [1]
  - Only via changes in the registry
    - `clt`: total cloud cover [1][2]
    - `cll`: low-level cloud cover [1]

[2]**NOTE:** CLDFRAC is computed by the radiative scheme thus, bear in mind to configure the `namelist.input` that: `auxhist9_interval` > `radt` otherwise one obtains repeated values of `clt`, `cll`, `clm`, `clh`

    * `clm`: mid-level cloud cover [1]
    * `clh`: high-level cloud cover [1]
    * `cape`: Convective Available Potential Energy [Jkg-1]
    * `cin`: Convective inhibition [Jkg-1]
    * `zlfc`: Height at the Level of free convection [m]
    * `plfc`: Pressure at the Level of free convection [Pa]
    * `li`: Lifted index [1]

- Accumulated or similar time dependency (computed at every time-step). They are initialized after each output time-step. Thus, they represent statistics (mean, accumulation) only from between output time-steps.
  - Core
    * `cltmean`: mean clt
    * `cllmean`: mean cll
    * `clmmean`: mean clm
    * `clhmean`: mean clh
    * `wsgsmax`: maximum surface wind gust [ms-1] (two different methods)
    * `ugsmax`: eastward maximum surface gust wind direction [ms-1]
    * `vgsmax`: northward maximum surface gust wind direction [ms-1]
    * `wsgspercen`: percentage of times when grid point got gust wind [%]
    * `totwsgsmax`: maximum surface wind gust [ms-1] (addition of different methods)
    * `totugsmax`: eastward maximum surface gust wind direction [ms-1]
    * `totvgsmax`: northward maximum surface gust wind direction [ms-1]
    * `totwsgspercen`: percentage of times when grid point got total gust wind [%]
    * `wsz100max`: maximum 100m wind [ms-1] (two different methods)
    * `uz100max`: eastward maximum 100m wind direction [ms-1]
    * `vz100max`: northward maximum 100m wind direction [ms-1]
    * `sund`: sunshine length [s]
    * `rsds`: mean surface Downwelling Shortwave Radiation [Wm-2]
    * `rlds`: mean surface Downwelling Longwave Radiation [Wm-2]
    * `hfls`: mean surface Upward Latent Heat Flux [Wm-2]
    * `hfss`: mean surface Upward Sensible Heat Flux [Wm-2]
    * `rsus`: mean surface Upwelling Shortwave Radiation [Wm-2]
    * `rlus`: mean surface Upwelling Longwave Radiation [Wm-2]
    * `rsusgen`: mean generic surface Upwelling Shortwave Radiation [Wm-2]
    * `rlusgen`: mean generic surface Upwelling Longwave Radiation [Wm-2]
    * `evspsbl`: mean evaporation [kgm-2s-1]
    * `evspsblpot`: mean potential evapotranspiration [kgm-2s-1]
    * `snc`: mean snow area fraction [%]
    * `snd`: mean snow depth [m]
    * `mrros`: mean surface Runoff [kgm-2s-1]
    * `mrro`: mean total Runoff [kgm-2s-1]
    * `mrsol`: mean total water content of soil layer [kgm-2]
    * `pr`: precipitation flux [kgm-2s-1]
    * `prl`: large scale precipitation flux [kgm-2s-1]
    * `prc`: convective precipitation flux [kgm-2s-1]
    * `prsh`: shallow-cumulus precipitation flux [kgm-2s-1]
    * `prsn`: solid precipitation flux [kgm-2s-1]

    * `snw`: accumulated snow [ksm-2]
    * `rsdt`: Top Of the Atmosphere incident shortwave radiation [kgm-2]
    * `rsut`: TOA outgoing shortwave radiation [kgm-2]
    * `rlut`: TOA outgoing Longwave radiation [kgm-2]
- Tier1
    * `capemin`: minimum CAPE [Jkg-1]
    * `cinmin`: minimum CIN [Jkg-1]
    * `zlfcmin`: minimum height at LFC [m]
    * `plfcmin`: minimum Pressure at LFC [Pa]
    * `lidxmin`: minimum Lifted index [1]
    * `capemax`: maximum CAPE [Jkg-1]
    * `cinmax`: maximum CIN [Jkg-1]
    * `zlfcmax`: maximum height at LFC [m]
    * `plfcmax`: maximum Pressure at LFC [Pa]
    * `lidxmax`: maximum Lifted index [1]
    * `capemean`: mean CAPE [Jkg-1]
    * `cinmean`: mean CIN [Jkg-1]
    * `zlfcmean`: mean height at LFC [m]
    * `plfcmean`: mean Pressure at LFC [Pa]
    * `lidxmean`: mean Lifted index [1]
- Additional
    * `tfog`: time of presence of fog [s]
    * `fogvisbltymin`: minimun visibility inside fog [km]
    * `fogvisbltymax`: maximun visibility inside fog [km]
    * `fogvisbltymean`: mean visibility inside fog [km]
    * `tdsmin`: minimum 2m dew point temperature [K]
    * `tdsmax`: maximum 2m dew point temperature [K]
    * `tdsmean`: mean 2m dew point temperature [K]
- Additionally added referred to the water budget in the atmosphere (not required by CORDEX):
    * `wbacdiabh`: Water-budget vertically integrated accumulated of diabatic heating from microphysics [K]
    * `wbacpw, wbacpw[c/r/s/i/g/h]`: Water-budget vertically integrated accumulated total tendency for water vapour, cloud, rain, snow, ice, graupel, hail [mm]
    * `wbacf, wbacf[c/r/s/i/g/h]`: Water-budget vertically integrated accumulated horizontal advection for water vapour, cloud, rain, snow, ice, graupel, hail [mm]
    * `wbacz, wbacz[c/r/s/i/g/h]`: Water-budget vertically integrated accumulated vertical advection for water vapour, cloud, rain, snow, ice, graupel, hail [mm]
    * `wbacdiabh{l/m/h}`: Water-budget vertically integrated accumulated of diabatic heating from microphysics at low, medium and high levels (same as cloudiness) [K]
    * `wbacpw[v/c/r/s/i/g/h]{l/m/h}`: Water-budget vertically integrated accumulated total tendency for water vapour, cloud, rain, snow, ice, graupel, hail at low, medium and high levels (same as cloudiness) [mm]
    * `wbacf[v/c/r/s/i/g/h]{l/m/h}`: Water-budget vertically integrated accumulated horizontal advection for water vapour, cloud, rain, snow, ice, graupel, hail at low, medium and high levels (same as cloudiness) [mm]
    * `wbacz[v/c/r/s/i/g/h]{l/m/h}`: Water-budget vertically integrated accumulated vertical advection for water vapour, cloud, rain, snow, ice, graupel, hail at low, medium and high levels (same as cloudiness) [mm]

- Pressure interplation (Core)
  - `hus_pl`: specific humidity [1]
  - `w_pl`: vertical wind speed [ms-1]
  - `uer_pl`: Earth-rotated wind x-component [ms-1]
  - `ver_pl`: Earth-rotated wind y-component [ms-1]
  - `ws_pl`: wind speed [ms-1]

5.1 clt: total cloudiness

This variable computes the total cloudiness above a grid point taking as input the cloud fraction of a given grid cell and level.

Cloud fraction in WRF is computed by the radiative scheme, which is called at a frequency given by `radt` parameter (in WRF's namelist). This aspect conditions the way how this variable is computed in order to be consistent with the cloudiness that the model experience due to radiative interaction. It would be possible to compute cloud fraction at each time-step using the different methodologies used by the radiative scheme, but in order to be consistent with the radiative cloud effects, it was discarded.

The most common implementation of 'clt' found in different other models assumes *'random overlapping'* and its implemented in most of the global climate models. Here the methodology from the GCM LMDZ (`http://lmdz.lmd.jussieu.fr/?set_language=en{}` Hourdin et al., 2006) was implemented. In this GCM, calculation of the total cloudiness is done inside the subroutine `newmicro.f90`.

5.2 cllmh: low, medium and high cloudiness

This variable computes the total cloudiness above a grid point at different vertical intervals (low: $p \geq 680hPa$, medium: $680 < p \geq 400\ hPa$, high: $p < 400\ HPa$) taking as input the cloud fraction of a given grid cell.

As in the case of the 'clt' calculation from LMDZ has already been implemented as an independent subroutine. See in figure 1 the result of the implementation.

5.3 wsgsmax: Maximum Near-Surface Wind Speed of Gust

The wind gust accounts for the wind from upper levels that is projected to the surface due to instability within the boundary layer. Resultant winds are are Earth-rotated, meaning that wind directions are towards the actual geographical North and East of the Earth. In the module two complementary ways of diagnose of the variable have been implemented.

- Brasseur01: An implementation of a wind gust following Turbulent Kinetic Energy ($\mathcal{TKE}$) estimates and stability by virtual temperature (θ_v, see mainly equation 1) reproducing Brasseur (2001) from the clWRF (clWRF, `http://www.meteo.unican.es/wiki/cordexwrf/SoftwareTools/ClWrf` Fita et al., 2010) [selected in WRF 'cordex' namelist section setting the parameter `wsgs_diag = 1`]

$$\frac{1}{z_p}\int_0^{z_p} \mathcal{TKE}(z)dz \geq \int_0^{z_p} g\frac{\Delta\theta_v(z)}{\Theta_v(z)}dz \quad (1)$$

- WRF afwa diagnostics: Inside the WRF module `module_diag_afwa.F` there is an implementation of the calculation of the wind gust which only occurs as a blending of upper-level winds (around 1 km above ground $zagl$; $zagl(k_{1000}) \geq 1000\ m$, see equation 2) above a given maximum precipitation inrensity of $prate_{hr}^{mm} \geq 50\ mmh^{-1}$ [`wsgs_diag = 2`]

$$\begin{aligned} \vec{va}_{1km} &= \vec{va}(k_{1000}-1) + [1000 - zagl(k_{1000}-1)]\frac{\vec{va}(k_{1000}) - \vec{va}(k_{1000}-1)}{zagl(k_{1000})} \\ \gamma &= \frac{150 - prate_{hr}^{mm}}{100} \\ \vec{va}_{blend} &= \vec{vas}\gamma + \vec{va}_{1km} \times (1-\gamma) \end{aligned} \quad (2)$$

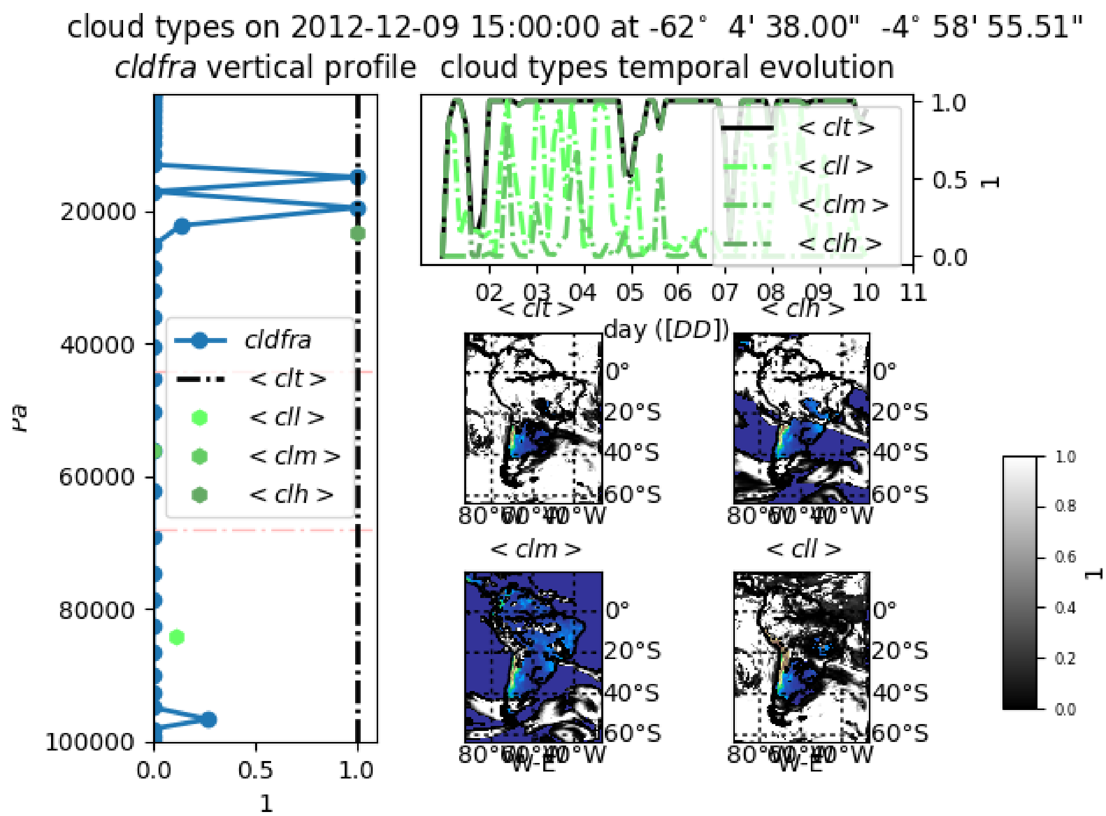

Figure 1: Vertical distribution of cloud fraction and the different cloud types at a given point (top left): cloud fraction ($cldfra$, full circles with line in blue), mean total cloud fraction ($cltmean$, vertical dashed line), mean low-level cloud fraction ($cllmean$ $p \geq 680$ hPa, dark green hexagon), mean mid-level ($clmmean$ $680 < p \geq 440$ hPa, green hexagon), mean high-level ($clhmean$ $p < 440$ hPa, clear green hexagon). Temporal evolution of cloud types at the given point (top right). Map of $cltmean$ with colored topography beneath to show-up cloud extent (middle middle), map of $clhmean$ (middle right), map of $clmmean$ (bottom middle) and map of $cllmean$ (bottom right)

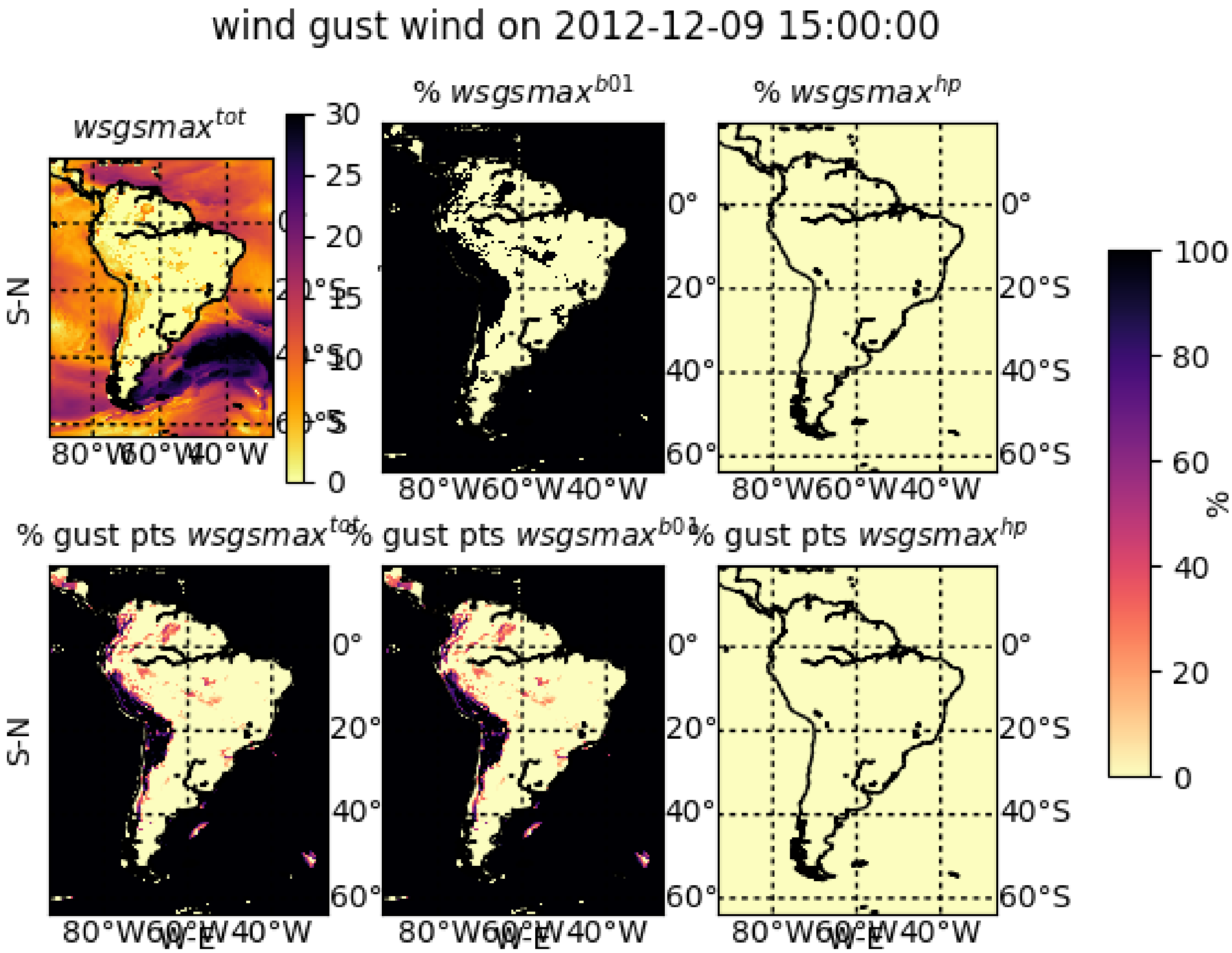

Figure 2: near surface wind gust estimates. 3h-maximum total wind gust strength ($wsgsmax^{tot}$, top left), percentage of $wsgsmax^{tot}$ due to Brasseur's application ($wsgsmax^{b01}$, top middle), percentage due to AFWA-heavy precipitation implementation ($wsgsmax^{hp}$, top right), percentage of time-steps where grid point got total wind gust (bottom left), percentage of time-steps where grid point got $wsgsmax^{b01}$ (bottom middle), percentage due to $wsgsmax^{hp}$ (bottom right)

These two methodologies have been implemented and can be switched by a new `namelist.input` parameter labeled `wsgs_diag` (in `cordex` section). Its default value is `1`

It comes out, that both methodologies provide wind gust estimation (WGE) from two different perspectives: mechanic and convective. In order to take into account both winds gusts, another variable as the addition of both estimations is provided as `totwsgsmax`, `totugsmax`, `totvgsmax`, `totwsgspercen`. On figure 2 is shown the different outcomes applying each approximation

5.4 wsgsmax100: Daily Maximum Near-Surface Wind Speed of Gust at 100 m

The wind gust at 100 m is understood that should follow a similar implementation than for the `wsgsmax`, but at 100 m, an extrapolation of such turbulent phenomena it would require a complete new set of equations which have not yet been placed.

Instead as a way to overcome it, the estimation of maximum wind speed at 100 m is provided. Provided winds are also Earth-rotated. After PhD thesis of Jourdier (2015), two different methodologies are implemented to estimate the wind at 100 m above ground:

- Following power-law wind vertical distribution, as it is depicted in equation 3 using the upper-level atmospheric wind speed below ($k^{<}_{100}$) and above ($k^{>}_{100}$) the height above ground of 100 m ($zagl$) `[wsz100_diag = 1]`

$$\begin{aligned} \vec{va}_{100} &= \vec{va}(k^{>}_{100}) \left(\frac{100.}{zagl(k^{>}_{100})} \right)^{\alpha_{x,y}} \\ \alpha_{x,y} &= \frac{\ln(\vec{va}(k^{>}_{100})) - \ln(\vec{va}(k^{<}_{100}))}{\ln(zagl(k^{>}_{100})) - \ln(zagl(k^{<}_{100}))} \end{aligned} \quad (3)$$

- Following logarithmic-law wind vertical distribution, as it is depicted in equation 4 using upper-level atmospheric wind speed below ($k^{<}_{100}$) and above ($k^{>}_{100}$) the height above ground of 100 m ($zagl$) `[wsz100_diag = 2]`

$$\begin{aligned} \ln(z_0) &= \frac{\vec{va}(k^{>}_{100}) \ln(zagl(k^{<}_{100})) - \vec{va}(k^{<}_{100}) \ln(zagl(k^{>}_{100}))}{\vec{va}(k^{>}_{100}) - \vec{va}(k^{<}_{100})} \\ \vec{va}_{100} &= \vec{va}(k^{>}_{100}) \frac{\ln(100.) - \ln(z_0)}{\ln(zagl(k^{>}_{100})) - \ln(z_0)} \end{aligned} \quad (4)$$

- Following Monin-Obukhov theory is implemented and was tested, but it is not useful for heights larger than few decameters ($z > 80.\ m$). However, the necessary code to extrapolate the wind at given height is left commented just in case someone wants to use it.

These two methodologies (Monin-Obukhov is not usable) have been implemented and can be switched by a new `namelist.input` parameter labeled `wsz100_diag` (in `cordex` section). Its default value is `1`. Even one can select another height for the estimation by providing the new parameter `z100m_wind` with a different value than 100 m (default value)

In figure 3 the different outcomes applying each approximation are shown. User is advised that there are some problems on Monin-Obukhov application under certain stable conditions (too small Obukhov length)

5.5 prw: precipitable water or water vapor path

This variable accounts for the column integrated amount of water vapor.

This one is already implemented in a old WRF tool for vertical interpolation called `p_interp.F`. The general equation following WRF standard variables as:

$$prw = \frac{mu + mub}{g} \sum_{iz=1}^{e\_vert} QVAPOR[iz](dnw[iz+1] - dnw[iz]) \quad (5)$$

where mu: perturbation dry air mass in column (Pa), mub: base-state dry air mass in column (Pa), g: gravity (ms^{-2}), e_vert: total number of vertical levels, $qvapor$: mixing ratio of water vapour ($kgkg^{-1}$), dnw: full-sigma eta-layer height ($-$). See an example on figure 4

5.6 clwvi: condensed water path

This variable provides similar information as prw, but for the liquid condensed water species. It is the same calculation as in 5, but replacing $QVAPOR$ by $QCLOUD + QRAIN$ being $QCLOUD$: condensed water mixing ratio and $QRAIN$: rain mixing ratio.

5.7 clivi: ice water path

This variable provides similar information as prw, but for the liquid condensed water species. It is the same calculation as in 5, but replacing $QVAPOR$ by $QICE + QSNOW + QGRAUPEL + QHAIL$ begin consecutively the mixing ratios of ice, snow, graupel and hail water species.

5.8 clgvi: graupel water path

This variable provides similar information as prw, but for the liquid condensed water species. It is the same calculation as in 5, but replacing $QVAPOR$ by $QGRAUPEL$. This variable is part of the 'Tier1' level and it is only accessible if pre-compilation variable `CDXWRF` is set to 1. See section **??** for more detail.

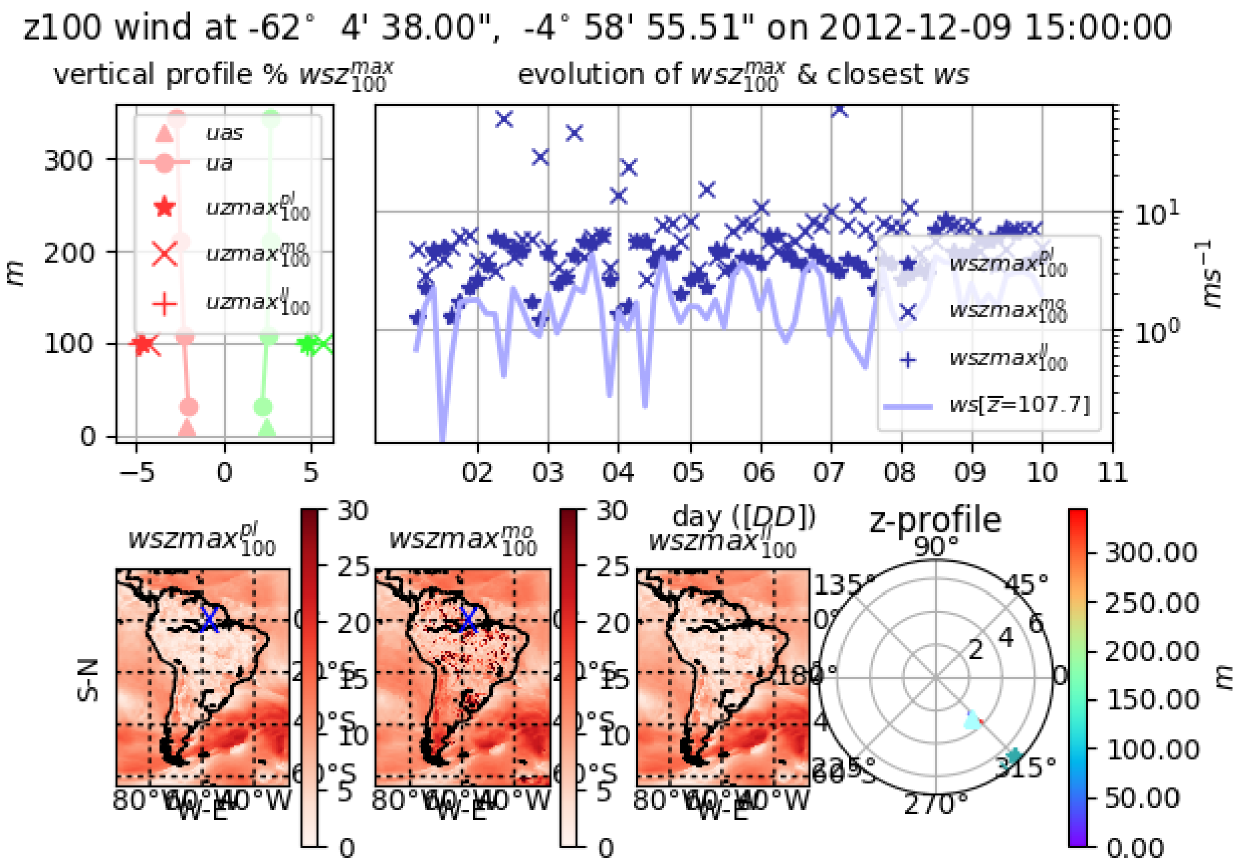

Figure 3: 100 m wind estimates. Comparison between upper-level winds and estimation at a given point and moment (upper left): 3h-maximum eastward wind (red) at 100 m by power-law ($uzmax^{pl}$, star marker), Monin-Obukhov theory ($uzmax^{mo}$, cross) by logarithmic law ($uzmax^{ll}$, sum) 10-m wind value (uas, filled triangle) and upper-level winds (ua, filled circles with line), also for the northward component (green). Temporal evolution of wind speed (top right) with all approximations and upper-level winds at the closest vertical level at 100 m (on log-y scale). Maps of both estimations (bottom left and middle) with the blue cross showing the point of previous figures. Vertical evolution of at the given point in Wind rose-like representation (bottom right)

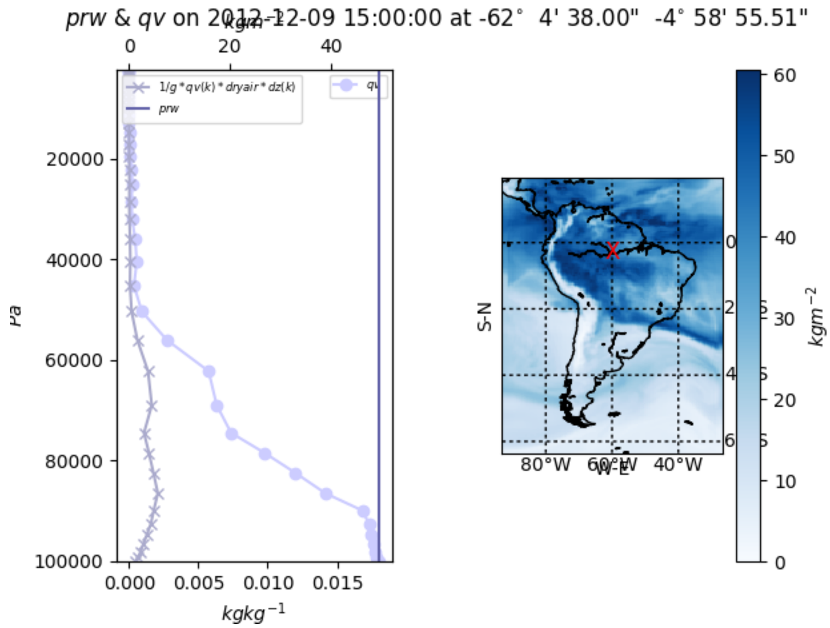

Figure 4: On a given point (left): water path (prw, vertical straight line in mm top x-axis), vertical profile of water vapour (qv, line with full circles in $kgkg^{-1}$ bottom x-axis), water path at each level (line with crosses). Map of water path (right), red cross shows where the vertical accumulation is retrieved

5.9 clhvi: hail water path

This variable provides similar information as prw, but for the liquid condensed water species. It is the same calculation as in 5, but replacing $QVAPOR$ by $QHAIL$. This variable is part of the 'Tier1' level and it is only accessible if pre-compilation variable `CDXWRF` is set to 1. See section **??** for more detail.

5.10 psl: sea level pressure

This variable accounts for the pressure at the sea level (extrapolation of the pressure at the level of the sea). It represents the value of the pressure that might have without the presence of orography.

Three different methodologies have been implemented:

- One using hydrostatic-Shuell method already implemented in the the module `phys/module_diag_afwa.F` (assuming a constant lapse-rate of 6.5 Kkm^{-1} `[psl_diag = 1]`
- Using smoothed surface pressure and a target upper-level pressure, already implemented in `p_interp.F90` `[psl_diag = 2]`
- ECMWF method taken from LMDZ from the module `pppmer.F90`, following the methodology by Mats Hamrud and Philippe Courtier from ECMWF `[psl_diag = 3]`

These three methodologies have been implemented and can be switched by a new `namelist.input` parameter labeled `psl_diag` (in `cordex` section). Its default value is `3` after CORDEX specifications of using ECMWF method. When user choices the 'ptarget' method (`psl_diag = 2`) one can select the degree of smoothing of the surface place by the selecting the number of times that the smoothing (as the mean of the point and its surrounding 8 neighbors) has to be applied (`psmooth`, default `5`) and the upper pressure to be used as target (`ptarget`, default 70000 Pa).

On figure 5 is shown the different outcomes applying each approximation. There are some problems with the ptarget methodology in both psl estimate and borders for each parallel process on applying the smoothing

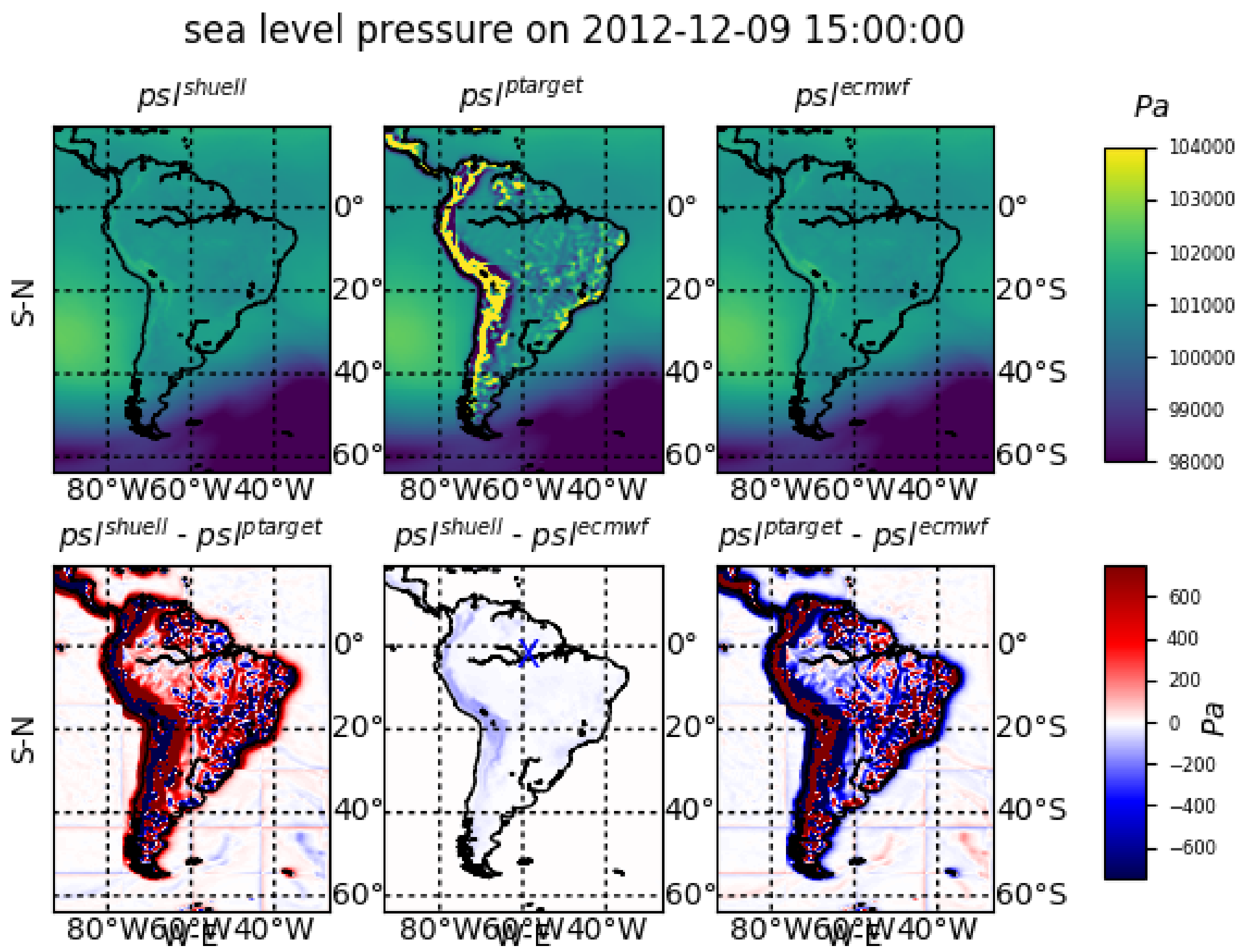

Figure 5: sea level pressure estimates. Following hydrostatic-Shuell method at a given time-step (psl^{shuell}, upper left), p-target ($psl^{ptarget}$, upper middle) and ECMWF (psl^{ecmwf}, upper right). Differences between methods $psl^{shuell} - psl^{ptarget}$ (bottom left), $psl^{shuell} - psl^{ecmwf}$ (bottom middle) and $psl^{ptarget} - psl^{ecmwf}$ (bottom right)

5.11 cape: convective available potential energy

This variable accounts for all the energy that convectively might be released.

From AMS glossary is described as[3]:

> On a thermodynamic diagram this is called positive area and can be seen as the region between the lifted parcel process curve and the environmental sounding, from the parcel's level of free convection to its level of neutral buoyancy. CAPE may be expressed as follows:
>
> $$CAPE = \int_{p_f}^{p_n} R_d(T_{vp} - T_{ve}) d \ln p \tag{6}$$
>
> where T_{vp} is the virtual temperature of a lifted parcel moving upward moist adiabatically from the level of free convection to the level of neutral buoyancy, T_{ve} is the virtual temperature of the environment, R_d is the specific gas constant for dry air, p_f is the pressure at the level of free convection, and p_n is the pressure at the level of neutral buoyancy. The value depends on whether the moist-adiabatic process is considered to be reversible or irreversible (*conventionally irreversible*, or a *pseudoadiabatic* process in which condensed water immediately falls out of the parcel) and whether the latent heat of freezing is considered (conventionally not). It is assumed that the environment is in hydrostatic balance and that the pressure of the parcel is the same as that of the environment. Virtual temperature is used for the parcel and environment to account for the effect of moisture on air density.

At this version of the module, only one implementation of the variable has been implemented. WRF model already provides a way of calculation of the variables inside the module `module_diag_afwa.F` (sinve WRF version V3.6) via the function `Buoyancy`, which at the same time it provides: Convective inhibition (CIN), Height at the Level of free convection (ZLFC), Pressure at the Level of free convection (PLFC) and Lifted index (LI). Tacking advantage of this, these extra four variables are also provided.

This vertical integrated diagnostics have a high computational cost. In order to minimize it, by default, they are only computed at output time-step. However, if user requires Tier1 variables which are related to the statistics of these diagnostics: capemin, capemax, capemean, cinmin, cinmax, cinmean, zlfcmin, zlfcmax, zlfcmean, plfcmin, plfcmax, plfcmean, limin, limax and limean, then these diagnostics are computed at all time-steps. This behavior of the module is regulated via the namelist parameter `convxtrm_diag` (defaul value is 0, meaning no computation), and setting the pre-compilation flag `CDXWRF` to 1 and perform some complementary modifications in module's Registry file `registry.cordex`. See section **??** for more detail.

5.12 cin: convective inhibition

This variable accounts for the process which inhibits the convection. Already provided by the implementation of the AFWA's CAPE calculation

From AMS glossary is described as[4]:

[3] http://glossary.ametsoc.org/wiki/Convective_available_potential_energy
[4] http://glossary.ametsoc.org/wiki/Convective_inhibition

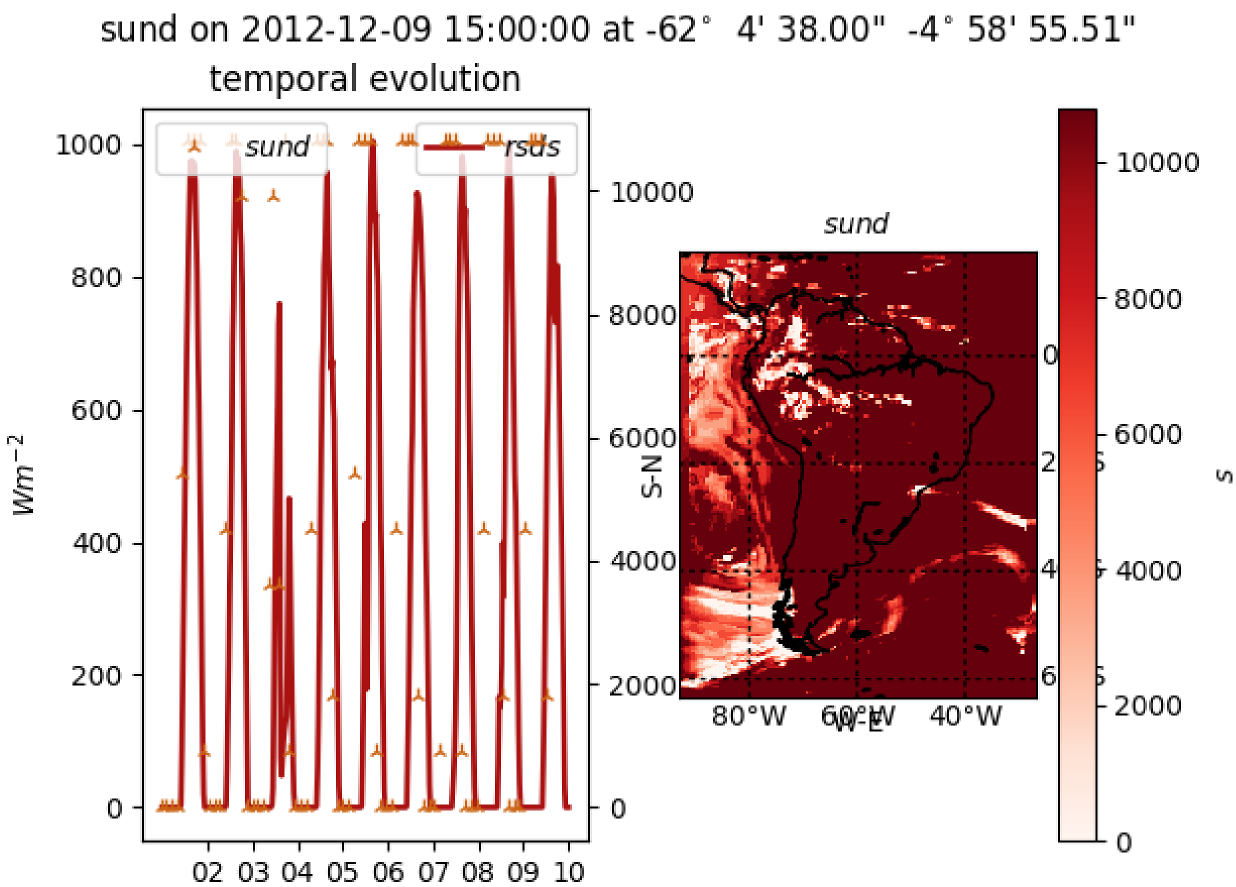

Figure 6: Temporal evolution (left, days of December 2012 on x-axis) of shortwave downward radiation (*swdown*, red line, left y-axis) and sunshine duration (*sund*, stars, right y-axis. *sund* map at a given time (right))

The energy needed to lift an air parcel upward adiabatically to the lifting condensation level (LCL) and then as a psuedoadiabatic process from the LCL to its level of free convection (LFC).
For an air parcel possessing positive $CAPE$, the CIN represents the negative area on a thermodynamic diagram. The negative area typically arises from the presence of a lid, or the amount of kinetic energy that must be added to a parcel to enable that parcel to reach the LFC. Even though other factors may be favorable for development of convection, if convective inhibition is sufficiently large, deep convection will not form. The convective inhibition is expressed (analogously to $CAPE$) as follows:

$$CIN = -\int_{p_i}^{p_f} R_d (T_{vp} - T_{ve}) d \ln p \tag{7}$$

where p_i is the pressure at the level at which the parcel originates, p_f is the pressure at the LFC, R_d is the specific gas constant for dry air, T_{vp} is the virtual temperature of the lifted parcel, and T_{ve} is the virtual temperature of the environment. It is assumed that the environment is in hydrostatic balance and that the pressure of the parcel is the same as that of the environment. Virtual temperature is used for the parcel and environment to account for the effect of moisture on air density.

5.13 sund: duration of sunshine

This variable accounts for the time where short-wave radiation is above 120 Wm^{-2}.
See results of the variable in figure 6

5.14 hur: relative humidity

3D atmospheric relative humidity on standard model η levels can be obtained following the Clausius-Clapeyron formula and its approximation from the well-known *August-Roche-Magnus* formula of saturated water vapor pressure e_s. This is an 'additional' variable which requires setting the pre-compilation flag `CDXWRF` to 2 and perform some complementary modifications in module's Registry file `registry.cordex`. See section **??** for more detail.

$$e_s = 6.1094 * e^{\frac{17.625*tempC}{tempC+243.04}} \tag{8}$$

$$w_s = \frac{0.622 * es}{presshPa - es} \tag{9}$$

$$hur = \frac{q}{ws * 1000.} \tag{10}$$

being $tempC$: temperature in Celsius degree ($°C$), $presshPa$: pressure (hPa), e_s: saturated water vapor pressure (hPa), w_s: saturated mixing ratio ($kgkg^{-1}$), q: water mixing ratio ($kgkg^{-1}$)

5.15 hus: specific humidity

3D atmospheric specific humidity [5] on standard model η levels is computed according to equation 11. This is an 'additional' variable which requires setting the pre-compilation flag `CDXWRF` to 2 and perform some complementary modifications in module's Registry file `registry.cordex`. See section **??** for more detail.

$$q = \frac{r_v}{r_v + 1} \tag{11}$$

where r_v: mixing ratio ($kgkg^{-1}$)

5.16 zg: geopotential height

This variable states for the 3D atmospheric geopotential height on standard model η levels. WRF model integrates the perturbation of the geopotential field from a reference or base one. Thus to obtain the full geopotential height is required to combine two different fields as it is shown in equation 12. This is an 'additional' variable which requires setting the pre-compilation flag `CDXWRF` to 2 and perform some complementary modifications in module's Registry file `registry.cordex`. See section **??** for more detail.

$$zg = PH + PHB \tag{12}$$

where PHB:, WRF base geopotential height (m^2s^{-2}), PH:, WRF perturbation geopotential height (m^2s^{-2})

5.17 press: air-pressure

This variable states for the 3D atmospheric pressure on standard model η levels. WRF model integrates the perturbation of the pressure field from a reference one. thus to obtain the full pressure is required to combine two different fields as it is shown in equation 13. This is an 'additional' variable which requires setting the pre-compilation flag `CDXWRF` to 2 and perform some complementary modifications in module's Registry file `registry.cordex`. See section **??** for more detail.

$$press = P + PB \tag{13}$$

where PB: WRF base pressure (Pa), P: WRF perturbation pressure (Pa)

[5] from the AMS glossary http://glossary.ametsoc.org/wiki/Specific_humidity

5.18 ta: air-temperature

This variable states for the 3D atmospheric temperature on standard model η levels. WRF model equations are based on potential temperature. Thus a conversion to actual temperature is required and it is performed as it is shown in equation 14. This is an 'additional' variable which requires setting the pre-compilation flag `CDXWRF` to 2 and perform some complementary modifications in module's Registry file `registry.cordex`. See section **??** for more detail.

$$ta = (T + 300)\left(\frac{P + PB}{p0}\right)^{R/C_p} \tag{14}$$

where T:, WRF temperature (which is as potential temperature, K), PB: WRF base pressure (Pa), P: WRF perturbation pressure (Pa), $p0$: pressure reference 100000 Pa

5.19 ua/va: air-wind Earth oriented

This variable states for the 3D atmospheric wind speed followin Earth coordinates on standard model η levels. WRF model equations use stagger winds following grid direction. In order to get actual winds following Earth geographical coordinates, a transformation shown in equation 15 is required. This is an 'additional' variable which requires setting the pre-compilation flag `CDXWRF` to 2 and perform some complementary modifications in module's Registry file `registry.cordex`. See section **??** for more detail.

$$\begin{cases} U_{unstg}(1:dimx, 1:dimy) = 0.5[U_{stg}(1:dimx-1, 1:dimy) + U_{stg}(2:dimx, 1:dimy)] \\ V_{unstg}(1:dimx, 1:dimy) = 0.5[V_{stg}(1:dimx, 1:dimy-1) + V_{stg}(1:dimx, 2:dimy)] \end{cases}$$
$$\begin{cases} ua = U_{unstg}cosa - V_{unstg}sina \\ va = U_{unstg}sina + V_{unstg}cosa \end{cases} \tag{15}$$

where U_{stg}: x-staggered WRF eastward wind (ms^{-1}), V_{stg}: y-staggered WRF northward wind (ms^{-1}), U_{unstg}: unstaggered WRF eastward wind (ms^{-1}), V_{unstg}: unstaggered WRF northward wind (ms^{-1}), $cosa$: local cosine of map rotation (1), $sina$, local sine of map rotation (1). This is an 'additional' variable which requires setting the pre-compilation flag `CDXWRF` to 2 and perform some complementary modifications in module's Registry file `registry.cordex`. See section **??** for more detail.

5.20 tauuv

Surface downdward wind stress at 10m. This variable account for the force that winds exercises to the surface. It is implemented following the equation 17. Winds are Earth-rotated.

$$tauv = \left(C_D uas^2, C_D vas^2\right) \tag{16}$$

where, C_D: drag coefficient(1) which in WRF is non-zero only for certain options of surface layer physics if `sf_sfclay_physics /= 1, 5` (1: MM5-similarity, 5: MYNN surface layer), uas: eastward 10 m surface wind (ms^{-1}), vas: northward 10 m surface wind (ms^{-1})

5.21 evspsblpot

This variable accounts for theoretical maximum evaporation that is possible to occur. Potential evapotranspiration is computed following its computation from ORCHIDEE model (Organising Carbon and Hydrology In Dynamic Ecosystems, http://orchidee.ipsl.fr/. The implementation is retrieved from the module `src_sechiba/enerbil.f90` and basically consists on an implementation of the Penman-Monteith formulation (Monteith, 1965). It is a simple formulation (see equation 18)

$$\begin{aligned} potevap &= \rho(1) qc \left(q2_{sat} - qv(1)\right) \\ qc &= C_D \sqrt{uas^2 + vas^2} \end{aligned} \tag{17}$$

where qc: surface drag coefficient (ms^{-1}), $q2_{sat}$: Saturated air at 2m ($kgkg-1$, using August-Roche-Magnus approximation and assuming to be $q2 \ \simeq qsfc$), uas, vas: 10 m wind components (ms^{-1}), C_D: Drag coefficient ($-$, only available for surface layer schemes MM5-similarity and MYNN)

Up to now there is only one implementation and it is selected via namelist parameter `potevap_diag`, up to now only with value 1 for the ORCHIDEE implementation

5.22 rsus

Surface Upwelling Shortwave Radiation, is understood as the shortwave radiation from land. It is provided accumulated by radiation schemes CAM and RRTMG (`sw_ra_scheme = 3,4`) in variable `swupb`.

5.23 rlus

Surface Upwelling Longwave Radiation, is understood as the longwave radiation from land. It is provided accumulated by radiation schemes CAM and RRTMG (`sw_ra_scheme = 3,4`) in variable `slupb`.

5.24 pr

Total precipitation flux is computed as the sum of all the types of precipitation as it is shown in equation 19

$$pr = \frac{prc + prl + prsh}{Nsteps \times \delta t} \tag{18}$$

where prc: convective precipitation (kgm^{-2}), prl: large-scale precipitation (kgm^{-2}), $prsh$: shallow-cumulus precipitation (kgm^{-2}), $Nsteps$: number of time steps and δt: time-step length (s).

5.25 prsn

Solid precipitation flux accounts for all the precipitation which is frozen. Depending on the micro-physics scheme it might account for the precipitation of: snow, graupel and hail. It is computed as it is shown in equation 20

$$prsn = \frac{pr \times sr}{Nsteps \times \delta t} \tag{19}$$

where pr: total precipitation, sr: fraction of solid precipitation (variable included in WRF), $Nsteps$: number of time steps and δt: time-step length (s).

6 Generic variables

Some of the diagnostics required by CORDEX depend on the approximations, equations, methodologies and observations used to compute them. This make model intercomparison exercises very difficult because values might differ from one implementation to another one. In a way to overcome this issue, a series of variables are also provided in a 'generic' form (when possible) meaning that they are obtained directly from well established variables. Thus, these generic form of the diagnostics become 'independent' of model's implementation.

6.1 zmlagen: generic boundary layer height

Boundary layer height is a clear example of model dependence and even scheme dependence of how a diagnostic is computed. Each pbl scheme has its own assumptions and has to be compiled in a specific way. This variable is provided with the 'Tier1' level of variables and requires to set the pre-compilation flag `CDXWRF` to 1 and perform some complementary modifications in module's Registry file `registry.cordex`. See section **??** for more details.

Pursuing this goal, we implemented a general definition for the boundary layer height as it was done in (García-Díez et al., 2013) after (Nielsen-Gammon et al., 2008). The method consists in defining the height of the boundary layer as the first level where potential temperature exceeds the minimum potential temperature reached in the mixed layer (ML) by more than 1.5 K. It has been implemented as it is shown below:

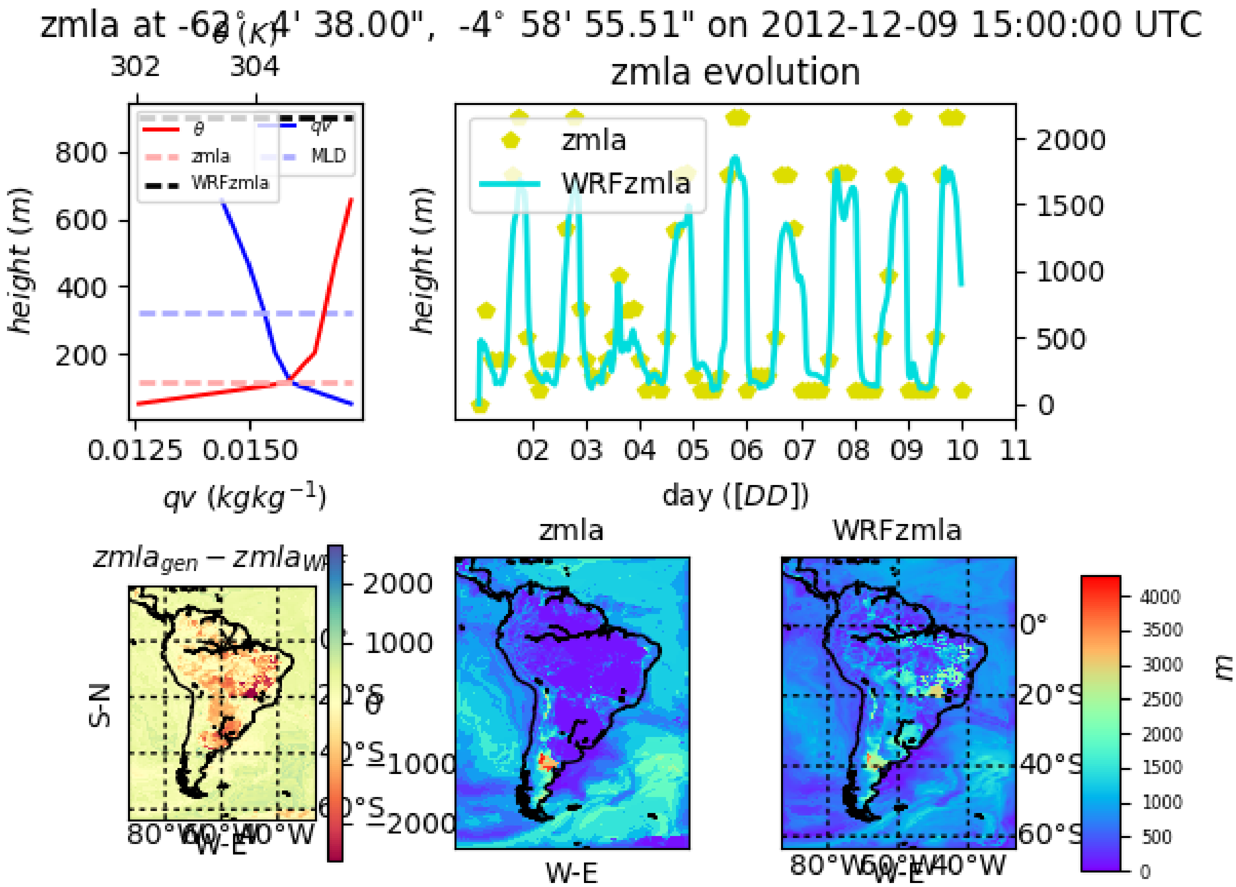

Figure 7: Vertical characteristics of the atmosphere at a given point (top left): potential temperature vertical profile (θ K, red line), vertical profile of mixing ratio (qv $kgkg^{-1}$, blue line), mixed layer depth (MLD, dashed horizontal line at 323.522 m), derived boundary layer height ($zmla$, horizontal dashed line at 107.122 m and WRF derived pbl scheme value (WRF_{zmla} at 903.017 m). Comparison of temporal evolutions (top right) between derived $zmla$ (yellow stars) and WRF's pbl scheme (blue line). Map of differences between derived and WRF simulated ($zmla - zmla_{WRF}$, bottom left), $zmla$ map (bottom middle) and $zmla_{WRF}$ (bottom right)

1. Mixed layer depth (k_{MLD}) is defined as the first layer in the atmosphere starting from the second atmospheric level, at which the variation of mixing ratio ($qv(k)$) is above from a certain value respect its value at the first level ($qv(1)$): $\frac{|qv(k_{MLD})-qv(1)|}{qv(1)} > \delta qv$ (here applied a $\delta qv \ = \ 0.1$)
2. Within the MLD the value with the minimum potential temperature is taken: $\theta min_{MLD} = min(\theta(1), ..., \theta(k_{MLD}))$
3. The level of the boundary layer height (k_{zmla}) is that one which starting from the surface the total variation of potential temperature is higher than a given value respect the minimum inside the MLD: $\theta(k_{zmla}) + \delta\theta > \theta min_{MLD}$ (here $\delta\theta \ = \ 1.5 \ K$)
4. The obtained boundary layer height ($zmla$) is obtained from the geopotential height at the level k_{zmla} above the ground ($zagl$): $zmla = zagl(k_{zmla}) = zg(k_{zmla}/g - hgt$.

Comparison of this implementation with the $zmla$ directly provided by WRF's pbl scheme is shown in figure 7. No general rule has been applied to determine the correct value of δqv used to determine depth of mixed layer. They can be determined by the `namelist.input` parameters `zmlagen_dqv` for δqv (default value 0.1) and `zmlagen_dtheta` for $\delta\theta$ (default value 1.5 K)

6.2 cdgen: generic drag coefficient

Drag coefficient at surface. Computation of drag coefficient depends on selected surface scheme. In order to avoid this scheme dependency, a general calculation of the coefficient has been introduced as it is shown in equation 21, after Garratt (1992).

$$C_D^{gen} = \left(\frac{u^*}{wss}\right)^2 \tag{20}$$

Being, u^*: from similarity theory (ms^{-1}), wss 10 m wind speed (ms^{-1})

6.3 tauuvgen: generic surface downdward wind stress

Generic surface downward wind stress at 10m.

It is implemented following the equation 22. Winds are Earth-rotated

$$tauvgen = \left(C_D^{gen} uas^2, C_D^{gen} vas^2\right) \tag{21}$$

where C_D^{gen} generic drag coefficient ($-$, see equation 21), uas: eastward 10 m wind, vas: northward 10 m wind.

6.4 rsusgen

Surface Upwelling Shortwave Radiation, is understood as the shortwave radiation from the surface. It is calculated in a generic way as the reflected shortwave radiation due to albedo as it is shown in equation 23

$$rsus = -alebdo * swdown \tag{22}$$

Being, $albedo$: albedo (1), $sdown$: downward at surface shortwave radiation (Wm^{-2})

6.5 rlusgen

Surface Upwelling Longwave Radiation, is understood as the longwave radiation from the surface. It is calculated in a generic way as the longwave radiation due to surface temperature following black body formulation as it is shown in equation 24

$$rlus = CtBoltzman * emiss * skt^4 \tag{23}$$

Being, $CtBoltzman$: Boltzman constant $(5.67051E^{-8}\ Wm^{-2}K^{-4})$, $emiss$: emissivity (1), skt: skin temperature (K)

7 Additional variables

Some other variables not required by CORDEX, but might be interesting for other purposes will be also added because it was thought that they might be useful to the community and to take advantage of all the work done for the 'Core' and 'Tier1' variables. These variables are obtained if the pre-compilation flag `CDXWRF` is set to 2 and some additional modifications are made in module's registry file `registry.cordex`. See section **??** for more details.

7.1 tds

Dew point temperature following August-Roche-Magnus approximation as it is shown in equation 25

$$\begin{aligned} \gamma &= \log(hurs) + \frac{b(tas - 273.15)}{(tas - 273.15) + c} \\ tds &= \frac{c\gamma}{b - \gamma} + 273.15 \end{aligned} \tag{24}$$

where tas: 2m temperature (K), $hurs$: 2m relative humidity (%), $b =$ 17.625, $c =$ 243.04.

Statistical values are provided in the output: minimum, maximum and mean within output time-steps as part of the 'additional' level of variables.

7.2 Water vapor balance terms

These variables cover the different terms of the water balance equation. The equation for any given water specie is given in equation 26:

$$\begin{aligned} TEN_q &= HOR_q + VER_q + MP_q \\ \frac{\partial q_q}{\partial t} &= -V_h \vec{\nabla} q_q - \ w\frac{\partial q_q}{\partial z} + \ SO_q - SI_q \end{aligned} \tag{25}$$

Where q stands for either of the five water species concentration ($kgkg^{-1}$, vapor, snow, ice, rain and liquid), Vh stands for horizontal wind speed (ms^{-1}), w stands for the vertical wind speed (ms^{-1}) and MP for the loss or gain of water due to cloud microphysical processes. The term in the left-hand side of the equation represents the water species tendency (TEN or 'PW'), referring to the difference between q at the model previous time step and at the end of the actual time step, divided by the time step. TEN equals to the horizontal advection (HOR or 'F', first term in right-hand side of the equation), the vertical advection (VER or 'Z', second term in right-hand side) and the sources (SO) or sink (SI) of atmospheric water due to microphysical processes (MP). All terms are expressed in $kgkg^{-1}s^{-1}$. However, SO, and SI ca not be provided because they are micro-physics dependent an make difficult to provide a general formula for them.

In order to obtain the total column mass of water due to each term (in units of mm), it is applied to each term of eq. 26 (similarly as in 5):

$$-\frac{1}{g}\int_{p_{sfc}}^{p_{top}} dp \tag{26}$$

Following the methodology of Huang et al. (2014) and Yang et al. (2011), Fita and Flaounas (2017) implemented the water budget terms in a new module in WRF in order to allow the computation of the terms during model integration. For the CORDEX module, only the vertically integrated variables will be implemented. Microphysics processes are reflected by the budget as sinks and sources of the different water species which are not included. This is because these processes edepend on the micro-physics scheme used during model run and it is out of the scope of the module to provide full cover of this need. It is know the the budget is closed, thus, residual of the terms must be the micor-phsysics term. Due to the complexity of each micro-physics scheme and the impossibility to generalize the calculation, the accumulation of diabatic heating from the micro-physics scheme is provided as a proxy.

All water species decomposition is shown in figures 8 and 9

It has also been grouped by vertical levels as it is done with the clouds: $p \geq 68000\ Pa$, $40000 \leq p < 68000\ Pa$, $p < 40000\ Pa$. Decomposition of each term is shown for water vapour (qv) and snow (qs) in figures from 10 to 13.

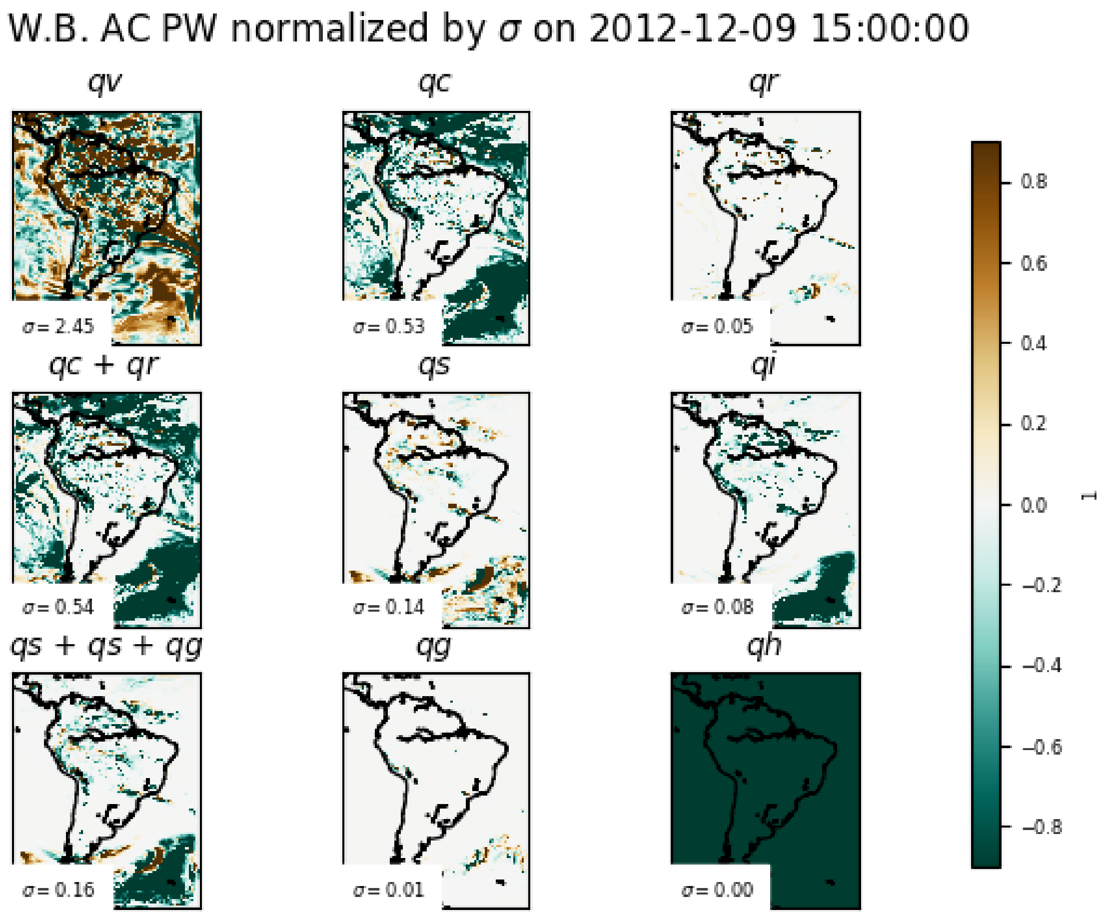

Figure 8: Normalized water budget 3h-accumulated vertically integrated total tendency 'PW' at a given time, for water vapour (qv, top left), cloud (qc, top middle), rain (qr, top right), water condensed species ($qc + qr$, middle left), snow (qs, middle middle), ice (qi, middle right), water solid species ($qs + qi + qg$, bottom left), graupel (qg, bottom middle), hail (qh, bottom right). Number on low left corner of the figure correspond to the standard deviation (σ in mm) value used for the normalization

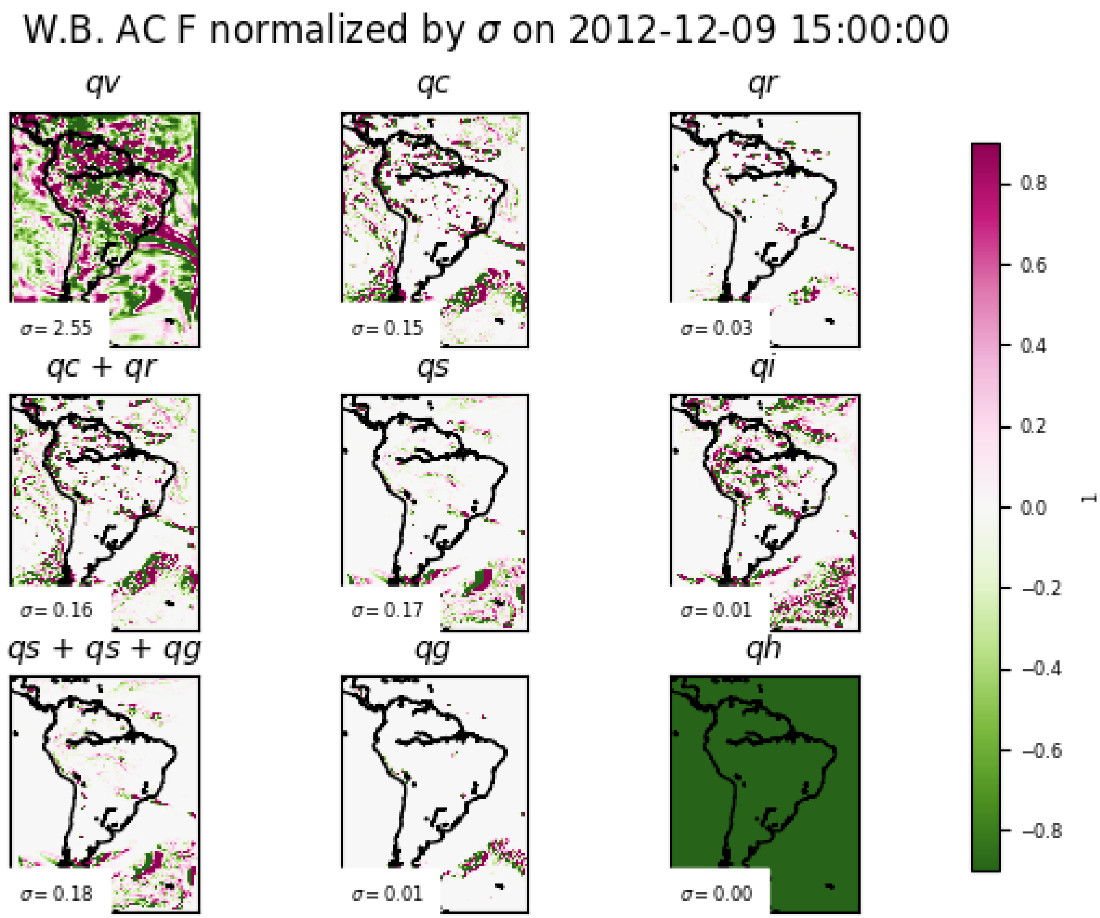

Figure 9: As in 8, but for Water budget 3h-accumulated vertically integrated horizontal advection 'F' at a given time

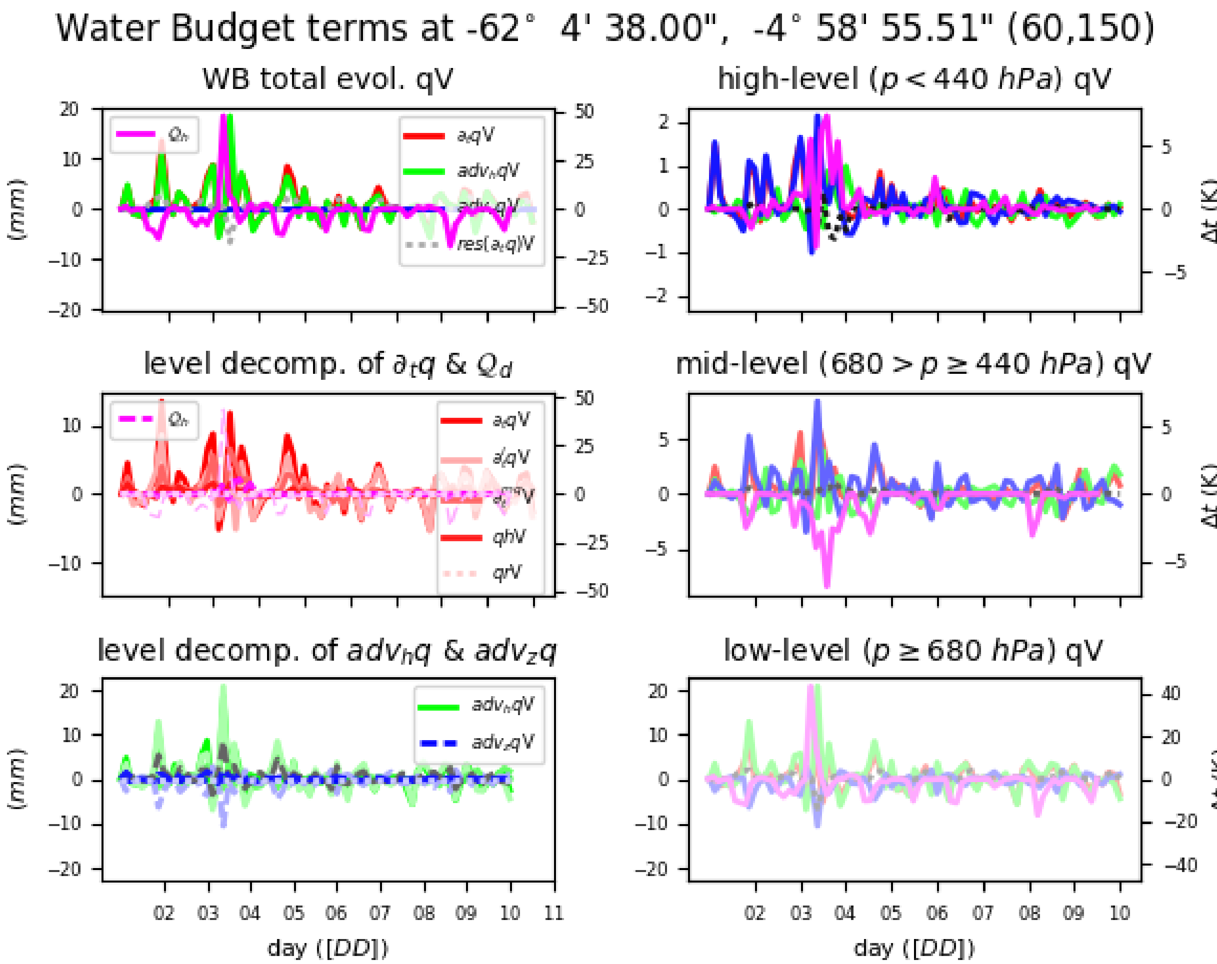

Figure 10: Water budget evolution at a given point for water vapour of vertically integrated water-budget terms: total tendency ‘PW’ ($\partial_t qv$, red), horizontal advection ‘F’ ($adv_h qv$, green), vertical advection ‘Z’ ($adv_z qv$, green), residual PW - F -Z ($res(\partial_t qv)$, gray dashed) and diabatic heating from micro-physics ($\mathcal{Q}_d$, pink) (top left), only high-level vertically integrated values ($p < 440$ hPa, top right), high/mid/low-level (degree of color intensity) decomposition of $partial_t qv$ (red) and $\mathcal{Q}_d$ (pink) and their respective residuals as dashed lines (middle left), only mid-level vertically integrated values ($680 > p \leq 440$ hPa, middle right), high/mid/low-level (degree of color intensity) decomposition of $adv_h qv$ (green) and $adv_z qv$ (blue) and their respective residuals as dashed lines (bottom left) and only low-level vertically integrated values ($p \geq 680$ hPa, bottom right)

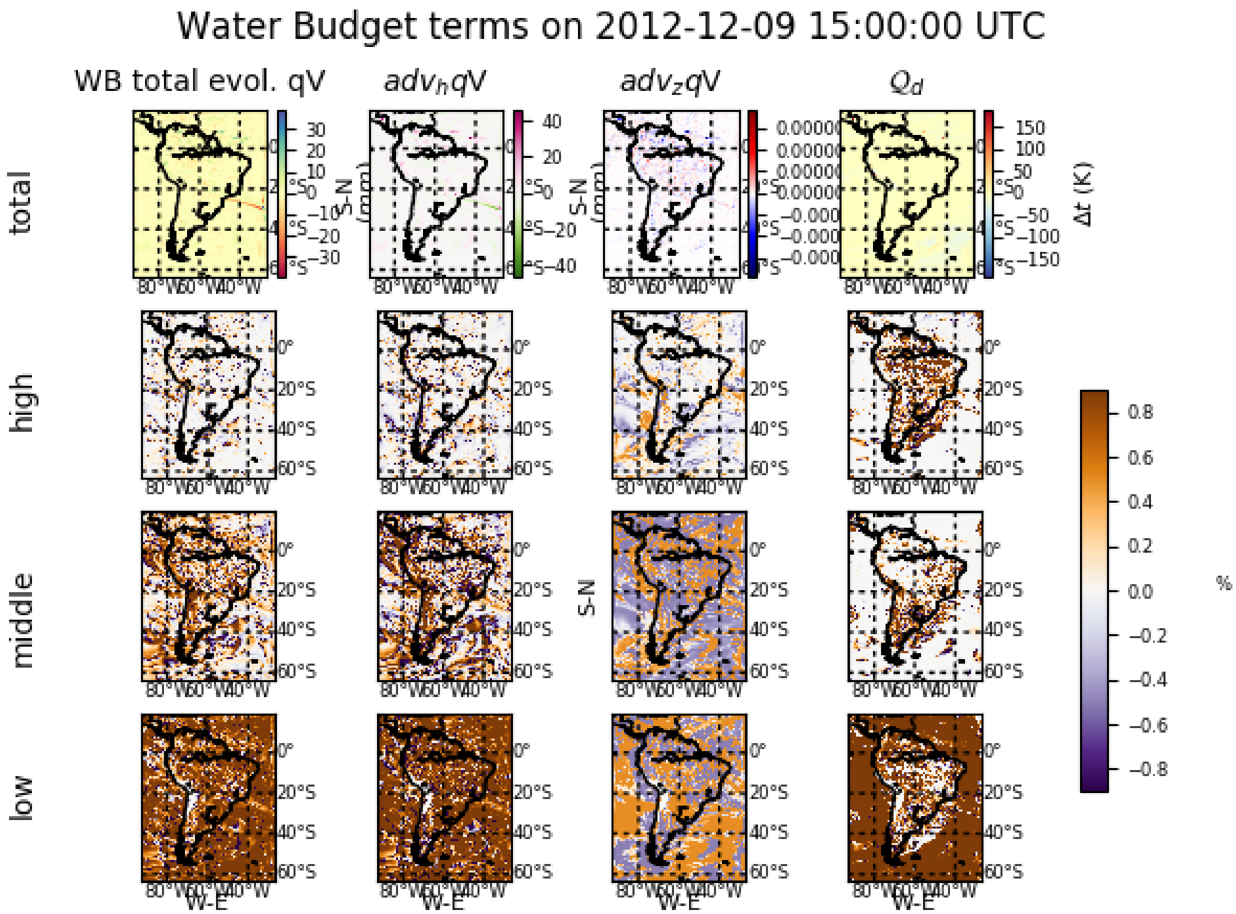

Figure 11: water vapour water budget maps of each component and diabtic heating from micro-physics at a given time and the percentual contribution at each different vertically integrated layer respective the total. total tendency 'PW' ($\partial_t qv$, first column), horizontal advection 'F' ($adv_h qv$, second col), vertical advection 'Z' ($adv_z qv$, third col.) and diabatic heating from micro-physics ($\mathcal{Q}_d$, 4th col). Percentage contribution of high level ($p < 440\ hPa$) integration to the total (second row), percentage for mid level ($680 > p \geq 440\ hPa$) integration to the total (third row) and percentage of low-level ($p \geq 680\ hPa$) integration (bottom row)

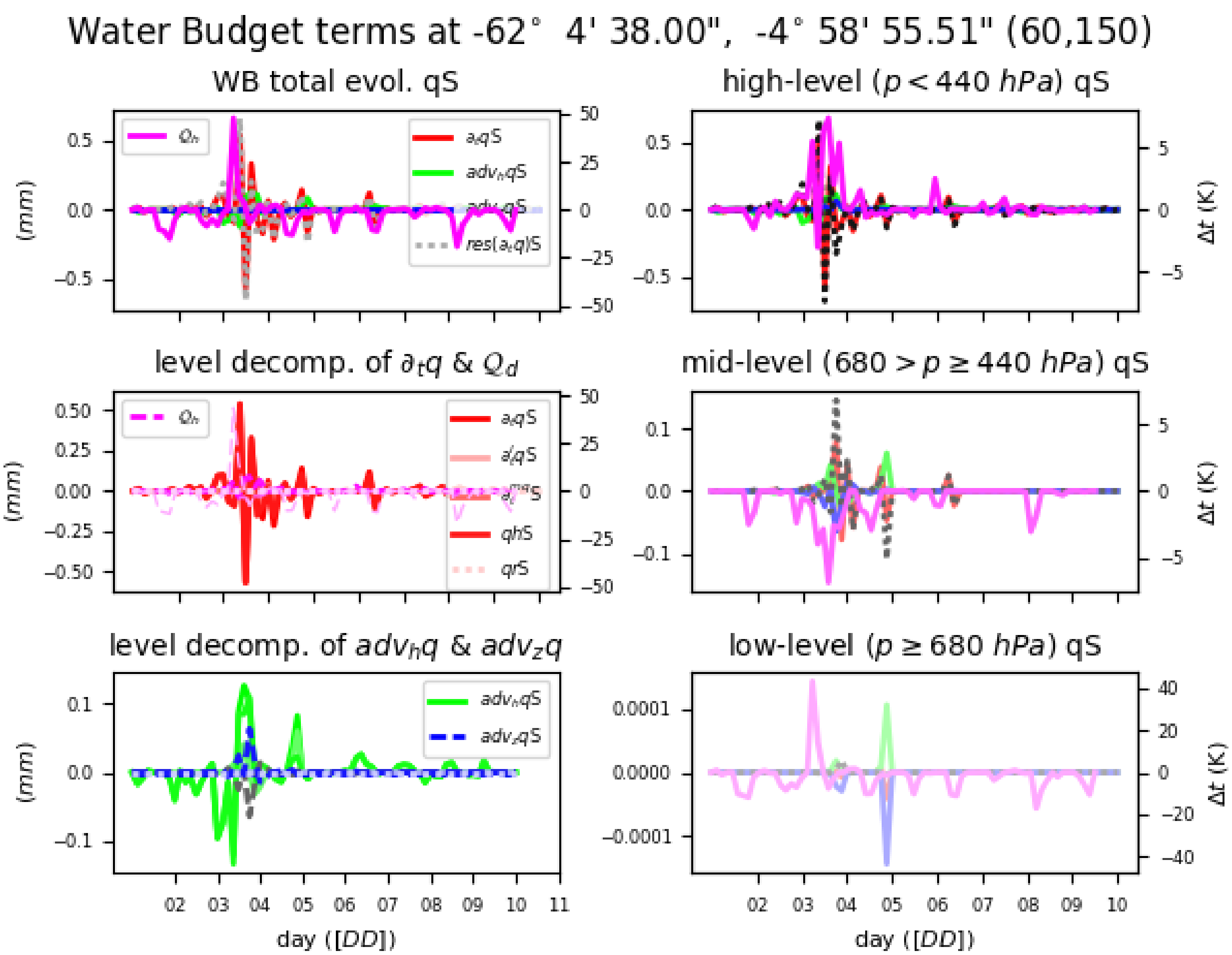

Figure 12: The same as in figure 10, but for snow

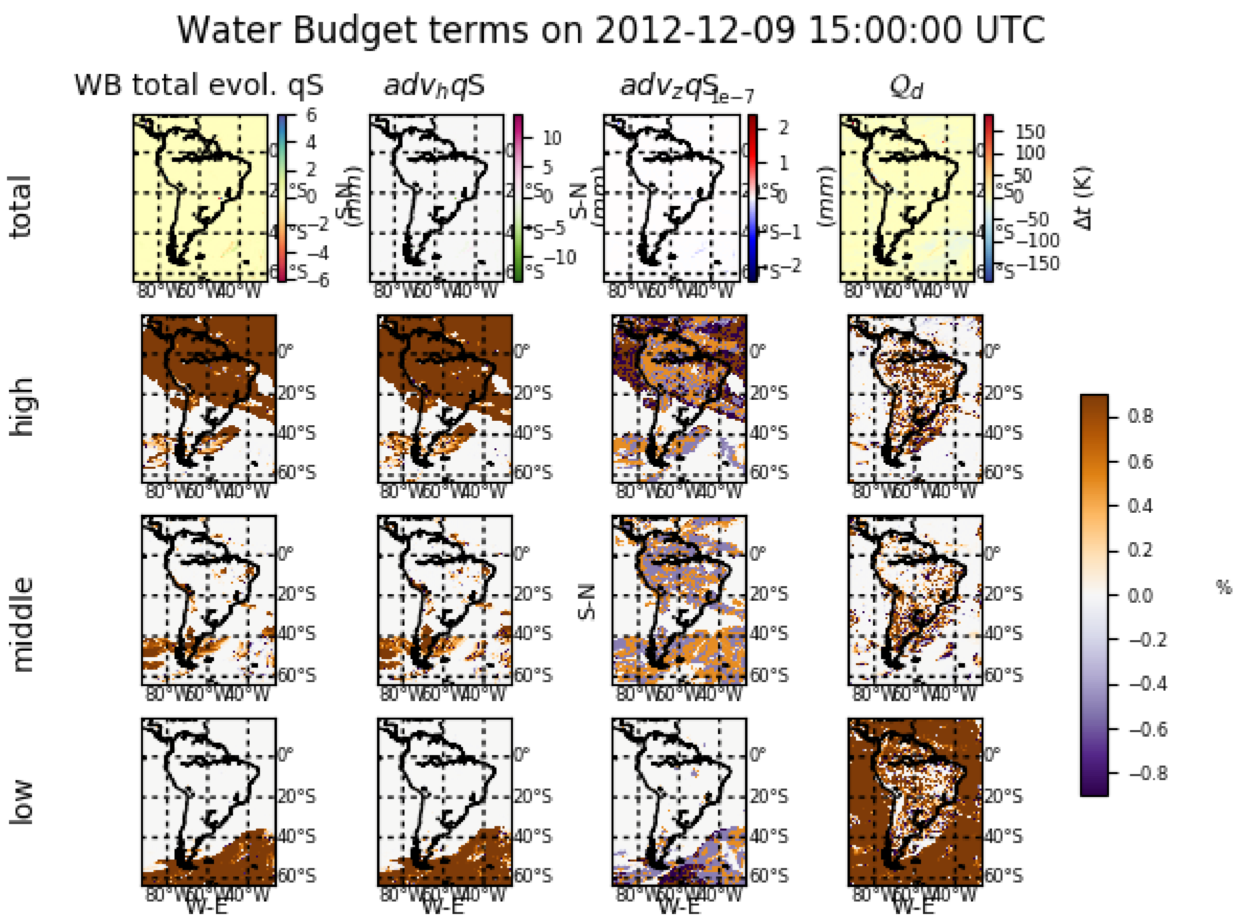

Figure 13: The same as in 11, bur for snow

7.3 tfog: time of presence of fog

A diagnostic of visibility has been introduced. From it, one can define fog as that moment where the visibility is lower than 1 km (WMO, 2010).

`tfog` accounts for the time in which the grid point has visibility lower than 1 km (see equation 28)

$$tfog = \sum_{it=1}^{\mathcal{N}_{fog}} \delta t \tag{27}$$

where $\mathcal{N}_{fog}$: number of time steps where visibility was below 1 km. δt: model time step (s)

7.4 fogvisblty: visibility inside fog

A diagnostic of visibility is introduced in order to provide a diagnostic for fog. Three different methods have been introduced:

- **K84:** Visibility is computed by means of liquid water (`QCLOUD`) and ice (`QICE`) concentrations. Following (Bergot et al., 2007) fog appears when there are liquid and/or ice water species at the lowest level. Visibility using (see equation 29 Kunkel, 1984) formula is computed on that grid points where fog appeared. Method selected with `[visibility_diag = 1]`

$$\begin{aligned} vis &= \begin{cases} visc = & 0.027(qc \times 1000)^{-0.88} \quad qcloud \neq 0 \\ visi = & 0.024(qi \times 1000)^{-1.0} \quad qice \neq 0 \end{cases} \\ vis &= min(visc, visi) \end{aligned} \tag{28}$$

  where qc: liquid water (cloud) mixing ratio ($kgkg^{-1}$), qi: ice mixing ratio ($kgkg^{-1}$). Visibility values are in km

- **RUC:** Visibility is computed using relative humidity (hur) as it is implemented in the RUC model (see equation 30 Smirnova et al., 2000). Method selected with `[visibility_diag = 2]`

$$vis = 60.0exp\left[\frac{-2.5(rh \times 100 - 15)}{80}\right] \tag{29}$$

  where rh: relative humidity (1) and can be from surface or first model layer. Visibility values are in km

- **FRAM-L:** Visibility is computed using relative humidity (hur) after (see equation 31 Gultepe and Milbrandt, 2010). In this work, it is proposed a probabilistic approach to the computation of the visibility in three different bins: 95% , 50% and 5% of probability to get certain visibility (for $rh > 30\%$). As a matter of compromise, in the module the calculation for the 50% of probability as the preferred one has been chosen. Thus, this method provides the visibility that might be with a 50% of probability. Method selected with `[visibility_diag = 3]` (default)

$$vis^{prob} = \begin{cases} 95\% & -9.68 \times 10^{-14} rh^{7.19} + 52.20 \\ 50\% & -5.19 \times 10^{-10} rh^{5.44} + 40.10 \\ 5\% & -0.000114 rh^{2.70} + 27.45 \end{cases} \tag{30}$$

  where rh: relative humidity (1) and can be from surface or first model layer. Visibility values are in km

Provided values in the output are the minimum, maximum and mean values within output time-steps

Different choices are controlled throughout `namelist.input` variables: `visibility_diag` method of visibility computation, `fogvars` source of the relative humidity. From first model layer (hur) `fogvars=1` (default value) or from surface ($hurs$) `fogvars=2`. See some preliminary results in figure 14

It is known that certain methods of visibility relay on numerical adjustments on certain observational data taken under certain circumstances and places (e.g.: for FRAM-L adjusted values come from observations from a Canadian airport). It would be desirable to provide a more generic all places/purposes (if possible) approach. It is recommended to take this value with certain care

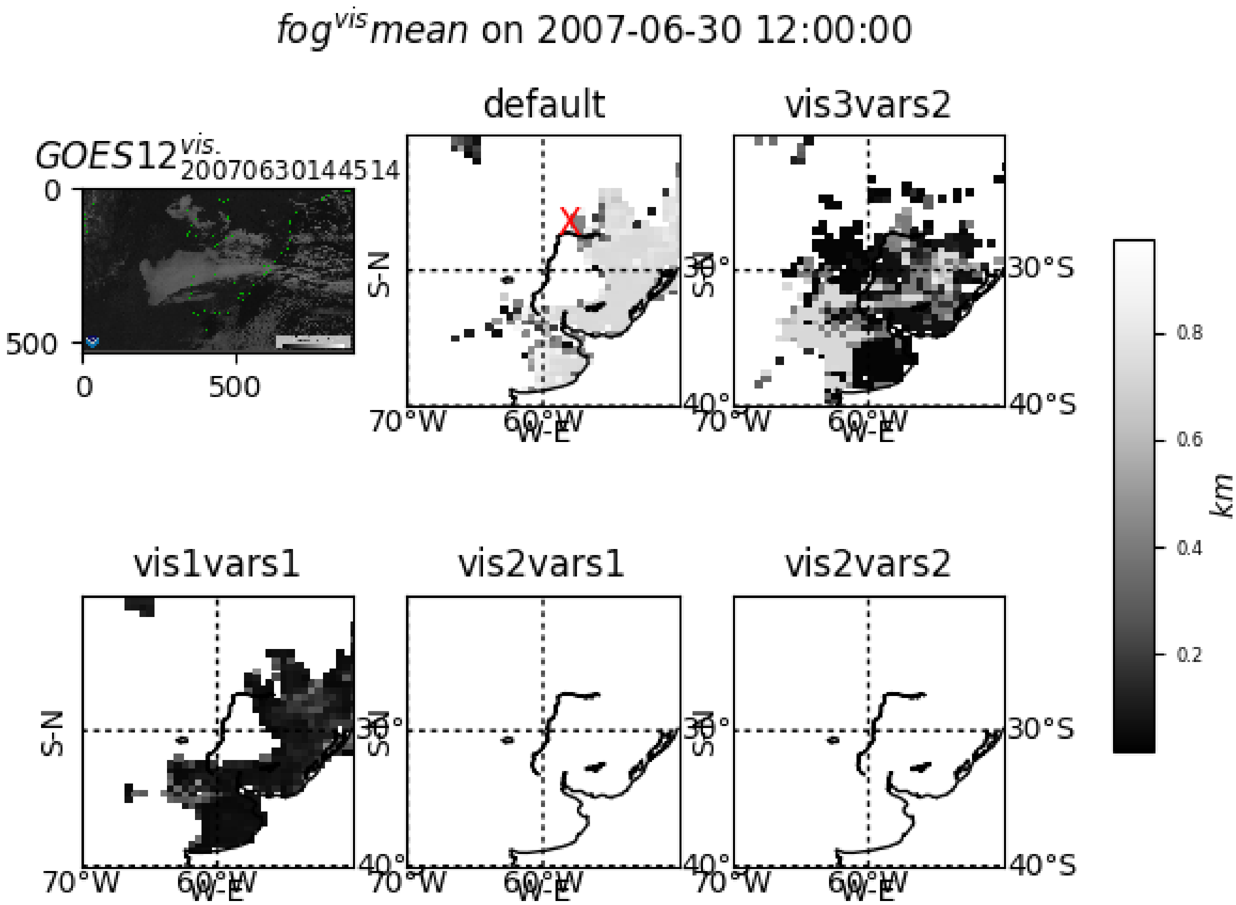

Figure 14: Comparison of the different configurations of the diagnostics of the mean fog visibility (in 1 hour) to the satellite image from GOES-12 at the same time in the visible channel (courtesy of NOAA-CLASS), default (fogvisibility=3, fogvars=1; top middle), vis3vars2 (top right), vis1vars1 (bottom left), vis2vars1 (bottom middle) and vis2vars2 (bottom right)

8 Missing variables

There are certain variables from CORDEX 'Tier1' which could not yet be introduced.

8.1 wsgsmax100: Daily Maximum Near-Surface Wind Speed of Gust at 100 m

The wind gust at 100 m is understood that should follow similar processes that the wind gust at the surface (like in `wsgsmax`). At this version of the module there has not been considered to be included in the search for the right equations and approximations.

8.2 ic_lightning, cg_lightning, tot_lightning: intra-cloud, ground and total lightning flashes

There is lightning scheme implementation in WRF. (`lightning_option` among other from `namelist.input`). However it is scheme and zone-dependant and it might require some adjustment prior it's use. It has been considered unnecessary to be added in this first version of the module.

It does not sees to provide cloud/ground discrimination

9 module_diag_cordex

See in appendix G the list of variables and their main properties added with the module into the `wrfcdx` output.

The module is basically based on two modules:

- `phys/module_diag_cordex.F`: Main module which manages the calls to the variables and the accumulations for the means, ...
- `phys/module_diagvar_cordex.F`: Module with the computation of all the variables

Different other complementary files have also been added or modified to the WRF code. This module is accompanied with a new `Registry/registry.cordex` where the variables, and namelist parameters of a new section in the `namelist.input` labeled `cordex` are defined. There are other necessary complementary modifications in other different WRF modules: `phys/module_diagnostics_driver.F` which accounts for the management of diagnostics in which the call to the main module subroutine has been added, `Registry/registry.diags` the registry file for the pressure interpolated variables in which the complementary interpolated variables have been introduced, `phys/module_diag_pld.F` the module which computes the pressure interpolation in which the complementary interpolated variables have been added, `dyn_em/start_em.F` which initialize the modified pressure interpolation. Also some modifications in the `main/depend.common` and `phys/Makefile` to get the new module compiled. For the inclusion of the water-budget variables some specific modifications have also been introduced in the `dyn_em/solve_em.F` module in order to keep the different advections of all water species. Finally a descriptive file called `README.cordex` is also provided with the description and synthesized instructions for compilation and use.

Output of the module is grouped in a single file (WRF's auxiliary history output #9) with a provisional file name: `wrfcordex_d<domain>_<date>` with the standard WRF parameters of output frequency, number of time-steps per file and format. Variables which are interpolated at pressure levels have been included in the pressure level auxiliary output file number 23 and following the file output via the namelist section `diags&`.

10 WRF output names

See in appendix G the list of variables added with the module into the `wrfcdx` output.

Acknowledgements All the coders of WRF, LMDZ, ORCHIDEE are acknowledged for their work on the developing and maintaining of the models. M. A. Jiménez from Universitat de les Illes Balears is acknowledged by her explanations on certain PBL calculations. J. Milovac from U. Hohenheim for her comments is also acknowledged. D. Argüeso from UIB. V. Galligani, J. Ruiz and M. Sebastián from CIMA.

References

Bergot, T., Terradellas, E., Cuxart, J., Mira, A., Liechti, O., Mueller, M., and Nielsen, N. W. (2007). Intercomparison of single-column numerical models for the prediction of radiation fog. *Journal of Applied Meteorology and Climatology*, 46(4):504–521.

Brasseur, O. (2001). Development and application of a physical approach to estimating wind gusts. *Monthly Weather Review*, 129(1):5–25.

Fita, L., Fernández, J., and García-Díez, M. (2010). Clwrf: Wrf modifications for regional climate simulation under future scenarios. *Proceedings of 11th WRF Users' Workshop*.

Fita, L. and Flaounas, E. (2017). Medicanes as subtropical cyclones: the december 2005 case from the perspective of surface pressure tendency diagnostics and atmospheric water budget. *Q. J. Royal Met. Soc.*, under revision.

García-Díez, M., Fernández, J., Fita, L., and Yagüe, C. (2013). Seasonal dependence of wrf model biases and sensitivity to pbl schemes over europe. *Q. J. of Roy. Met. Soc.*, 139:501–514.

Garratt, J. (1992). *The Atmospheric Boundary Layer*. Cambridge Univ. Press, Cambridge, U.K.

Gultepe, I. and Milbrandt, J. A. (2010). Probabilistic parameterizations of visibility using observations of rain precipitation rate, relative humidity, and visibility. *Journal of Applied Meteorology and Climatology*, 49(1):36–46.

Hourdin, F., Musat, I., Bony, S., Braconnot, P., Codron, F., Dufresne, J.-L., Fairhead, L., Filiberti, M.-A., Friedlingstein, P., Grandpeix, J.-Y., Krinner, G., LeVan, P., Li, Z.-X., and Lott, F. (2006). The lmdz4 general circulation model: climate performance and sensitivity to parametrized physics with emphasis on tropical convection. *Clim. Dyn.*, 27(7-8):787–813.

Huang, H.-L., Yang, M.-J., and Sui, C.-H. (2014). Water budget and precipitation efficiency of typhoon morakot (2009). *J. Atmos. Sci.*, 71:112–129.

Jourdier, B. (2015). Ressource éolienne en france métropolitaine : méthodes dâĂŹévaluation du potentiel, variabilité et tendances. *Climatologie: École Doctorale Polytechnique, 2015. Français. ph:+33 01238226*, pages 1–229.

Kunkel, B. A. (1984). Parameterization of droplet terminal velocity and extinction coefficient in fog models. *Journal of Climate and Applied Meteorology*, 23(1):34–41.

Monteith, J. L. (1965). Evaporation and environment. the state and movement of water in living organisms. *19th Symp. Soc. Exp. Biol*, pages 205–234.

Nielsen-Gammon, J. W., Powell, C. L., Mahoney, M. J., Angevine, W. M., Senff, C., White, A., Berkowitz, C., Doran, C., and Knupp, K. (2008). Multisensor estimation of mixing heights over a coastal city. *Journal of Applied Meteorology and Climatology*, 47(1):27–43.

Skamarock, W. C., Klemp, J. B., Dudhia, J., Gill, D. O., Duda, D. M. B. M. G., Huang, X.-Y., Wang, W., and Powers, J. G. (2008). A description of the advanced research wrf version 3. *NCAR TECHNICAL NOTE*, 475:NCAR/TNÂŋ475+STR.

Smirnova, T. G., Benjamin, S. G., and Brown, J. M. (2000). Case study verification of ruc/maps fog and visibility forecasts. *Preprints, 9 th Conference on Aviation, Range, and Aerospace Meteorlogy, AMS, Orlando, FL, Sep. 2000*, 2.3:6.

WMO (2010). Guide to meteorological instruments and methods of observation. *Weather - Climate - Weather*, pages 1–176.

Yang, M. J., Braun, S. A., and Chen, D.-S. (2011). Water budget of typhoon nari (2001). *Mon. Wather Rev.*, 139:3809–3828.

A clt

A.1 code from LMDZ

This is the main core of the calculation of clt from LMDZ extracted from subroutine `newmicro.f90` and implemented in PyNCplot.

```
FUNCTION var_clt(clfra, dz)
(...)
  REAL(r_k), PARAMETER                                  :: ZEPSEC=1.0D-12
  REAL(r_k), PARAMETER                                  :: zero=0.d0
  REAL(r_k), PARAMETER                                  :: one=1.d0

(...)

    zclear = one
    zcloud = zero

    DO iz=1,dz
      zclear = zclear*(one-MAX(clfra(iz),zcloud))/(one-MIN(zcloud,1.-ZEPSEC))
      var_clt = one - zclear
      zcloud = clfra(iz)
    END DO
FUNCTION var_clt
```

B cllmh

B.1 cllmh: code from LMDZ

This is the main core of the calculation of cllmh from LMDZ extracted from subroutine `newmicro.f90` and implemented in PyNCplot.

```
  FUNCTION var_cllmh(clfra, p, dz)
(...)
! Low limit pressure for medium clouds [Pa]
  REAL(r_k), PARAMETER                                  :: prmhc = 44000.d0
! Low limit pressure for High clouds [Pa]
  REAL(r_k), PARAMETER                                  :: prmlc = 68000.d0

(...)
    REAL(r_k), DIMENSION(3)                             :: var_cllmh

(...)
    zclearl = one
    zcloudl = zero
    zclearm = one
    zcloudm = zero
    zclearh = one
    zcloudh = zero

    var_cllmh = one

    DO iz=1, dz
      IF (p(iz) < prmhc) THEN
        var_cllmh(3) = var_cllmh(3)*(one-MAX(clfra(iz),zcloudh))/(one-MIN(zcloudh,one-ZEPSEC))
```

```
      zcloudh = clfra(iz)
    ELSE IF ( (p(iz) >= prmhc) .AND. (p(iz) < prmlc) ) THEN
      var_cllmh(2) = var_cllmh(2)*(one-MAX(clfra(iz),zcloudm))/(one-MIN(zcloudm,one-ZEPSEC))
      zcloudm = clfra(iz)
    ELSE IF (p(iz) >= prmlc) THEN
      var_cllmh(1) = var_cllmh(1)*(one-MAX(clfra(iz),zcloudl))/(one-MIN(zcloudl,one-ZEPSEC))
      zcloudl = clfra(iz)
    ELSE
      PRINT *,'  ' // TRIM(fname) // ': This is weird, pressure:', p(iz), ' Pa fails out!!'
      PRINT *,'    from high, low cloud pressures:', prmhc, ' ,', prmlc,' Pa at z-level:', iz
      PRINT *,'    p_high > p:', prmhc,'> ',p(iz),' Pa'
      PRINT *,'    p_low > p >= p_high:', prmlc,'> ',p(iz),' >=', prmhc,' Pa'
      PRINT *,'    p_low >= p:', prmlc,'>= ',p(iz),' Pa'
      STOP
    END IF
  END DO

  var_cllmh = one - var_cllmh
```

C wsgsmax

C.1 afwa

```
  ! Calculate the max 10 m wind speed between output times
  ! -------------------------------------------------------
  ! UPDATE 20150112 - GAC
  ! Diagnose from model 10 m winds, and blend with 1 km AGL
  ! winds when precipitation rate is > 50 mm/hr to account
  ! for increased surface wind gust potential when precip
  ! is heavy and when winds aloft are strong.  Will use the
  ! higher of the surface and the blended winds. Blending
  ! is linear weighted between 50-150 mm/hr precip rates.
  ! --------------------------------------------------------
  DO j = jms, jme
    DO i = ims, ime
      wind_vel = uv_wind ( grid % u10(i,j) , grid % v10(i,j) )
      prate_mm_per_hr = ( grid % afwa_precip(i,j) / grid % dt ) * 3600.

      ! Is this an area of heavy precip?  Calculate 1km winds to blend down
      ! -------------------------------------------------------------------
      IF ( prate_mm_per_hr .GT. 50. ) THEN
        is_target_level=.false.
        DO k=kms,kme
          IF ( ( zagl(i,k,j) >= 1000. ) .and. &
               ( .NOT. is_target_level ) .and. &
               ( k .ne. kms ) ) THEN
            is_target_level = .true.
            u1km = u_phy(i,k-1,j) + (1000. - (zagl(i,k-1,j))) &
                   * ((u_phy(i,k,j) - u_phy(i,k-1,j))/(zagl(i,k,j)))
            v1km = v_phy(i,k-1,j) + (1000. - (zagl(i,k-1,j))) &
                   * ((v_phy(i,k,j) - v_phy(i,k-1,j))/(zagl(i,k,j)))
            EXIT ! We've found our level, break the loop
          ENDIF
```

```
        ENDDO

        ! Compute blended wind
        ! --------------------
        factor = MAX ( ( ( 150. - prate_mm_per_hr ) / 100. ), 0. )
        ublend = grid % u10(i,j) * factor + u1km * (1. - factor)
        vblend = grid % v10(i,j) * factor + v1km * (1. - factor)
        wind_blend = uv_wind ( ublend, vblend )

        ! Set the surface wind to the blended wind if higher
        ! --------------------------------------------------
        IF ( wind_blend .GT. wind_vel ) THEN
          wind_vel = wind_blend
        ENDIF
      ENDIF

      IF ( wind_vel .GT. grid % wspd10max(i,j) ) THEN
        grid % wspd10max(i,j) = wind_vel
      ENDIF
    ENDDO
  ENDDO
```

C.2 lmdz

```
      !-------gustiness calculation-------!
      IF (iflag_gusts==0) THEN
         gustiness(1:klon)=0
      ELSE IF (iflag_gusts==1) THEN
         gustiness(1:klon)=f_gust_bl*ale_bl(1:klon)+f_gust_wk*ale_wake(1:klon)
      ELSE IF (iflag_gusts==2) THEN
         gustiness(1:klon)=f_gust_bl*ale_bl_stat(1:klon)+f_gust_wk*ale_wake(1:klon)
         ! ELSE IF (iflag_gusts==2) THEN
         !    do i = 1, klon
         !       gustiness(i)=f_gust_bl*ale_bl(i)+sigma_wk(i)*f_gust_wk&
         !           *ale_wake(i) !! need to make sigma_wk accessible here
         !    enddo
         ! ELSE IF (iflag_gusts==3) THEN
         !    do i = 1, klon
         !       gustiness(i)=f_gust_bl*alp_bl(i)+f_gust_wk*alp_wake(i)
         !    enddo
      ENDIF
```

D psl

D.1 afwa

```
        MSLP = PSFC
!
!       COMPUTE LAYER TAU (VIRTUAL TEMP*RD/G).
        TVRT = TLEV1*(1.0+0.608*QLEV1)
        !TAU  = TVRT*RD*GI
!
!       COMPUTE TAU AT THE GROUND (Z=ZSFC) AND SEA LEVEL (Z=0)
!       ASSUMING A CONSTANT LAPSE RATE OF GAMMA=6.5DEG/KM.
```

```
         TVRSFC = TVRT + (ZLEV1 - ZSFC)*GAMMA
         TAUSFC = TVRSFC*RD*GI
         TVRSL  = TVRT + (ZLEV1 - ZSL)*GAMMA
         TAUSL  = TVRSL*RD*GI
!
!        IF NEED BE APPLY SHEULL CORRECTION.
         IF ((TAUSL.GT.TAUCR).AND.(TAUSFC.LE.TAUCR)) THEN
            TAUSL=TAUCR
         ELSEIF ((TAUSL.GT.TAUCR).AND.(TAUSFC.GT.TAUCR)) THEN
            TAUSL = TAUCR-CONST*(TAUSFC-TAUCR)**2
         ENDIF
!
!        COMPUTE MEAN TAU.
         TAUAVG = 0.5*(TAUSL+TAUSFC)
!
!        COMPUTE SEA LEVEL PRESSURE.
         MSLP = PSFC*EXP(ZSFC/TAUAVG)
```

D.2 lmdz

```
!>=======================================================================
!!
!! Auteur(s) I.Musat (LMD/CNRS) date: 20151106
!!
!! Objet: Calcul pression au niveau de la mer cf. Arpege-IFS
!! ctstar: calcule la temperature de l'air a la surface (tasfc) et
!!                 la temperature de l'air standard a la surface (tastd)
!! pppmer: calcule la slp a partir de tasfc, tastd, de la pression a la surface (pab1)
!!                 et du geopotentiel de la surface
!!=======================================================================
!! nlon--input-R-temperature au milieu de chaque couche (en K)
!! t--input-R-temperature au milieu de chaque couche (en K)
!! pab--input-R-pression pour chaque inter-couche (en Pa)
!! pal---input-R-pression pour le mileu de chaque couche (en Pa)
!! pphis---input-R-geopotentiel du sol (en m2/s2)
!! tasfc---output-R-temperature air au sol (en K)
!! tastd---output-R-temperature air 'standard' au sol (en K)
!! pmer---output-R-pression au niveau de la mer (en Pa)
!!=======================================================================
```

D.3 p_interp

```
N = 1.0

expon=287.04*.0065/9.81

! Fill in missing values
IF ( extrapolate == 0 ) RETURN       !! no extrapolation - we are out of here

! First find where about 400 hPa is located
kk = 0
find_kk : do k = 1, num_metgrid_levels
   kk = k
   if ( interp_levels(k) <= 40000. ) exit find_kk
end do find_kk
```

```
     data_out=0.
     do itt = 1, ito
       do j = 1, iy
       do i = 1, ix

!                We are below both the ground and the lowest data level.

!                First, find the model level that is closest to a "target" pressure
!                level, where the "target" pressure is delta-p less that the local
!                value of a horizontally smoothed surface pressure field.  We use
!                delta-p = 150 hPa here. A standard lapse rate temperature profile
!                passing through the temperature at this model level will be used
!                to define the temperature profile below ground.  This is similar
!                to the Benjamin and Miller (1990) method, using
!                700 hPa everywhere for the "target" pressure.

!         ptarget = (psfc(i,j,itt)*.01) - 150.
        ptarget = 700.
        dpmin=1.e4
        kupper = 0
        loop_kIN : do kin=iz,1,-1
          kupper = kin
          dp=abs( (pres_field(i,j,kin,itt)*.01) - ptarget )
          if (dp.gt.dpmin) exit loop_kIN
            dpmin=min(dpmin,dp)
          enddo loop_kIN
        ptarget=ptarget*100.
!         pbot=max(pres_field(i,j,1,itt),psfc(i,j,itt))
!         zbot=0.

!         tbotextrap=tk(i,j,kupper,itt)*(pbot/pres_field(i,j,kupper,itt))**expon
!         tvbotextrap=virtual(tbotextrap,qv(i,j,1,itt))

!         data_out(i,j,itt,1) = (zbot+tvbotextrap/.0065*(1.-(interp_levels(1)/pbot)**expon))
        tbotextrap=tk(i,j,kupper,itt)*(psfc(i,j,itt)/ptarget)**expon
        tvbotextrap=virtual(tbotextrap,qv(i,j,kupper,itt))
        data_out(i,j,itt,1) = psfc(i,j,itt)*((tvbotextrap+0.0065*ter(i,j))/tvbotextrap)**(1/expon)
!!        IF (i==ix/2 .AND. j==iy/2 ) THEN
!        IF (ter(i,j) > 2500.) THEN
!!          PRINT *,itt,' ptarget',ptarget,'kupper:',kupper
!!          PRINT *,'tk:',tk(i,j,kupper,itt),'psfc:',psfc(i,j,itt)
!!          PRINT *,'tbot:',tbotextrap,'tvbot:',tvbotextrap,'ter:',ter(i,j)
!!          PRINT *,'qv:',qv(i,j,kupper,itt),'mslp:',data_out(i,j,itt,1)
!!        ENDIF

       enddo
       enddo
     enddo
```

E cape

E.1 afwa

```
!~ Thermo / dynamical constants
! ----------------------------
REAL                    :: Rd          !~ Dry gas constant
   PARAMETER ( Rd = 287.058 )          !~ J deg^-1 kg^-1
REAL                    :: Cp          !~ Specific heat constant pressure
   PARAMETER ( Cp = 1004.67 )          !~ J deg^-1 kg^-1
REAL                    :: g           !~ Acceleration due to gravity
   PARAMETER ( g  = 9.80665 )          !~ m s^-2
REAL                    :: RUNDEF
   PARAMETER ( RUNDEF = -9.999E30 )

!~ Initialize variables
! --------------------
ostat  = 0
CAPE   = REAL ( 0 )
CIN    = REAL ( 0 )
ZLFC   = RUNDEF
PLFC   = RUNDEF

!~ Look for submitted parcel definition
!~ 1 = Most unstable
!~ 2 = Mean layer
!~ 3 = Surface based
! -------------------------------------
IF ( parcel > 3 .or. parcel < 1 ) THEN
   prcl = 1
ELSE
   prcl =  parcel
END IF

!~ Initalize our parcel to be (sort of) surface based.  Because of
!~ issues we've been observing in the WRF model, specifically with
!~ excessive surface moisture values at the surface, using a true
!~ surface based parcel is resulting a more unstable environment
!~ than is actually occuring.  To address this, our surface parcel
!~ is now going to be defined as the parcel between 25-50 hPa
!~ above the surface. UPDATE - now that this routine is in WRF,
!~ going to trust surface info. GAC 20140415
! ----------------------------------------------------------------

!~ Compute mixing ratio values for the layer
! ----------------------------------------
DO k = sfc, nz
  ws   ( k )   = SaturationMixingRatio ( tK(k), p(k) )
  w    ( k )   = ( rh(k)/100.0 ) * ws ( k )
END DO

srclev      = sfc
srctK       = tK    ( sfc )
srcrh       = rh    ( sfc )
srcp        = p     ( sfc )
```

```
srcws       = ws    ( sfc )
srcw        = w     ( sfc )
srctheta    = Theta ( tK(sfc), p(sfc)/100.0 )

!~ Compute the profile mixing ratio.  If the parcel is the MU parcel,
!~ define our parcel to be the most unstable parcel within the lowest
!~ 180 mb.
!  -------------------------------------------------------------------
mlev = sfc + 1
DO k = sfc + 1, nz

   !~ Identify the last layer within 100 hPa of the surface
   !  -----------------------------------------------------
   pdiff = ( p (sfc) - p (k) ) / REAL ( 100 )
   IF ( pdiff <= REAL (100) ) mlev = k

   !~ If we've made it past the lowest 180 hPa, exit the loop
   !  -------------------------------------------------------
   IF ( pdiff >= REAL (180) ) EXIT

   IF ( prcl == 1 ) THEN
      !IF ( (p(k) > 70000.0) .and. (w(k) > srcw) ) THEN
      IF ( (w(k) > srcw) ) THEN
         srctheta = Theta ( tK(k), p(k)/100.0 )
         srcw = w ( k )
         srclev  = k
         srctK   = tK ( k )
         srcrh   = rh ( k )
         srcp    = p  ( k )
      END IF
   END IF

END DO

!~ If we want the mean layer parcel, compute the mean values in the
!~ lowest 100 hPa.
!  ----------------------------------------------------------------
lyrcnt =  mlev - sfc + 1
IF ( prcl == 2 ) THEN

   srclev   = sfc
   srctK    = SUM ( tK (sfc:mlev) ) / REAL ( lyrcnt )
   srcw     = SUM ( w  (sfc:mlev) ) / REAL ( lyrcnt )
   srcrh    = SUM ( rh (sfc:mlev) ) / REAL ( lyrcnt )
   srcp     = SUM ( p  (sfc:mlev) ) / REAL ( lyrcnt )
   srctheta = Theta ( srctK, srcp/100. )

END IF

srcthetaeK = Thetae ( srctK, srcp/100.0, srcrh, srcw )

!~ Calculate temperature and pressure of the LCL
!  --------------------------------------------
tlclK = TLCL ( tK(srclev), rh(srclev) )
```

```
plcl  = p(srclev) * ( (tlclK/tK(srclev))**(Cp/Rd) )

!~ Now lift the parcel
!  -------------------

buoy  = REAL ( 0 )
pw    = srcw
wflag = .false.
DO k  = srclev, nz
   IF ( p (k) <= plcl ) THEN

      !~ The first level after we pass the LCL, we're still going to
      !~ lift the parcel dry adiabatically, as we haven't added the
      !~ the required code to switch between the dry adiabatic and moist
      !~ adiabatic cooling.  Since the dry version results in a greater
      !~ temperature loss, doing that for the first step so we don't over
      !~ guesstimate the instability.
      !  ----------------------------------------------------------------

      IF ( wflag ) THEN
         flag  = .false.

         !~ Above the LCL, our parcel is now undergoing moist adiabatic
         !~ cooling.  Because of the latent heating being undergone as
         !~ the parcel rises above the LFC, must iterative solve for the
         !~ parcel temperature using equivalant potential temperature,
         !~ which is conserved during both dry adiabatic and
         !~ pseudoadiabatic displacements.
         !  --------------------------------------------------------------
         ptK   = The2T ( srcthetaeK, p(k), flag )

         !~ Calculate the parcel mixing ratio, which is now changing
         !~ as we condense moisture out of the parcel, and is equivalent
         !~ to the saturation mixing ratio, since we are, in theory, at
         !~ saturation.
         !  ------------------------------------------------------------
         pw = SaturationMixingRatio ( ptK, p(k) )

         !~ Now we can calculate the virtual temperature of the parcel
         !~ and the surrounding environment to assess the buoyancy.
         !  ----------------------------------------------------------
         ptvK  = VirtualTemperature ( ptK, pw )
         tvK   = VirtualTemperature ( tK (k), w (k) )

         !~ Modification to account for water loading
         !  -----------------------------------------
         freeze = 0.033 * ( 263.15 - pTvK )
         IF ( freeze > 1.0 ) freeze = 1.0
         IF ( freeze < 0.0 ) freeze = 0.0

         !~ Approximate how much of the water vapor has condensed out
         !~ of the parcel at this level
         !  --------------------------------------------------------
         freeze = freeze * 333700.0 * ( srcw - pw ) / 1005.7
```

```
      pTvK = pTvK - pTvK * ( srcw - pw ) + freeze
      dTvK ( k ) = ptvK - tvK
      buoy ( k ) = g * ( dTvK ( k ) / tvK )

   ELSE

      !~ Since the theta remains constant whilst undergoing dry
      !~ adiabatic processes, can back out the parcel temperature
      !~ from potential temperature below the LCL
      !  -------------------------------------------------------
      ptK   = srctheta / ( 100000.0/p(k) )**(Rd/Cp)

      !~ Grab the parcel virtual temperture, can use the source
      !~ mixing ratio since we are undergoing dry adiabatic cooling
      !  ----------------------------------------------------------
      ptvK  = VirtualTemperature ( ptK, srcw )

      !~ Virtual temperature of the environment
      !  --------------------------------------
      tvK   = VirtualTemperature ( tK (k), w (k) )

      !~ Buoyancy at this level
      !  ---------------------
      dTvK ( k ) = ptvK - tvK
      buoy ( k ) = g * ( dtvK ( k ) / tvK )

      wflag = .true.

   END IF

ELSE

   !~ Since the theta remains constant whilst undergoing dry
   !~ adiabatic processes, can back out the parcel temperature
   !~ from potential temperature below the LCL
   !  -------------------------------------------------------
   ptK   = srctheta / ( 100000.0/p(k) )**(Rd/Cp)

   !~ Grab the parcel virtual temperture, can use the source
   !~ mixing ratio since we are undergoing dry adiabatic cooling
   !  ----------------------------------------------------------
   ptvK  = VirtualTemperature ( ptK, srcw )

   !~ Virtual temperature of the environment
   !  --------------------------------------
   tvK   = VirtualTemperature ( tK (k), w (k) )

   !~ Buoyancy at this level
   !  ---------------------
   dTvK ( k ) = ptvK - tvK
   buoy ( k ) = g * ( dtvK ( k ) / tvK )

END IF
```

```
      !~ Chirp
      !  -----
!          WRITE ( *,'(I15,6F15.3)' )k,p(k)/100.,ptK,pw*1000.,ptvK,tvK,buoy(k)

   END DO

   !~ Add up the buoyancies, find the LFC
   !  ----------------------------------
   flag   = .false.
   lfclev = -1
   nbuoy  = REAL ( 0 )
   pbuoy = REAL ( 0 )
   DO k = sfc + 1, nz
      IF ( tK (k) < 253.15 ) EXIT
      CAPE = CAPE + MAX ( buoy (k), 0.0 ) * ( hgt (k) - hgt (k-1) )
      CIN  = CIN  + MIN ( buoy (k), 0.0 ) * ( hgt (k) - hgt (k-1) )

      !~ If we've already passed the LFC
      !  -------------------------------
      IF ( flag .and. buoy (k) > REAL (0) ) THEN
         pbuoy = pbuoy + buoy (k)
      END IF

      !~ We are buoyant now - passed the LFC
      !  -----------------------------------
      IF ( .not. flag .and. buoy (k) > REAL (0) .and. p (k) < plcl ) THEN
         flag = .true.
         pbuoy = pbuoy + buoy (k)
         lfclev = k
      END IF

      !~ If we think we've passed the LFC, but encounter a negative layer
      !~ start adding it up.
      !  ----------------------------------------------------------------
      IF ( flag .and. buoy (k) < REAL (0) ) THEN
         nbuoy = nbuoy + buoy (k)

         !~ If the accumulated negative buoyancy is greater than the
         !~ positive buoyancy, then we are capped off.  Got to go higher
         !~ to find the LFC. Reset positive and negative buoyancy summations
         !  ----------------------------------------------------------------
         IF ( ABS (nbuoy) > pbuoy ) THEN
            flag   = .false.
            nbuoy  = REAL ( 0 )
            pbuoy  = REAL ( 0 )
            lfclev = -1
         END IF
      END IF

   END DO

   !~ Calculate lifted index by interpolating difference between
   !~ parcel and ambient Tv to 500mb.
```

```
! ----------------------------------------------------------
DO k = sfc + 1, nz

   pm = 50000.
   pu = p ( k )
   pd = p ( k - 1 )

   !~ If we're already above 500mb just set lifted index to 0.
   !~ --------------------------------------------------------
   IF ( pd .le. pm ) THEN
      lidx = 0.
      EXIT

   ELSEIF ( pu .le. pm .and. pd .gt. pm) THEN

      !~ Found trapping pressure: up, middle, down.
      !~ We are doing first order interpolation.
      !  ------------------------------------------
      lidxu = -dTvK ( k ) * ( pu / 100000. ) ** (Rd/Cp)
      lidxd = -dTvK ( k-1 ) * ( pd / 100000. ) ** (Rd/Cp)
      lidx = ( lidxu * (pm-pd) + lidxd * (pu-pm) ) / (pu-pd)
      EXIT

   ENDIF

END DO

!~ Assuming the the LFC is at a pressure level for now
! ---------------------------------------------------
IF ( lfclev > 0 ) THEN
   PLFC = p   ( lfclev )
   ZLFC = hgt ( lfclev )
END IF

IF ( PLFC /= PLFC .OR. PLFC < REAL (0) ) THEN
   PLFC = REAL ( -1 )
   ZLFC = REAL ( -1 )
END IF

IF ( CAPE /= CAPE ) cape = REAL ( 0 )

IF ( CIN  /= CIN  ) cin  = REAL ( 0 )
```

E.2 lmdz

Not simple to get the right code, computation is split in different subroutines being the main driving one `phylmd/cva_driver.F90`

F cin

F.1 afwa

Similar as in with CAPE (E.1)

F.2 lmdz

Table 1: Equivalencies among CORDEX CF variables' name and how are they defined in WRF module (from `registry.cordex`)

CF name	**WRF name**	**description**	**units**
2D			
lon	CDXLON	LONGITUDE	degrees_east
lat	CDXLAT	LATITUDE	degrees_north
capev	CDXCAPE	CONVECTIVE AVAILABLE POTENTIAL ENERGY	Jkg-1
CAPEMIN	capemin	minimum convective available potential energy	Jkg-1
CAPEMAX	capemax	maximum convective available potential energy	Jkg-1
CAPEMEAN	capemean	mean convective available potential energy	Jkg-1
cin	CIN	CONVECTIVE INHIBITION	Jkg-1
CINMIN	cinmin	minimum convective inhibition	Jkg-1
CINMAX	cinmax	maximum convective inhibition	Jkg-1
CINMEAN	cinmean	mean convective inhibition	Jkg-1
cltmean	CLTMEAN	MEAN TOTAL CLOUDINESS IN CORDEX OUTPUT	1
cllmean	CLLMEAN	MEAN LOW-LEVEL CLOUDINESS (p >= 68000 Pa) IN CORDEX OUTPUT	1
clmmean	CLMMEAN	MEAN MID-LEVEL CLOUDINESS (44000 <= p < 68000 Pa) IN CORDEX OUTPUT	1
clhmean	CLHMEAN	MEAN HIGH-LEVEL CLOUDINESS (p < 44000 Pa) IN CORDEX OUTPUT	1
mrso	MRSO	TOTAL SOIL CONTENT	kgm-2
prw	PRW	WATER VAPOR PATH	kgm-2
psl	PSL	SEA LEVEL PRESSURE	Pa
clwvi	CLWVI	LIQUID WATER PATH	kgm-2
clivi	CLIVI	ICE WATER PATH	kgm-2
clgvi	CLGVI	GRAUPEL WATER PATH	kgm-2
clhvi	CLHVI	HAIL WATER PATH	kgm-2
hurs	HURS	2M RELATIVE HUMIDITY	1
huss	HUSS	2M SPECIFIC HUMIDITY	1
ps	CDXPS	surface pressure	Pa
ts	CDXTS	skin temperature	K

G WRF CORDEX variables definition

Here is provided in different tables (from 1 to 3) of equivalencies among CORDEX CF-names and their definitions in WRF (directly from `Registry/registry.cordex`)

Here is provided the table (4) with the definitions of the water-budget additional variables in WRF (directly from `Registry/registry.cordex`)

Table 2: Continuation of table 1

CF name	WRF name	description	units
lfcp	LFCP	PRESSURE LEVEL FREE CONVECTION	Pa
LFCPMIN	lfcpmin	minimum pressure level free convection	Pa
LFCPMAX	lfcpmax	maximum pressure level free convection	Pa
LFCPMEAN	lfcpmean	mean pressure level free convection	Pa
lfcz	LFCZ	HEIGHT LEVEL FREE CONVECTION	m
LFCZMIN	lfczmin	minimum height level free convection	m
LFCZMAX	lfczmax	maximum height level free convection	m
LFCZMEAN	lfczmean	mean height level free convection	m
li	LI	LIFTED INDEX	1
LIMEAN	limean	mean lifted index	1
LIMIN	limin	minimum lifted index	1
LIMAX	limax	maximum lifted index	1
slw	SLW	TOTAL SOIL LIQUID WATER CONTENT	kgm-2
uas	UAS	10M EASTWARD WIND SPEED	ms-1
vas	VAS	10M NORTHWARD WIND SPEED	ms-1
WSS	wss	10M WIND SPEED	ms-1
wsgsmax	WSGSMAX	Maximum near-surface wind speed of gust	ms-1
usgsmax	USGSMAX	Eastward maximum near-surface wind speed of gust	ms-1
vsgsmax	VSGSMAX	Northward maximum near-surface wind speed of gust	ms-1
wsgspercen	WSGSPERCEN	Percentage of time steps where grid point got wind gust	%
totwsgsmax	TOTWSGSMAX	Total (TKE + h. pr) Maximum near-surface wind speed of gust	ms-1
totugsmax	TOTUGSMAX	Total Eastward maximum near-surface wind speed of gust	ms-1
totvgsmax	TOTVGSMAX	Total Northward maximum near-surface wind speed of gust	ms-1
totwsgspercen	TOTWSGSPERCEN	Percentage of time steps where grid point got total wind gust	%
zmlagen	ZMLAGEN	Generic boundary layer height theta(zmlagen) > min(theta[mix. layer]) + 1.5K	m
wsz100max	WSZ100MAX	Maximum 100m nwind speed	ms-1
UZ100	uz100	Eastward 100 m wind speed	ms-1
VZ100	vz100	Northward 100 m wind speed	ms-1
WSZ100MAX	wsz100max	Maximum 100m nwind speed	ms-1
UZ100MAX	uz100max	Eastward maximum 100 m wind speed	ms-1
VZ100MAX	vz100max	Northward maximum 100 m wind speed	ms-1
SUND	sund	SUNSHINE LENGTH (ac. time SWDOWN > 120. Wm-2)	second
TAUU	tauu	northward downward wind stress at 10 m	m2s-2
TAUV	tauv	eastward downward wind stress at 10 m	m2s-2
TAUUGEN	tauugen	generic northward downward wind stress at 10 m	m2s-2
TAUVGEN	tauvgen	generic eastward downward wind stress at 10 m	m2s-2
RSDS	rsds	mean surface Downwelling Shortwave Radiation	Wm-2
RLDS	rlds	mean surface Downwelling Longwave Radiation	Wm-2
HFLS	hfls	mean surface Upward Latent Heat Flux	Wm-2
HFSS	hfss	mean surface Upward Sensible Heat Flux	Wm-2
RSUS	rsus	mean surface Upwelling Shortwave Radiation	Wm-2
RLUS	rlus	mean surface Upwelling Longwave Radiation	Wm-2
RSUSGEN	rsusgen	mean generic surface Upwelling Shortwave Radiation	Wm-2
RLUSGEN	rlusgen	mean generic surface Upwelling Longwave Radiation	Wm-2
EVSPSBL	evspsbl	mean evaporation	kgm-2s-1
EVSPSBLPOT	evspsblpot	mean potential evapotranspiration	kgm-2s-1
CDCDX	cdcdx	drag coefficient	-
CDGEN	cdgen	generic drag coefficient	-

Table 3: Continuation of table 1

CF name	WRF name	description	units
SNC	snc	mean snow area fraction	%
SND	snd	mean snow depth	m
PMRROS	pmrros	previous accumulated surface Runoff	kgm-2
PMRRO	pmrro	previous accumulated total Runoff	kgm-2
MRROS	mrros	mean surface Runoff	kgm-2s-1
MRRO	mrro	mean total Runoff	kgm-2s-1
MRSOL	mrsol	mean total water content of soil layer	kgm-2
pr	pr	precipitation flux	kgm-2s-1
prl	prl	large scale precipitation flux	kgm-2s-1
prc	prc	convective precipitation flux	kgm-2s-1
prsh	prsh	shallow-cumulus precipitation flux	kgm-2s-1
prsn	prsn	solid precipitation flux	kgm-2s-1
snw	snw	accumulated snow precipitation	kgm-2
rsdt	rsdt	mean top of the atmosphere (TOA) incident shortwave radiation kgm-2	
rsut	rsut	mean TOA outgoing shortwave radiation	kgm-2
rlut	rlut	mean TOA outgoing Longwave radiation	kgm-2
tfog	tfog	time of presence of fog	seconds
fogvisbltymin	fogvisbltymin	minimum of visibility inside fog	km
fogvisbltymax	fogvisbltymax	maximum of visibility inside fog	km
fogvisbltymean	fogvisbltymean	mean of visibility inside fog	km
tdsmin	tdsmin	minimum surface dew point temperature	K
tdsmax	tdsmax	maximum surface dew point temperature	K
tdsmean	tdsmean	mean surface dew point temperature	K
3D			
HUR	HUR	AIR RELATIVE HUMIDITY	1
HUS	HUS	AIR SPECIFIC HUMIDITY	1
ZG	ZG	AIR GEOPOTENTIAL HEIGHT	m
PRESS	PRESS	AIR PRSSURE	Pa
TA	TA	AIR TEMPERATURE	K
UA	UA	AIR EASTWARD WIND SPEED	ms-1
VA	VA	AIR NORTHWARD WIND SPEED	ms-1
WS	ws	AIR WIND SPEED	ms-1

Table 4: Water-budget equivalencies and their name and how are they defined in WRF module (from `registry.cordex`)

name	WRF name	description	units
$\mathcal{Q}_h ac$	WBACDIABH	Water Budget column integrated and time accumulation of diabatic heating from Micro-Physics	K
$\partial_t qvac$	WBACPW	Water Budget column integrated and time accumulated for water vapor content	mm
$\partial_t qcac$	WBACPWC	Water Budget col. int. & time accumulated for cloud content	mm
$\partial_t qrac$	WBACPWR	Water Budget col. int. & time accumulated for rain content	mm
$\partial_t qsac$	WBACPWS	Water Budget col. int. & time accumulated for snow content	mm
$\partial_t qiac$	WBACPWI	Water Budget col. int. & time accumulated for ice content	mm
$\partial_t qhac$	WBACPWH	Water Budget col. int. & time accumulated for hail content	mm
$\partial_t qgac$	WBACPWG	Water Budget col. int. & time accumulated for graupel content	mm
$adv_h qvac$	WBACF	W.B. c-int. acc. hor. convergence of water vapour (+, conv.; -, div.)	mm
$adv_h qcac$	WBACFC	W.B. c-int. acc. hor. convergence of cloud (+, conv.; -, div.)	mm
$adv_h qrac$	WBACFR	W.B. c-int. acc. hor. convergence of rain (+, conv.; -, div.)	mm
$adv_h qsac$	WBACFS	W.B. c-int. acc. hor. convergence of snow (+, conv.; -, div.)	mm
$adv_h qiac$	WBACFI	W.B. c-int. acc. hor. convergence of ice (+, conv.; -, div.)	mm
$adv_h qhac$	WBACFH	W.B. c-int. acc. hor. convergence of hail (+, conv.; -, div.)	mm
$adv_h qgac$	WBACFG	W.B. c-int. acc. hor. convergence of graupel (+, conv.; -, div.)	mm
$adv_z qvac$	WBACZ	W.B. c-int. acc. ver. convergence of water vapour (+, conv.; -, div.), always 0	mm
$adv_z qcac$	WBACZC	W.B. c-int. acc. ver. convergence of cloud (+, conv.; -, div.)	mm
$adv_z qrac$	WBACZR	W.B. c-int. acc. ver. convergence of rain (+, conv.; -, div.)	mm
$adv_z qsac$	WBACZS	W.B. c-int. acc. ver. convergence of snow (+, conv.; -, div.)	mm
$adv_z qiac$	WBACZI	W.B. c-int. acc. ver. convergence of ice (+, conv.; -, div.)	mm
$adv_z qhac$	WBACZH	W.B. c-int. acc. ver. convergence of hail (+, conv.; -, div.)	mm
$adv_z qgac$	WBACZG	W.B. c-int. acc. ver. convergence of graupel (+, conv.; -, div.)	mm
Low-mid-level			
$\mathcal{Q}_h^l ac$	WBACDIABHL	W.B. low level acc. of diabatic heating from MP	K
$\mathcal{Q}_h^m ac$	WBACDIABHM	W.B. mid-level acc. of diabatic heating from MP	K
$\mathcal{Q}_h^m ac$	WBACDIABHH	W.B. high-level acc. of diabatic heating from MP	K
$\partial_t^l qvac$	WBACPWLV	W.B. low level (p >= 68000 Pa) acc. for QV	mm
$\partial_t^m qvac$	WBACPWMV	W.B. mid level (44000 Pa <= p < 68000 Pa) acc. for QV	mm
$\partial_t^h qvac$	WBACPWHV	W.B. high level (p < 44000 Pa) acc. for QV	mm
$adv_h^l qvac$	WBACFLV	W.B. low-lev. acc. hor. convergence of QV	mm
$adv_h^m qvac$	WBACFMV	W.B. mid-lev. acc. hor. convergence of QV	mm
$adv_h^h qvac$	WBACFHV	W.B. high-lev. acc. hor. convergence of QV	mm
$adv_z^l qvac$	WBACZLV	W.B. low level acc. ver. convergence of QV	mm

Table 5: Continuation of table 4

$adv_z^m qvac$	WBACZMV	W.B. mid level acc. ver. convergence of QV	mm
$adv_z^h qvac$	WBACZHV	W.B. high level acc. ver. convergence of QV	mm
$\partial_t^l qcac$	WBACPWLC	W.B. low level (p >= 68000 Pa) acc. for QC	mm
$\partial_t^m qcac$	WBACPWMC	W.B. mid level (44000 Pa <= p < 68000 Pa) acc. for QC	mm
$\partial_t^h qcac$	WBACPWHC	W.B. high level (p < 44000 Pa) acc. for QC	mm
$adv_h^l qcac$	WBACFLC	W.B. low-lev. acc. hor. convergence of QC	mm
$adv_h^m qcac$	WBACFMC	W.B. mid-lev. acc. hor. convergence of QC	mm
$adv_h^h qcac$	WBACFHC	W.B. high-lev. acc. hor. convergence of QC	mm
$adv_z^l qcac$	WBACZLC	W.B. low level acc. ver. convergence of QC	mm
$adv_z^m qcac$	WBACZMC	W.B. mid level acc. ver. convergence of QC	mm
$adv_z^h qcac$	WBACZHC	W.B. high level acc. ver. convergence of QC	mm
$\partial_t^l qrac$	WBACPWLR	W.B. low level (p >= 68000 Pa) acc. for QR	mm
$\partial_t^m qrac$	WBACPWMR	W.B. mid level (44000 Pa <= p < 68000 Pa) acc. for QR	mm
$\partial_t^h qrac$	WBACPWHR	W.B. high level (p < 44000 Pa) acc. for QR	mm
$adv_h^l qrac$	WBACFLR	W.B. low-lev. acc. hor. convergence of QR	mm
$adv_h^m qrac$	WBACFMR	W.B. mid-lev. acc. hor. convergence of QR	mm
$adv_h^h qrac$	WBACFHR	W.B. high-lev. acc. hor. convergence of QR	mm
$adv_z^l qrac$	WBACZLR	W.B. low level acc. ver. convergence of QR	mm
$adv_z^m qrac$	WBACZMR	W.B. mid level acc. ver. convergence of QR	mm
$adv_z^h qrac$	WBACZHR	W.B. high level acc. ver. convergence of QR	mm
$\partial_t^l qsac$	WBACPWLS	W.B. low level (p >= 68000 Pa) acc. for QS	mm
$\partial_t^m qsac$	WBACPWMS	W.B. mid level (44000 Pa <= p < 68000 Pa) acc. for QS	mm
$\partial_t^h qsac$	WBACPWHS	W.B. high level (p < 44000 Pa) acc. for QS	mm
$adv_h^l qsac$	WBACFLS	W.B. low-lev. acc. hor. convergence of QS	mm
$adv_h^m qsac$	WBACFMS	W.B. mid-lev. acc. hor. convergence of QS	mm
$adv_h^h qsac$	WBACFHS	W.B. high-lev. acc. hor. convergence of QS	mm
$adv_z^l qsac$	WBACZLS	W.B. low level acc. ver. convergence of QS	mm
$adv_z^m qsac$	WBACZMS	W.B. mid level acc. ver. convergence of QS	mm
$adv_z^h qsac$	WBACZHS	W.B. high level acc. ver. convergence of QS	mm
$\partial_t^l qiac$	WBACPWLI	W.B. low level (p >= 68000 Pa) acc. for QI	mm
$\partial_t^m qiac$	WBACPWMI	W.B. mid level (44000 Pa <= p < 68000 Pa) acc. for QI	mm
$\partial_t^h qiac$	WBACPWHI	W.B. high level (p < 44000 Pa) acc. for QI	mm
$adv_h^l qiac$	WBACFLI	W.B. low-lev. acc. hor. convergence of QI	mm
$adv_h^m qiac$	WBACFMI	W.B. mid-lev. acc. hor. convergence of QI	mm
$adv_h^h qiac$	WBACFHI	W.B. high-lev. acc. hor. convergence of QI	mm
$adv_z^l qiac$	WBACZLI	W.B. low level acc. ver. convergence of QI	mm
$adv_z^m qiac$	WBACZMI	W.B. mid level acc. ver. convergence of QI	mm
$adv_z^h qiac$	WBACZHI	W.B. high level acc. ver. convergence of QI	mm
$\partial_t^l qgac$	WBACPWLG	W.B. low level (p >= 68000 Pa) acc. for QG	mm
$\partial_t^m qgac$	WBACPWMG	W.B. mid level (44000 Pa <= p < 68000 Pa) acc. for QG	mm
$\partial_t^h qgac$	WBACPWHG	W.B. high level (p < 44000 Pa) acc. for QG	mm
$adv_h^l qgac$	WBACFLG	W.B. low-lev. acc. hor. convergence of QG	mm
$adv_h^m qgac$	WBACFMG	W.B. mid-lev. acc. hor. convergence of QG	mm
$adv_h^h qgac$	WBACFHG	W.B. high-lev. acc. hor. convergence of QG	mm
$adv_z^l qgac$	WBACZLG	W.B. low level acc. ver. convergence of QG	mm
$adv_z^m qgac$	WBACZMG	W.B. mid level acc. ver. convergence of QG	mm
$adv_z^h qgac$	WBACZHG	W.B. high level acc. ver. convergence of QG	mm

Table 6: Continuation of table 4

$\partial_t^l qhac$	WBACPWLH	W.B. low level (p >= 68000 Pa) acc. for QH	mm
$\partial_t^m qhac$	WBACPWMH	W.B. mid level (44000 Pa <= p < 68000 Pa) acc. for QH	mm
$\partial_t^h qhac$	WBACPWHH	W.B. high level (p < 44000 Pa) acc. for QH	mm
$adv_h^l qhac$	WBACFLH	W.B. low-lev. acc. hor. convergence of QH	mm
$adv_h^m qhac$	WBACFMH	W.B. mid-lev. acc. hor. convergence of QH	mm
$adv_h^h qhac$	WBACFHH	W.B. high-lev. acc. hor. convergence of QH	mm
$adv_z^l qhac$	WBACZLH	W.B. low level acc. ver. convergence of QH	mm
$adv_z^m qhac$	WBACZMH	W.B. mid level acc. ver. convergence of QH	mm
$adv_z^h qhac$	WBACZHH	W.B. high level acc. ver. convergence of QH	mm