MUSC practical guidance

david.nemec[a]chmi.cz
last update: 2026-04-23

Contents

1 A quick introduction to the MUSC world

MUSC (Modèle Unifié, Simple Colonne) is a single-column version of our model, activated by setting `NCONF=1`, and requiring only minor changes of the namelist and script. Such a configuration can be useful mainly for developing physical parameterizations because one avoids 3D feedback. Several idealized cases have been prepared consistently across ACCORD Canonical System Configurations (CSCs), allowing their intercomparison and validation against LES or observations for some of them. MUSC is also very cheap to run, and one can analyze more variables than from the 3D output.

This document is focused on the ALARO CSC CY46 for the simplicity of the description of namelist changes. However, the MUSC parts are identical to all recent versions of the model code of all CSCs (CY38 and newer for sure). The differences come from dynamics and physics parts as some switches differ accross CSCs or are occasionally moved to different namelist blocks. The reader should consult the dynamics and physics documentations for more details for other cycles.

From the technical/practical point of view, the only differences between MUSC and the 3D configuration are the case-specific namelist for MUSC and the initial file (`ICMSHALADINIT`; containing forcing,

Table 1: MUSC-specific routines.

variable	description
arpifs/adiab/cp_forcing.F90	application of upper-air forcing
arpifs/phys_dmn/surf_ideal_flux.F90	surface forcing
arpifs/phys_dmn/writemusc.F90	writes many MUSC fields
arpifs/module/yomlsforc.F90	module with definitions of forcing parameters
arpifs/setup/sulsforc.F90	initialization of forcing parameters

which is always on full levels), so one can use their operational executable with subtly modified script for 3D integrations.

The case-specific upper-air forcing is stored in the ICMSHALADINIT file and assigned by the case-specific namelist. The surface forcing can be prescribed with or without SURFEX. Based on that, it is either stored in the ICMSHALADINIT file or in the ICMSHALADINIT.sfx file. If only surface temperature is known as surface forcing, then SURFEX must be used; otherwise, fluxes are wrong. Latent heat and sensible heat fluxes are prescribed instead if SURFEX is not used. All cases presented here do not use SURFEX.

MUSC has two different output formats. In addition to ICMSH files, it also produces LFA files, which can be read with DDH tools. The LFA files are written after every time step and contain more variables than the ICMSH files.

MUSC can be run without new tools once the initfile and namelist are ready. Program ascii2fa can be used to create new initfiles. One can use their preferred scripting/programming language from the ASCII output extracted with DDH tools for visualization. A database of cases is prepared and described in Section 4.

As an alternative, Environment for MUSC Simulations (EMS) is being developed by Romain Roehrig (https://github.com/romainroehrig/EMS), which runs "everything on one click," and is written in Python. However, it currently does not seem that it will be commonly used in ACCORD, so it will not be further mentioned in this document.

There are specific MUSC routines in the code (listed in Table 1). Generally, MUSC runs the full physics by default. However, forcing must be applied, and fields must be written.

2 Running MUSC

2.1 General overview

To run MUSC, one needs:

1. A version of the ALADIN model executable; it can be the operationally used one.

2. Initial and forcing file (2 in 1). It is ordinary ICMSHALADINIT. In addition to typical fields known from the 3D model, the upper-air forcing is SXXXFORCXXXX (X are numbers), and the surface forcing is SURFFORCXXXX.

3. Namelist for the specific case. Several variables might be forced (even at once), so the forced variables and their order must be specified in the namelist. Details are described in Section 3.

One then updates the script for the 3D integration job.

2.2 Changes to the 3D script

One must use one thread on one processor with NPROMA=-4. For Slurm, use #SBATCH -N 1 in the header. If the ALADIN executable is executed as:

`mpiexec.hydra -np ${NPROC} -perhost ${PROCPERNODE} ./ALADIN > lola`,

set `NPROC=1` and `PROCPERNODE=1`.

Otherwise, instead of the 3D `ICMSHXXXXINIT`, one must use the initfile for the selected case and the case-specific namelist. These things need to be changed in the script, but every machine has its specifics, so more changes might be needed.

3 MUSC namelist options

3.1 General namelist options

In `&NAMCTO`, `LSFORC=.TRUE.` must be set to read uppper-air forcing or nudging, and it is necessary for producing LFA output because `&NAMLSFORC` is read by routine `setup/sulsforc.F90`, as the LFA output is set in this namelist and the routine is not called otherwise. MUSC can be run without forcing or nudging while producing LFA files by not assigning any forcing or nudging in `&NAMLSFORC` (see Subsection 3.2). Surface forcing is set with `LSFORCS=.TRUE.` without SURFEX. If SURFEX is used, then `LSFORCS=.FALSE.` and `LMSE=.TRUE.` in `&NAMARPHY`. Furthermore, `NFPOS=0` to switch off FullPos and `LNHEE=.FALSE.` to keep the model hydrostatic.

In `&NAMGFL`, `NGFL_FORC` sets the number of the upper-air forcings (number of forced+nudged variables multiplied by the number of times of their forcing; for example, `NGFL_FORC=60` if two variables are forced or nudged, and each of them has 30 forcing values). The forcing is assigned in `&NAMLSFORC`, which is discussed in detail in Subsection 3.2. Changing the `NREQIN` attribute of GFL fields might also be necessary according to the available/desired variables in the `ICMSHXXXXINIT` file. One can change the name of forcing or nudging used with `YFORC_NL(1)%CNAME`; for example, `YFORC_NL(1)%CNAME='BANANA'` changes the name of the first forcing or nudging from the default `'SXXXFORC0001'` to `'SXXXBANANA'`. The corresponding fields `SXXXBANANA` must be present in the `ICMSHXXXXINIT` file on all full levels.

In `&NAMDIM`, `NPROMA=-4`, because MUSC usually consists of 4 points (one point cannot be used because of spectral dynamics), and `NPROC=1` in `&NAMPAR0`. In `&NAMDYN`, `NSITER=0` because `&NAMDYNA` is empty, switching off the predictor-corrector scheme.

In `&NEMGEO`, variables `RLAT_ACAD` and `RLON_ACAD` define the latitude and longitude, which is important for radiation. However, it does not influence the Coriolis parameter, which must be forced by `RCORIO_FORC` in `&NAMLSFORC`, if needed.

In `&NAMPHY`, one might want to make changes in physics. For instance, some cases run without radiation (`LRAY=.FALSE.` for ACRANEB2).

In `&NAMPHYDS`, variable `NSFORC` defines the number of surface forcings (the number of forced variables times the number of times they are forced).

In practice, only a few variables outside of `&NAMLSFORC` change between cases. Such variables are listed in Table 2.

3.2 Namelist &NAMLSFORC

This namelist sets various forced/nudged variables defined in `arpifs/module/yomlsforc.F90`. The meaning of variables is very nicely described there. It is recommended to use `LMUSCLFA=.TRUE.`, because it switches on the LFA output files, containing more variables than the ICMSH files.

To force a specific upper-air variable, one must set its switch to `.TRUE.`; the switches are listed in Table 3. The variables used for setting properties of the forced upper-air variables are listed in Table 4. Each variable has `N<variable>_DEB` for the first index of forcing of the given quantity and `N<variable>_NUM` for the number of forcings. For example, `NLSW_DEB=38, NLSW_NUM=37` means that the vertical velocity forcing is stored from `SURFFORC0038` to `SURFFORC0074` (74=38+37-1). Each variable also has `N<variable>_FREQ` for its update frequency and `NL_<variable>_TIME`, an array of forcing time steps.

Table 2: Commonly changed variables for a new case in other namelists than &NAMLSFORC.

variable	namelist	description
NGFL_FORC	&NAMGFL	number of upper-air forcings
LRAY	&NAMPHY	switch for radiation (ACRANEB2)
NSFORC	&NAMPHYDS	number of surface forcings
CSTOP	&NAMRIP	duration of simulation
RLAT_ACAD	&NEMGEO	latitude for radiation
RLON_ACAD	&NEMGEO	longitude for radiation

Table 3: Switches for triggering forcing and nudging of various variables.

variable	description
LGEOST_UV_FRC	geostrophic wind forcing
LUV_ADV_FRC	advection of wind
LT_ADV_FRC	temperature advection forcing
LQV_ADV_FRC	water vapor advection forcing
LSW_FRC	vertical velocity forcing
LSOMEGA_FRC	vertical velocity ω forcing
LT_NUDG	nudging of temperature
LQV_NUDG	nudging of specific humidity
LUV_NUDG	nudging of wind

Table 4: Possible forced upper-air variables. Each variable has N<variable>_DEB, N<variable>_NUM, N<variable>_FREQ, and NL_<variable>_TIME, only geostrophic wind has shared NGEOST_UV_FREQ and NL_GEOST_UV_TIME and wind advection NUV_ADV_FREQ and NL_UV_ADV_TIME.

variable	description
GEOST_U	u-component of geostrophic wind [m/s]
GEOST_V	v-component of geostrophic wind [m/s]
NU_ADV	advection of the u-component of wind [m/s/s]
NV_ADV	advection of the v-component of wind [m/s/s]
LSW	vertical velocity w [m/s]
LSOMEGA_FORC	vertical velocity ω [Pa/s]
QV_ADV	advection of vater vapor q_v [kg/kg/s]
T_ADV	advection of temperature T [K/s]

Table 5: Possible forced surface variables. The time-dependent variables have N<variable>_FORC_DEB, N<variable>_FORC_NUM, and NL_<variable>_ADV_TIME. Variables, which are not time-dependent, have assigned a value, for example, RCORIO_FORC=1.E-04. Surface pressure can be forced only from a file read in routine arpifs/adiab/cp_forcing_ps.F90.

variable	description	time-dependent
TS	surface temperature T_s [K]	yes
US	friction velocity u_* [m/s]	yes
SH	sensible heat flux [J/m^2/s]	yes
LH	latent heat flux [J/m^2/s]	yes
SPS	surface pressure p_s [Pa]	yes
RCORIO_FORC	Coriolis parameter f [s^{-1}]	no
RZ0_FORC	roughness length z_0 [m]	no
RALB_FORC	surface albedo	no
REMIS_FORC	surface emissivity	no

Table 6: Possible nudged variables. Again, each variable has options N<variable>_DEB, N<variable>_NUM, NL_<variable>_TIME, and N<variable>_FREQ with the same meaning as in Table 4, except for wind nudging, which has NUV_NUDG_NUM, NL_UV_NUDG_TIME, and NUV_NUDG_FREQ shared between both components, so only the _DEB is separated.

variable	description
U_NUDG	u-component of wind [m/s]
V_NUDG	v-component of wind [m/s]
T_NUDG	temperature T [K/s]
QV_NUDG	specific humidity q_v [kg/kg]

Consequently, there are two ways how to indicate forcing time steps. One can explicitly set times for each forcing, e.g., NL_T_ADV_TIME(10)=16660., which assigns the tenth forcing of the temperature advection (SXXXFORC0010 if NT_ADV_DEB=1) to time 16660 s of the run. This way might cause quite a long &NAMLSFORC but gives full control of the specified times and allows for non-equidistantly seeded forcing. The other option is to set N<variable>_FREQ to a given value that gives the frequency (time step) of the forcing (e.g., NLSW_FREQ=300 means the update of vertical velocity forcing every 300 seconds).

The surface variables, which can be forced, are listed in Table 5. In this case, surface forcing is triggered by LSFORCS=.TRUE. in &NAMCT0. There are no additional switches for variables here; they are triggered by setting their N<variable>_FORC_NUM > 0 or assigning them a value. The time-dependent variables (except for the surface pressure, which is read from a file in routine arpifs/adiab/cp_forcing_ps.F90) have N<variable>_FORC_DEB, N<variable>_FORC_NUM, and N<variable>_ADV_TIME. They do not have the N<variable>_FREQ option for unknown reasons. Although the latitude and longitude are set in &NEMGEO, the Coriolis parameter must be defined separately in variable RCORIO_FORC.

Several variables can also be nudged. Nudging is activated using LUV_NUDG for wind, LT_NUDG for temperature, and LQV_NUDG for water vapor (see Table 6). While forcing values are linearly interpolated between given forcing times, nudging applies relaxation to the prescribed values. Again, each variable has options N<variable>_DEB, N<variable>_NUM, N<variable>_FREQ, and NL_<variable>_TIME with the same meaning as in the case of forcing. On top of that, the relaxation must be set in the namelist with RELAX_TAUT for temperature, RELAX_TAUQ for water vapor, and RELAX_TAUU for wind. They must be set in the namelist because their default value is zero (with zero, the execution terminates on division by zero in arpifs/adiab/cp_forcing.F90).

From the technical point of view, nudging is also stored in SXXXFORCXXXX in the ICMSHXXXXINIT file as for forcing. Thus, the only difference is how SXXXFORCXXXX is assigned in the namelist.

Moreover, there is a possibility of reading nudging for variables listed in Table 6 from a file. These values are read in arpifs/adiab/cp_forcing.F90 if N<variable>=-1.

The number of forcing of one variable cannot exceed 250, which is caused by the shape of the NL_<variable>_TIME arrays, which are hard-coded in yomlsforc.F90. One can also find that every variable has an allocatable array in yomlsforc.F90, usually named NT_<variable>_TIME, which is unavailable in the namelist. Arrays NL_<variable>_TIME are assigned to this array in arpifs/setup/sulsforc.F90.

3.2.1 *Example*

Here is a complete example of the &NAMLSFORC namelist for the MPACE case.

```
&NAMLSFORC
LGEOST_UV_FRC=.TRUE., !  geostrophic wind is forced
LMUSCLFA=.TRUE., !  use MUSC with LFA files on output
LQV_ADV_FRC=.TRUE., !  forcing of advection of water vapor
LQV_NUDG=.FALSE., !  nudging of water vapor is not used
LSOMEGA_FRC=.TRUE., !  vertical velocity ω is forced
LSW_FRC=.FALSE., !  vetical velocity w is not forced
LT_ADV_FRC=.TRUE., !  forcing of advection of temperature
LT_NUDG=.FALSE., !  no nudging of temperature
LUV_ADV_FRC=.FALSE., !  no advection of wind as forcing
LUV_NUDG=.FALSE., !  no nudging of wind
NGEOST_U_DEB=1, !  forcing of the u_g starts at position 1:  SXXXFORC0001
NGEOST_U_NUM=25, !  there is 25 times of forcing u-component of geostrophic wind
NGEOST_V_DEB=26, !  forcing of v_g starts at position 26:  SXXXFORC0026
NGEOST_V_NUM=25, !  there is 25 times of forcing v-component of geostrophic wind
NLSOMEGA_DEB=101, !  forcing of ω starts at position 101:  SXXXFORC0101
NLSOMEGA_NUM=25, !  there is 25 times of forcing ω
NQV_ADV_DEB=76, !  forcing of the advection of q_v starts at position 76:  SXXXFORC0076
NQV_ADV_NUM=25, !  there is 25 times of forcing of advection of water vapor
NT_ADV_DEB=51, !  forcing of the advection of T starts at position 51:  SXXXFORC0051
NT_ADV_NUM=25, !  there is 25 times of forcing of advection of temperature
NTS_FORC_DEB= 1, !  first forcing of surface temperature is SURFFORC0001
NTS_FORC_NUM= 1, !  there is only one forcing of surface temperature
RCORIO_FORC=1.37764E-04, !  Coriolis forcing (is s^-1)
/
```

4 Running and preparing MUSC cases

A private GitHub repository with several MUSC cases contains everything needed for running a case in MUSC: https://github.com/DavNemec/musc_cases. Each case has its own directory, which contains several files.

Several idealized cases are available in the private GitHub repository. Not fully validated cases are only in the devel branch. The cases in the master branch are validated and should be ready to use. Each case has its own more detailed description with references to the original papers in the private GitHub repository.

Some of the cases in the repository are taken from the DEPHY database (https://github.com/GdR-DEPHY/DEPHY-SCM), and their definition in this case should be the same. However, the values may differ due to different interpolations to vertical levels.

Some of the idealized cases in the repository are:

- ARMCU: cumulus over land (not sea). It runs without radiation (`LRAY=.FALSE.` for ACRANEB2). Suitable for validating turbulence, shallow convection, and microphysics.
- BOMEX: cumulus over sea. It runs without radiation (as ARMCU). Good for validating turbulence and shallow convection.
- DYCOMS-II RF01: stratocumulus over ocean. This case is suitable particularly for verifying the entrainment at the top of the PBL. Radiation is turned on (`LRAY=.TRUE.` for ACRANEB2). Good for turbulence and shallow convection.
- DYCOMS-II RF02: stratocumulus over the ocean. There was less entrainment than in the RF01 subcase. In both cases, the entrainment reduces the cloud, so RF02's cloud is denser. It should produce a drizzling stratocumulus. Good for microphysics, turbulence, and shallow convection.
- GABLS1: a dry case of the stable boundary layer without radiation. Good for turbulence. **Beware that the implementation without SURFEX differs from the original prescription.** The original has prescribed surface temperature and fluxes are computed indepedently, but this implementation prescribes also fluxes.
- MPACE: arctic mixed cloud. The cloud should predominately consist of cloud water but produce some snow precipitation. Good for microphysics.
- SCLD: freezing drizzle case. Suitable for microphysics. It should run without any parameterization, except for microphysics. However, the ALARO CSC cannot switch off turbulence completely using a namelist option, so it still has turbulence on by default, but its contribution is minimal.

4.1 Content of a case directory - creation of the initfile and namelist

One of the files is the case-specific namelist settings containing forcing assignments. These values are supposed to be copied into the full namelist with other settings, which are shared with other cases and often with the 3D model. There is also an example namelist with all other fields, so one can copy-paste the case-specific namelist choices to this namelist and run the model with it.

Another file is the `make_init_case.sh` script, which creates the initial file with prescribed forcing (typical `ICMSHXXXXINIT`). Users can adjust the name of this file by modifying the variable `initfile_name`. A file containing coefficients A and B of the η-coordinate is assigned to the variable `vertical_levels_file`. The coefficient A must be normalized by the surface reference pressure (101325 Pa). This script runs a simple Fortran program, always named `name_of_the_case.f90`, which creates the input ASCII data with required meteorological fields on model levels.

The Fortran script usually goes in small steps in pressure from the ground to the atmosphere. The height and values of variables are computed for every such step. When the pressure drops below the pressure of an η-level, the current values of variables are assigned to this model level. The error of the geopotential height at the top of the standard troposphere is less than 0.005%. The assignment of values to model levels is more straightforward for data from radiosondes as one can do it immediately based on the measured pressure. The script's output creates an ASCII file with upper-air variables and forcings called `upper_air_case.dat`.

Surface data are provided in a file `surf_fields_case.dat`. Then, surface fields and upper-air fields are concatenated into one ASCII file, of which the first column contains the variable name, the second its value, and the third is 1 for spectral or 0 for a gridpoint field.

Once the ASCII file is generated, it needs to be converted to an FA file. Several utilities can be used for that according to the value of `program_for_conversion` in the `make_init_case.sh` script, ensuring that the tool can be widely used. The available values are:

- `program_for_conversion=ascii2fa`, which uses `ascii2fa` developed by Ján Mašek. This option creates the ASCII file and converts it to the FA file on a 64-bit Linux system as the statically linked binary is available in the repository.
- `program_for_conversion=gl` if one wants to use GL for conversion. It creates the input ASCII file for GL.
- `program_for_conversion=acadfa1d`, which creates namelist `nam1D` for `ascii2fa` from the model source code, relying on `acadfa1d`.

4.2 Preparing new cases - general remarks

Preparation of a new case from various sources, such as literature, model output, rawinsonde, or an theoretical case, can be done by creating an ASCII file containing all necessary variables on model levels and surface, which is then converted to an FA file with `ascii2fa`, `acadfa1d`, or `GL` tools.

One must always consider which variables are to be forced/nudged and prepare the initial profile of temperature, specific humidity, wind, and other variables with `NREQIN` $\neq 0$. In addition, the logarithm of the surface pressure (`SURFPRESSION`) and the grid-point and spectral surface geopotentials `SURFGEOPOTENTIEL` and `SPECSURFGEOPOTEN` must be specified. Finally, for the ALARO CSC, the surface temperature is required to ensure the correct functioning of the turbulence scheme and possibly roughness lengths (`SURFGZ0.THERM`, `SURFZ0.FOIS.G`, and `SURFZ0REL.FOIS.G`). Moreover, albedo (`SURFALBEDO`) and emissivity (`SURFEMISSIVITE`) are needed if radiation is used. Other surface fields are disregarded. However, most must still be provided, although they do not influence results, as the surface forcing is prescribed through surface fluxes.

Programs `EPyGrAM` or `fade` can be used to extract profiles from a model run. One needs to extract the fields mentioned above for the desired start time and probably also prescribe some forcing or nudging.

From the vertical profile from a model profile or from a case prescription, an ASCII file containing all necessary model variables at η-levels must be created. Such a file contains the variable name in the first column, its value in the second column, while the third column indicates whether the variable is spectral (1) or gridpoint (0). For example, the line

```
S054TEMPERATURE 285.32 1
```

indicates that the temperature at the level 54 is 285.32 K, and that it is a spectral variable. As `ascii2fa` is a standard Fortran program, columns can be separated by spaces (no matter how many) or tabulators.

The FA file for the new case can be created using `ascii2fa`. One needs to provide the input fields from the ASCII file, a file with A and B coefficients of the η-coordinate. The first column contains coefficient A divided by the surface pressure ($A \in [0, 1]$), the second column is the coefficient B. Finally, one sets the name of the output file, the time, and latlon coordinates.

An example of the namelist for `ascii2fa`:

```
&NAM
RLON=-156.75, !  longitude
RLAT=71.3,    !  latitude
IYEAR=2004,   !  year
IMONTH=10,    !  month
IDAY=9,       !  day
IHOUR=17,     !  hour
IMIN=00,      !  minute
CFILE_DAT='data_mpace.dat',           !  file with input data
CFILE_AB='eta_L87_chmi',              !  file with coefficients for eta
CFILE_FA='initfile_L87_ALAROCZ_MPACE', !  output fa file
/
```

where CFILE_DAT is the ASCII file with model variables, and CFILE_AB is the file with coefficients of the η-cooordinate. Note that the time is important for radiation. The longitude and latitude set here are disregarded as both the Coriolis parameter and the longitude and latitude are set in the namelist (see Section 3).

Alternatively, if one wants to use GL instead of ascii2fa, the program convert4gl.sh, which is available in the repository with cases (see Section 4), should transform the input for ascii2fa into the correct input for GL. Similarly, convert4acadfa.sh can be used with acafa1d.

4.3 Preparing new cases - step by step guidance

To prepare a new MUSC case in accordance to other cases within the database of cases, one needs to:

1. Identify a source of data, such as a published article, model output, rawinsonde observations, or a field campaign dataset.

2. Extract vertical profiles of temperature, winds, and specific humidity. Other upper-air fields are optional. Select variables for forcing or nudging. Regarding the surface fields, surface latent heat flux, sensible heat flux, and surface temperature are required, while friction velocity is optional. If the case runs with radiation, albedo and emissivity are also necessary. Finally, determine the time and location of the case.

3. Copy an existing directory from the database, preferably corresponding to a similar case, and rename all files so their name corresponds to the new case.

4. In the case.f90 file, edit the vertical loop (definitions of variables with their interpolations) going in small Δp increments from the ground upwards according to the selected vertical profiles of T, q_v, u, v, and optionally other variables, such as TKE or q_l. Similarly for the upper-air forcing. After that, edit the writing of ASCII output and modify the surface forcing at the very end of the case.f90 file.

5. Modify the names of files in make_init_case.sh to correspond with the new case.

6. Modify the namelist nam_ascii2fa_case to specify time and location of the case, as well as the files used.

7. Modify the surf_fields_case.dat file. The required changes depend on whether the radiation scheme is used or not.

   - If radiation is not used, only SURFTEMPERATURE, SURFGEOPOTENTIEL, and SPECSURFGEOPOTEN must be modified. The land-sea mask can also be set (SURFIND.TERREMER; 0 over the sea and 1 over the land), as well as the roughness lengths (SURFGZ0.THERM, SURFZ0.FOIS.G, and SURFZ0REL.FOIS.G).
   - If radiation is enabled, additional fields are required: SURFEMISSIVITY and SURFALBEDO read by the ALARO CSC regardless of the namelist values, and set REMIS_FORC and RALB_FORC in &NAMLSFORC, which is needed for the AROME CSC, regardless of the values in the initfile. Finally, one need to prescribe aerosols for the cases with radiation or if aerosols are used in microphysics.

8. Update case-specific namelist namelist_case:

   - &NAMGFL: NGFL_FORC, which sets the total number of upper-air forcings+nudgings. Set NREQINs.
   - &NAMLSFORC: assign forcing, add forcing of albedo, emissivity and Coriolis parameter if necessary.
   - &NAMPHY: LRAY, which determines using of the ALARO CSC radiation.

- &NAMPHYDS: NSFORC, which sets total number of surface forcings.
- &NAMRIP: CSTOP, which sets the duration of simulation.
- &NEMGEO: set latlon coordinates (RLAT_ACAD and RLON_ACAD), important for radiation.

5 MUSC output and plotting

As briefly explained in Section 1, MUSC produces two types of output files: ICMSH and LFA. The LFA files are written every time step and contain many more variables (see Section 6 for how to add new ones).

Using the DDH tools, one can employ `lfaminm <lfa_file>` (example: `lfaminm Out.000.0000.lfa`) to read available variables and their minimal and maximal values in the vertical profile. The vertical profile of a selected variable can be extracted using `lfac <file> <variable>` (example: `lfac Out.000.0000.lfa PAPHI` for geopotential at half-levels at time zero). One can use their preferred software for plotting.

Peter Smerkol has developed a flexible, widely usable, and fast tool for plotting, which is now a part of the private GitHub repository with cases: `https://github.com/DavNemec/musc_cases/tree/devel/muscplot`. It allows for plotting time-series, vertical profiles, distributions, differences and much more. The associated documentation can be found in the above-mentioned GitHub repository.

6 Adding new variables to MUSC LFA output

Adding the desired variable to the LFA file requires only a few lines of code. The cleanest way is to add the variable to routine `writemusc.F90`, called from `writephysio.F90`, which is called from `mf_phys.F90`. Sometimes, variables are also saved from other routines if they do not appear outside of this routine (`aplpar.F90` is a good example of such a routine).

However, one can save a variable in any routine for code development. It can be done by adding:

```
USE YOMLSFORC, ONLY: LMUSCLFA, NMUSCLFA
```

at the beginning of the routine, where use statements are. The upper-air variable can be written by:

```
IF (LMUSCLFA) CALL WRSCMR(NMUSCLFA,"Wanted variable output name",variable,KLON,KLEV).
```

For example, (`fooname` is the name to be stored in the LFA file, `foovar` is the name of the variable in the routine having dimensions `KLON`, `KLEV`):

```
IF (LMUSCLFA) CALL WRSCMR(NMUSCLFA,"fooname",foovar,KLON,KLEV).
```

If one wants to save a surface field, it can be done with:

```
IF (LMUSCLFA) CALL ECR1D(NMUSCLFA,"Wanted variable output name",variable,1,KLON).
```

Do not forget to add their interface to the routine with:

```
#include "wrscmr.intfb.h"
```

or

```
#include "ecr1d.intfb.h".
```

If one wants to add many new variables to LFA files (over around 100), the value of `JPLIS` in `Xrd/ddh/lfa_R8I4.F90` in subroutine `LFAPPLFAREU` might need to be increased as it gives the dimension/length of LFA files.

It is also possible to use the so-called easy diagnostics `GFL%EZDIAG`, which writes fields to the ICMSH files. It is a GFL field carried up to the level of `aplpar.F90` for the ALARO CSC or `apl_arome.F90` for the AROME or HARMONIE-AROME CSCs. One can adjust the code to fill it with desired data. It is a

multidimensional GFL field, and variable NGFL_EZDIAG in &NAMGFL defines its size. The names of its items can be set by YEZDIAG_NL(N)%CNAME='NAME' ($N = 1, 2, ..., $NGFL_EZDIAG) in &NAMGFL.

7 Quick guidelines for typical situations

This section provides concise guidelines for typical situations new MUSC users are likely to encounter, focusing on running the existing cases, modifying vertical levels and output fields, and handling outputs.

7.1 Running MUSC with existing cases

The easiest way is to modify the ordinary script for e001 slightly. All that is needed is to give it the initfile and namelist for the case instead of the 3D inputs (no boundary files are needed for MUSC). Also, ask for only one processor thread with NPROMA=-4. See Section 2 for details.

7.2 Changing the number of vertical levels

The η-coordinate coefficients A and B must be changed. To extract the coefficients, frodo on an ICMSH file can be used, or they are in the NODE files (A and B at half levels). Beware, A must be divided by the surface pressure ($A \in [0, 1]$). Then, one creates a file with A in the first column and B in the second column and assigns it to the variable vertical_levels_file in the make_init_case.sh script for the case from our database.

7.3 Adding new variables on output

For testing/development of code, put among the use statements:

```
USE YOMLSFORC, ONLY: LMUSCLFA,NMUSCLFA
```

and save the variable to LFA files with:

```
IF (LMUSCLFA) CALL WRSCMR(NMUSCLFA,"Wanted variable output name",variable,KLON,KLEV).
```

7.4 Reading and plotting of LFA files

Using the DDH tools. Use lfaminm <lfa_file> to read available variables and lfac <file> <variable> for extracting a vertical column of a variable from a file. For plotting, Peter Smerkol's tool can be used: https://github.com/DavNemec/musc_cases/tree/devel/muscplot.

8 Useful links

- a very useful documentation (things has not changed significantly since then): http://www.umr-cnrm.fr/gmapdoc/IMG/pdf_DOC_1D_MODEL.pdf
- DEPHY case database: https://github.com/GdR-DEPHY/DEPHY-SCM
- Our case database: https://github.com/DavNemec/musc_cases

Acknowledgement

The author is grateful to Mario Hrastinski for reviewing the document and providing extensive feedback with many suggestions, and to Ana Sljivic for reviewing the document and clarifying certain behaviours of MUSC.