Supplement of Geosci. Model Dev., 18, 1265–1286, 2025
https://doi.org/10.5194/gmd-18-1265-2025-supplement

Geoscientific Model Development
Open Access
EGU

Supplement of

The MESSy DWARF (based on MESSy v2.55.2)

Astrid Kerkweg et al.

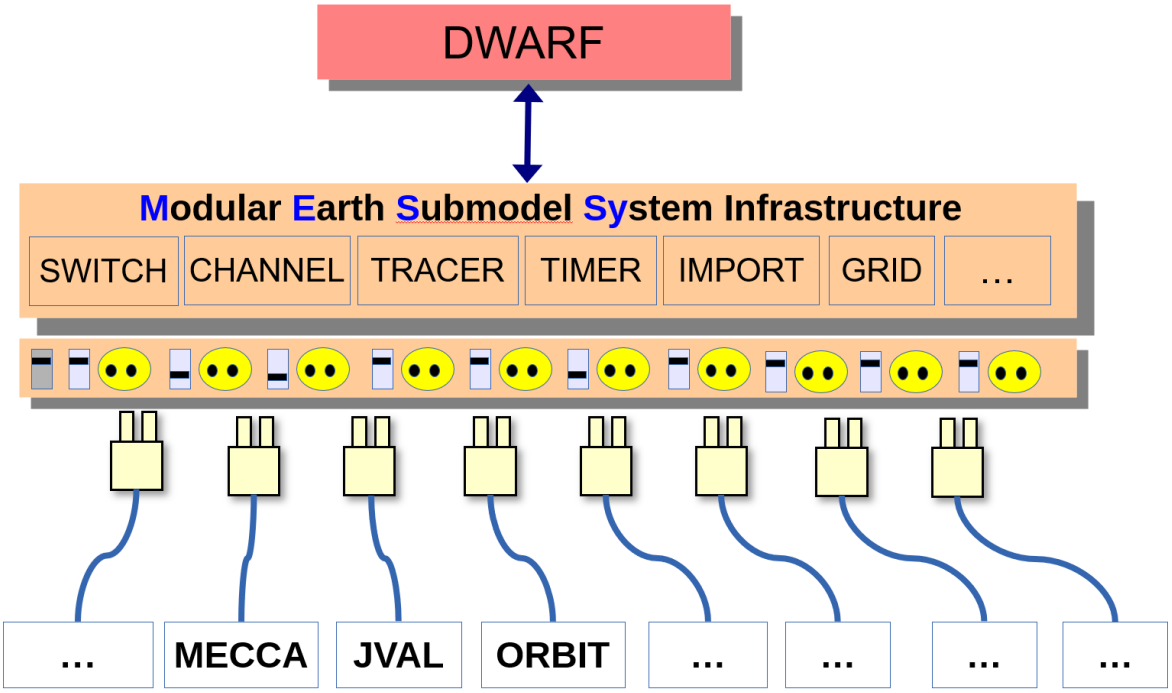
Correspondence to: Astrid Kerkweg (a.kerkweg@fz-juelich.de)

User Manual

for the generic MESSy basemodel

DWARF

Version 1.0



Date: December 10, 2024

Contents

S1 Introduction

Up to now, MESSy has been only connected to 3-D legacy models. Additionally, very simple non-parallelised (box-)models, the so-called MESSy BaseModels (MBMs), have been developed specificially for testing individual submodels. However, thanks to the MESSy structure and the expanded capabilites of the infrastructure submodels, it became possible to write a generic MESSy model, i.e., a model working like the MESSy-fied legacy models, but without legacy model. Whereas in the MESSy-fied legacy models, the basemodel layer of MESSy (BML) is the code of the legacy model itself, the BML of the generic MESSy model simply consists of the calls to the MESSy entry points in the correct order, i.e. an initial phase, and endless do-loop, which becomes a time-loop by using the MESSy infrastructure submodel TIMER, which also triggers the exit from this loop, and a finalizing phase.

The basic idea is to build a self-consistent model just using the MESSy software. The self-consistent MESSy model is named DWARF in reference to the Weather and Climate dwarfs build within the ESCAPE project (Müller et al., 2019), which were first to publish the idea of building such a framework for technical testing. As the MESSy system is still lacking a submodel containing a dynamical core, it is currently not possible to run the DWARF as a full dynamical model. However, the DWARF can be run for a lot of scientific model applications, which do not require a dynamical core, e.g., box or column model applications. Additionally, this self-consistent MESSy model can be used to technically test individual submodels (e.g. for porting the code to GPUs or for performance analysis on CPUs) with a minimum of overhead. The remaining overhead is still required, as in most cases the submodels need chemically and physically consistent input.

This DWARF manual contains a description of the DWARF software. To bridge the gap between users new to MESSy directly starting to use the DWARF and well experienced MESSy users, both, some basic MESSy knowledge as well as some very MESSy specific code design issues are addressed. However, this manual can not address all basic MESSy concepts. Please refer to the MESSy webpage (`https://www.messy-interface.org/`), and Jöckel et al. (2005, 2010) to understand the basic ideas of MESSy.

S2 The MESSy Code distribution

S2.1 Becoming a MESSy User

The MESSy DWARF is part of the MESSy software. Access to the MESSy code distribution is provided to all members of the MESSy consortium. To become a member, your institution needs to become a member of the MESSy consortium and new users need to provide a letter of intent (LOI) and sign the MESSy Community End-User Licence Agreement (EULA). Details can be found at `https://www.messy-interface.org/LICENCE` (last access: 2024-06-06).

Unfortunately, we can not yet distribute the MESSy code as open source, as within MESSy code parts have been intregrated from other license bound models. Even for the MESSy DWARF the current grid definition and parallel communication is (even though much simplified) based on the COSMO model (Doms and Baldauf, 2021) and the MESSy TIMER is based on the time control routines of ECHAM5 (Roeckner et al., 2003).

For MESSy users code access is provided via a zip-file after becoming a member of the MESSy consortium. However, for active developments of the source code access to the MESSy GitLab, which is hosted on the GitLab server of DKRZ (Deutsches Klima-RechenZentrum) `https://gitlab.dkrz.de/MESSy/MESSy`, can be granted upon request.

S2.2 MESSy – directory structure

This chapter provides a general overview of the MESSy directory structure. It can probably be skipped by users well acquainted with the MESSy software.

Listing (an exerpt of) the root directory of the MESSy code distribution, Fig. S1 reveals a number of files providing information about the code: e.g., about the Licenses (`LICENSE`, `LICENSE`), about code changes (`MESSy1_CHANGELOG`, `MESSy2_CHANGELOG` and `CHANGELOG`), a `DISCLAIMER` or `README`s. Most valuable, the file `DIRSTRUCT` contains an overview of the directory structure of the MESSy distribution. Figures S1 and S2 display (not fully complete) listings of the directory structure.

Apart from the text files, the root directory mainly contains

```
aclocal.m4
|- bin     <empty>                  executables will be put here
|- build   <empty>                  build directory in case of cmake
|- cesm1 [Makefile.in]              legacy basemodel CESM1
|  |-- ...
|
CHANGELOG
|- cmake                            cmake files
CMakeLists.txt
|- config                           configure information
configure
configure.in
CONTRIBUTING.md
|- cosmo [Makefile.in]              legacy basemodel COSMO
|  |-- src
|  |-- src_i2c
|  |-- ...
|
DIRSTRUCT
DISCLAIMER
|- echam5 [Makefile.in]             legacy basemodel ECHAM5
|  |-- src
|  |-- modules
|  |-- ...
|
|- icon [Makefile.in]               legacy basemodel ICON
|  |--src
|  |-- ...
|
|- lib     <empty>                  libraries will be put here
|- libsrc                           source code of libraries
|  |-- netcdf90
|  |-- isorropia
|  |-- oasis3-mct
|  |-- ...
|- LICENCE
LICENSE                             MESSy license
Makefile.in
|- messy                            MESSy source code
|  |
|  |-- ...
|
MESSy1_CHANGLOG
MESSy2_CHANGLOG
|- mod     <empty>                  compiled module files
|- mpiom [Makefile.in]              legacy basemodel MPIOM
|  |-- src
|  |-- src_hamooc
|
|- patches
README-cmake.md
README.md
WARNINGS
|- workdir <empty>                  standard working directory
```

Figure S1: First excerpt from the directory structure: the structure of the root directory of the MESSy code distribution. Color code: files providing information about the MESSy code distribution, standard working directory, configure/ make files, directories for compiled code: binaries (from tools, mbm and legacy basemodels) and libraries, source code for libraries, e.g., oasis3-mct, mmd, ..., the source code of the legacy basemodels and the MESSy source code.

- configure and make files: if a user works on a computer, on which MESSy is already used, no adaptions are required to these files. However, if a new host is used, the host specific configuration file (`config/mconfig.hostname@domain`) needs to be created. This file contains the machine dependent compiler and environment settings.

If you are using a LINUX laptop or workstation, you do not have to create an own `mconfig.*` file. Instead you can use the `mconfig.others` file to add your system specific paths to the required software.

- directories for compiled code:
  - bin for binaries from compiled tools, MESSy BaseModels (mbm) and legacy basemodels and
  - lib for libraries and
  - mod for compiled module files of libraries written in Fortran90 and newer.
- a standard work directory.
- the source code of
  - the libraries (libsrc) included in the MESSy source code distribution, e.g., oasis3-mct, mmd, pio, mct, isorropia, the DWD grib1 library, ...
  - the legacy basemodels ECHAM5, COSMO, CESM1 and ICON
  - the whole MESSy software.

The directory structure of the messy directory mirrors the MESSy software layers (see Fig. S2 and Jöckel et al., 2005). The messy subdirectory contains:

- source code of the MESSy submodels, more specificially,
  - the source code of the submodel core layer (smcl), i.e., the driving model independent Fortran modules of the MESSy submodels,
  - the source code of the submodel interface layer (smil), i.e., the interface layer of the regular MESSy submodels, and
  - the source code of the basemodel interface layer (bmil), i.e., the interface layer of the MESSy infrastructure submodels.
- the latex documentation of some MESSy submodels, which can be automatically compiled by `gmake docu`,
- the source code of the MESSy BaseModels (mbm), e.g., blank, caaba, dwarf. The list provided in Fig. S2 is by far not complete.
- a multitude of namelist setups (nml) of diverse MESSy models / configurations. Within the directory, the namelists of the MBMs are located in `nml/MBM/name_of_mbm`.
- additional tools (further) developed by the MESSy community, which need to be compiled by `gmake tools`, e.g., to create a new chemical mechanism, KPP and KP4 have to be compiled.
- utilities. This directory contains the default runscript (`xmessy_mmd`) as well as a multitude of helpful scripts e.g., `init_restart` to re-initialise a restart of a check-pointed simulation.
- legacy basemodel specific MESSy source code. Some MESSy code developments are limited to one legacy model only (mostly ECHAM5). This code is part of a subdirectory with the name of the respective legacy basemodel. Thanks to this structure, automatic compilation of the code for the other basemodels can be omitted without the need of adding further preprocessor directives to the code. Another advantage is, that the limitation to one specific basemodel is obvious if the submodel interface code is located in the basemodel specific subdirectories.

S3 The MESSy DWARF

The MESSy DWARF is a generic MESSy basemodel consisting of MESSy software only. The MBM DWARF itself provides just the technical environment based on the MESSy infrastructure to run individual or selections of regular MESSy submodel(s) without any legacy model. This implies, that a process, which is not yet depicted by any MESSy submodel cannot be simulated by the MESSy DWARF. Actually, the only part that is missing to run DWARF as a full dynamical atmospheric model, is a dynamical core. So far no MESSy submodel exists to

```
|- messy
|  |
|  |-- smcl                          submodel core layer modules
|  |
|  |-- smil                          shared submodel interface layer modules
|  |
|  |-- bmil                          shared basemodel interface layer modules
|  |
|  |-- docu                          latex documentation of submodels where available
|  |
|  |-- nml                           namelist setups
|  |    |-- DEFAULTS
|  |    |-- ...
|  |    |-- ...
|  |
|  |
|  |-- mbm                           messy base models
|  |    |-- blank
|  |    |-- caaba
|  |    |-- dwarf
|  |    |-- import_grid
|  |    |-- jval
|  |    |-- scav
|  |    |-- tracer
|  |    |-- ...
|  |
|  |-- tools (tools to be compiled)
|  |         This directory contains several tools developed by the MESSy community.
|  |    |
|  |    |-- biogen
|  |    |-- edgar2nc
|  |    |-- kp4
|  |    |-- kpp
|  |    |-- ncdx
|  |    |-- ...
|  |
|  |-- util                               utility scripts (incl. run-script)
|  |
|  |-- echam5                             legacy basemodel ECHAM5 specific ...
|  |    |-- bmil                          . basemodel interface layer modules
|  |    |-- smil                          . submodel interface layer modules
|  |
|  |-- cesm1                              legacy basemodel CESM1 specific ...
|  |    |-- smil                          . submodel interface layer modules
|  |
|  |-- cosmo                              legacy basemodel COSMO specific ...
|  |
|  |-- icon                               legacy basemodel ICON specific ...
|  |    |-- smil                          . submodel interface layer modules
|  |
|  |-- ...
|  |    |-- bmil
|  |    |-- smil
|  |
|
```

Figure S2: Second part of directory structure listing. Structure of the messy directory of the MESSy code distribution. Listed are subdirectories containing: the source code of the MESSy submodels, the documentation for some MESSy submodels, a multitude of namelist setups for diverse MESSy models / configurations, the source code of the MESSy BaseModels (mbm), e.g., blank, caaba, dwarf , ..., additional tools developed by the MESSy community, which need to be compiled, utilities, e.g., the runscript and legacy basemodel specific MESSy source code.

solve the equations of motion, therefore, currently DWARF is not able to run as a fully 3-D dynamical model. However, the prognostic variables (temperature, u and v wind, specific humidity, liquid (cloud) water and ice water are defined within the DWARF specific MESSy submodel DWARFDCD. They can be initialised via namelist and

are integrated, as their tendencies can be altered by MESSy submodels.

Therefore DWARF builds the environment to run an arbitrary subset of MESSy submodels, either for scientific purposes or to simplify performance analysis of individual submodels, e.g. for porting to GPUs.

S3.1 The MESSy DWARF Code structure

Section S2.2 provides an overview of the whole MESSy code distribution directory structure. According to this, the subdirectory

- `messy/mbm/dwarf` contains the code of the DWARF.
- `messy/nml/MBM/dwarf` contains the namelist setups for some DWARF configurations (see Sect. S4) including the header for the runscripts and
- `messy/util/xmessy_mmd.*` contains the runscripts, once they have been created from the headers in the namelist directories. They might need to be adapted to be used for a DWARF simulation (see Sect. S3.5),

Figure S3 shows a listing of the directory `messy/mbm/dwarf`. This directory simply contains some Makefiles required for the compilation within the MESSy environment and four subdirectories according to the four MESSy code layers.

```
|- messy/mbm/dwarf
|    |-- main.mk
|    |-- Makefile
|    |-- Makefile.m
|    |
|    |-- bml
|    |-- bmil
|    |-- smil
|    |-- smcl
```

Figure S3: Directory structure of DWARF. DWARF simply contains some Makefiles and the four source code directories according to the MESSy code structure.

For the legacy models (ECHAM5, ICON, COSMO and CESM1) the basemodel layer (BML) corresponds to the source code distribution of the respective legacy model. In contrast, the BML of DWARF consists of one file only (`messy/mbm/dwarf/bml/dwarf_main.f90`). `dwarf_main.f90` contains the main program and simply consists of a calling sequence of the MESSy entry points of the MESSy infrastructure submodel CONTROL, i.e. it provides an initial and a finalizing phase and in between an endless do-loop getting the meaning of a time loop by using the MESSy infrastructure model TIMER (the latter also provides the information for exiting the endless-do loop). In the legacy models, the entry points are called from the legacy model at those respective points during the initial phase, the integration loop, and the finalizing phase, where the respective MESSy infrastructure or process calculations are required.

The basemodel interface layer (BMIL, `messy/mbm/dwarf/bmil/`) consists of the interfaces of MESSy infrastructure submodels. The directory contains mostly links to the code files in the overall MESSy BMIL directory `messy/bmil`, i.e., these are the same files, which are also used by the legacy models. Only the basemodel specific include files (`messy_main_grid_def_dwarf.inc`, `messy_main_grid_def_mem_dwarf.inc`, `messy_main_channel_dwarf.inc` and `messy_main_data_dwarf.inc`), the CONTROL interface (`messy_main_control_dwarf.f90`) and the message passing interface (MPI) based submodel for parallelisation (`messy_main_mpi_bi.f90`) are DWARF specific files. In some cases, the commonly used files need to process data in a basemodel specific way. This is achieved by the usage of preprocessor directives (ppds). For the legacy models the ppd name is usually the name of the basemodel in capital letters (e.g., ECHAM5, COSMO, ICON) sometimes accompanied by a version specifier (e.g. COSMOv5). The ppd for the DWARF is MESSYDWARF.

The same ppds as in the BMIL are also used in the submodel interface layer (SMIL). The code compiled for the SMIL resides in two directories: (i) the standard `messy/smil/` directory and (ii) the DWARF specific subdirectory (`messy/mbm/dwarf/smil/`). The latter contains the corresponding Fortran module files for DWARF specific submodels and links to some SMIL files hosted by other MBMs (e.g., by the MBM BLANK, which is the 0-D, non-parallelised predecessor of DWARF).

The compiled objects of the SMCL Fortran modules of all MESSy submodels (located at `messy/smcl`) are linked into a library. Therefore the source code of these submodels is not part of the `messy/mbm/dwarf/smcl` source code directory, as it is sufficient to link the library during the make process. However, the directory `messy/mbm/dwarf/smcl` contains (like the `smil` directory) the files of DWARF specific submodels and the links to submodels specific for MBMs.

S3.2 How to configure the MESSy DWARF?

For production simulations the MESSy DWARF can be configured with

```
./configure --disable-ECHAM5
```

The `--disable-ECHAM5` option is necessary, as the compilation of ECHAM5/MESSy = EMAC is enabled by default in the MESSy code distribution. To avoid the compilation of EMAC , it needs to be disabled when configuring for DWARF only. In case the configuration file `config/mconfig.others` should be used, the parameter `CONF=others` is required in addition:

```
./configure --disable-ECHAM5 CONF=others
```

Just configuring as stated above, in most cases (depending on the defaults in the config files) the code will be optimized during compilation. For debugging purposes and during code developments, it is recommended to configure with

```
./configure --disable-ECHAM5 --disable-VCSTAG RUNMODE=DEBUG
```

- `RUNMODE=DEBUG` triggers the compilation of the code with run-time compiler checks, as defined in the `config/mconfig.hat.domain` file[1]. An alternative is to use `RUNMODE=DEBUGOPT`, which uses optimisation and run-time compiler checks. However, optimisation errors can not be found that way and the time required for compilation might be much longer depending on the system.
- `--disable-VCSTAG`: If working with a git clone of the MESSy code distribution, the Version Control System TAG (VCSTAG) is added to the code distribution by default. As this causes the full code to be recompiled after a small change in one of the code files, it is recommended to switch off this feature in the development phase.

S3.3 How to compile the DWARF?

To compile only the DWARF after configuration, type

```
gmake mbm target=dwarf
```

To compile the individual submodel manuals, which are part of the MESSy code distribution, run

```
gmake docu
```

The created pdf-files are in the directory `messy/docu/pdf` .

[1] In case you are using the Intel compiler `RUNMODE=DEBUG` might lead to false errors due to issues of the Intel compiler. In this case just use
`./configure --disable-ECHAM5 --disable-VCSTAG`

S3.4 The concept of DWARF

The basic idea of DWARF is to build a MESSy BaseModel (MBM) to run MESSy submodels fully self-consistently (i.e., without legacy model) within the MESSy framework. Many meaningful applications of DWARF can be envisioned e.g, box or column model applications, pure numerical tests of individual submodels etc..

Before the development of DWARF, the MESSy infrastructure was always coupled to a dynamical model (e.g. ECHAM5 or COSMO) which provided

1. external data or initial/input data defining the physical conditions of the system (e.g. sea-land fraction, roughness length, albedo ...),
2. the dynamical core and thus the prognostic variables,
3. a domain/grid definition and the parallel decomposition of the grid and
4. MPI based methods providing the most important routines for parallel communication.

These four aspects need to be replaced by other means in DWARF.

1. In a simulation with a dynamical (or legacy) model, the basemodel reads in the so-called "external data" (i.e., sea-land fractions, root depth etc.). These are provided via the MESSy infrastructure for the use in the MESSy submodels. In contrast, for the DWARF these data needs to be directly read in via the MESSy infrastructure submodel IMPORT and can be coupled as channel objects directly within the individual MESSy submodels.
2. The prognostic variables are a special case and therefore treated differently. The MESSy process submodels depend on the existence of the prognostic variables as such, as MESSy submodels access and modify prognostic variables (including tracers) via TENDENCY (Eichinger and Jöckel, 2014[2]). Therefore prognostic variables need to be defined in the DWARF, even though a dynamical core does not exit. This is done by the DWARF specific submodel DWARFDCD (DWARFs Dynamical Core Dummy). In DWARFDCD the prognostic variables are defined and initialisation and Newtonian relaxation towards imported data can be optionally requested. The prognostic variables defined by DWARFDCD are:
   - the dry air temperature `t` in [K],
   - the horizontal wind vector components `u` and `v` oriented along geographical longitudes and latitudes, in [m/s],
   - the specific humidity `q` in [kg/kg],
   - the cloud liquid water content `xl` in [kg/kg] and
   - the cloud ice water content `xi` in [kg/kg].

   Due to the integration scheme and the operator splitting, only the variables carrying the start value (or the end result of the previous integration) and the tendencies are required, both are allocated by the MESSy submodel DWARFDCD. Additionally, the MESSy infrastructure submodel TENDENCY performs the integration at the end of the timestep and the reset of the tendencies to zero at the beginning of each new time step.

   The prognostic variables `tm1_3d`, `um1_3d`, `vm1_3d`, `qm1_3d`, `xlm1_3d` and `xim1_3d` can be initialised within DWARFDCD by providing a channel object name in the `&CPL` namelist of the `dwarfdcd.nml` namelist file for the respective entries `inp_t`, `inp_u`, `inp_v`, `inp_q`, `inp_xl`, and `inp_xi`. Figure S4 shows an example of a `dwarfdcd.nml` namelist file. In this example, the channel objects `inp_t` and `inp_q` are set for the initialisation of the temperature (`tm1_3d`) and the water vapour mixing ratio (`qm1_3d`), respectively. Additionally, if required, the prognostic variables can be nudged (by Newtonian relaxation) to the provided initial or boundary field by defining a nudging coefficient (relaxation time in [s]) (`nudgedt_X`, with X being one of the prognostic variables) in the `&CTRL` namelist of the `dwarfdcd.nml` namelist file.

[2]In contrast to the information in this publication the usage of TENDENCY is now mandatory.

```
------------------------------------------------------------------------------------------
! -*- f90 -*-
&CTRL
!nudgedt_t  = 3600,
!nudgedt_q  = 3600,
!nudgedt_xl = 3600,
!nudgedt_xi = 3600,
/

&CPL
inp_t   = 'import_grid', 'inp3d_tm1',
inp_q   = 'import_grid', 'inp3d_qm1',
/
------------------------------------------------------------------------------------------
```

Figure S4: Example DWARF namelist file (`dwarfdcd.nml`) as used in DWARFDCD.

3. Usually the model grid and domain is determined by the legacy basemodel and the MESSy software operates on the respective grid. However, for the DWARF the grid/domain definition has to take place in the MESSy infrastructure itself. The generic submodel GRID (more precisely GRID_DEF) defines a grid consisting of 3 rectangular dimensions (2 horizontal/ 1 vertical)[3]. Their length and geo-location is set in the namelist `&GRID_DEF` of the namelist file `grid.nml`. The decomposition of the domain is performed by the MESSy infrastructure submodel DECOMP. In contrast to the MESSy-fied legacy basemodels, a DECOMP namelist file `decomp.nml` is required for the DWARF. Further details are provided in Sect. S3.5.3.2.

4. Usually, the MESSy infrastructure utilizes directly the MPI communication routines defined by the respective basemodel in accordance to the parallel decomposition of the basemodel. However, as DWARF defines its own parallel decomposition, also the MPI communication routines (e.g., `p_bcast` for different variable types and `gather` or `scatter` routines) need to be defined by the DWARF BMIL. Therefore, DWARF has its own `messy_main_mpi_bi.f90` code file, containing the definition of all these routines[4].

S3.5 How to set up and run DWARF?

This section describes the basics of how to set up and run a DWARF simulation. Section S4 provides some examples for DWARF configurations. To be more descriptive, this section already refers to the example DWARF configurations of Sect. S4 to demonstrate the different setup possibilities. The examples are

- a ProtoType DWARF (PT DWARF) setup, which uses only the infrastructure submodels and the regular submodel DWARFDCD, and
- a more general 3-D Atmospheric Chemistry DWARF (AC DWARF).

S3.5.1 The namelist directories

The MESSy DWARF is a MESSy BaseModel (MBM). Therefore, all namelist setups of DWARF are located in the namelist subdirectory

`messy/nml/MBM/dwarf`

The namelist setups of the three example DWARF configurations represented in Sect. S4 are

- `messy/nml/MBM/dwarf/dwarf_pt` and
- `messy/nml/MBM/dwarf/dwarf_ac`,

[3]It is planned to make the grid definition more flexible in the future.
[4]It is planned to use YAXT for the parallelisation and decomposition, respectively, in the future.

respectively. New DWARF setups can be added to the directory `messy/nml/MBM/dwarf` .

Each of the namelist directories contains the namelist files required for this specific setup. For the MESSy infrastructure submodels the following namelist files are necessary:

- `channel.nml` : configuration of output
- `data.nml` : calculation of mid-level and interface pressure, of the coriolis parameter, of the vorticity and of the density of dry air
- `decomp.nml` : definition of the parallel domain decomposition of the DWARF (see Sect. S3.5.3.2)
- `grid.nml` : definition of the DWARF grid (see Sect. S3.5.3.2)
- `import.nml` : definition of data import
- `qtimer.nml` : scheduling of restarts / check-pointing according to schedulers (queue length) constrains
- `switch.nml` : switches to activate the individual regular MESSy submodels
- `tendency.nml` : tendency diagnostic of prognostic variables
- `timer.nml` : definition of the date and time settings of the simulation, event handling, including regular check-pointing (see Sect. S3.5.3.1)
- `tracer.nml` : initialisation of tracers, definition of tracer families, activation of the positive definiteness checker (PDEF), overwriting the default tracer meta-data (TPROP).

S3.5.2 The runscript

The MESSy distribution provides one general runscript: `xmessy_mmd`. But each namelist subdirectory provides one standard header of the file, which is usually used for this setup. Running

```
gmake runscripts
```

generates all the runscripts. Running e.g.

```
./messy/util/xmkr MBM/dwarf/dwarf_ac
```

generates the runscript `messy/util/xmessy_mmd.MBM-dwarf-dwarf_ac`. `MBM/dwarf/dwarf_ac` is here the namelist directory path upward from `messy/nml`.

In the following the most important entries in the runscript that might need to be changed are explained:

- scheduler: The first block in the runscript is scheduler specific. Activate the block of scheduler statements for your specific system. If no one used the runscript on your architecture so far, add the missing parts.
- `EXP_NAME`: a meaningful experiment name should be provided. The names of all CHANNEL output files start with this string (<= 15 characters).
- `WORKDIR`: if not set, the simulation is executed in the `workdir` subdirectory in the MESSy code distribution. However, on larger computing systems it is desirable to locate the working directory on a non-backup partition able to deal with large, temporary output. Therefore the desired path can be set by, e.g.,

  ```
  WORKDIR=/scratch/userB/sim01
  ```

  Be aware, that the log-files are dumped to the directory from which the runscript has been submitted. Therefore, it is recommended to submit the runscript from the working directory to store output data, namelists and log-files consistently in one place.

  On some systems it is possible to set `WORKDIR` by

```
WORKDIR=‘pwd‘
```

Here the working directory path is set automatically to the directory from which the jobscript is submitted.

- simulation time settings: The start and the end date of the simulation need to be defined. This is achieved by setting the date components of
  - the start date: `START_YEAR`, `START_MONTH`, `START_DAY`, `START_HOUR`, `START_MINUTE`
  - and the stop date: `STOP_YEAR`, `STOP_MONTH`, `STOP_DAY`, `STOP_HOUR`, `STOP_MINUTE`.

  These variables replace the respective placeholders in the `&CTRL` namelist of the `timer.nml` namelist file and optionally some other namelists, during the initialisation. If necessary, the `START_SECOND` and `STOP_SECOND` could also be provided, but need to be added to the `timer.nml` as well.
- check-pointing settings: the MESSy TIMER generally includes a functionality to schedule check-pointing (restarts). For simplicity, the unit of the restart interval (`RESTART_UNIT`), its number (`RESTART_INTERVAL`) and the number of cycles (`NO_CYCLES`) can also be set in the runscript and are replaced in the `timer.nml` during the initialisation by the runscript. The product of `RESTART_INTERVAL` and `RESTART_UNIT` is the interval in which restart files are written, however the simulation is only terminated, and optionally restarted,after `NO_CYCLES` of these intervals.
- `NML_SETUP` indicates the chosen namelist directory, here, `[messy/nml/] MBM/dwarf/dwarf_ac` .
- `MINSTANCE`: as there are many MESSy models around (the legacy models and all MBMs) and additionally, it is possible to run more than one model in parallel (especially, in MECO(n) or OASIS setups), `MINSTANCE` provides the information, which model should be run. For DWARF the setting `MINSTANCE[1]=dwarf` is required.
- `NPX[1],NPY[1]` list the numbers of tasks in the two horizontal directions, respectively. They replace the entries in the namelist file `decomp.nml`. The product of `NPX[1]` and `NPY[1]` has to equal the number of requested tasks in the scheduler setting at the top of the runscript.

S3.5.3 Important general namelist settings

In contrast to the usual 3-D MESSy basemodels, the DWARF configuration is fully defined by MESSy namelists. More precisely, the 3-D basemodels

- establish the time stepping of the model (Sect. S3.5.3.1),
- define the grid and its parallel decomposition (Sect. S3.5.3.2), and
- they provide means of model initialisation (Sect. S3.5.4).

These three tasks need, in addition to the usual configuration of the MESSy submodels, to be accomplished with a DWARF namelist configuration. This is described in the following subsections.

S3.5.3.1 Date and Time handling

Date and time of a DWARF simulation are controlled by the MESSy infrastrucuture submodel TIMER. Figure S5 displays an example for a TIMER namelist file. It contains to namelists. The start and stop dates (`MODEL_START` and `MODEL_STOP` in units of year, month, day, hour, minute, second) of the simulation are set via the runscript (see Sect. S3.5.2). Additionally, check-pointing is possible by definition of a so-called "restart event" (`IO_RERUN_EV`). Possible entries for `RESTART_UNIT` are `'years', 'months', 'days', 'hours', 'minutes', 'seconds'` and `'steps'` and the `RESTART_INTERVAL` is any meaningful positive integer number. (For the meaning of the additional two entries, please refer to the TIMER manual, which is part of the supplement of Jöckel et al. (2010)). Restart files are written in intervals as defined by `RESTART_INTERVAL` and `RESTART_UNIT`, however, the simulation is only interrupted after `NO_CYCLES` cycles. For example:

```
IO_RERUN_EV = 1,'hours','first',0
NO_CYCLES   = 12
```

```
!-------------------------------------------------------------------------------------------------------------------
&CTRL
CAL_TYPE    = 0,  !# 0: julian calender
!                 !# 1: 360 day year
MODEL_START = $START_YEAR,$START_MONTH,$START_DAY,$START_HOUR,$START_MINUTE,00,
MODEL_STOP  = $STOP_YEAR,$STOP_MONTH,$STOP_DAY,$STOP_HOUR,$STOP_MINUTE,00,
lresume     = ${MSH_LRESUME},
!
!# set model time step length here
delta_time  = 120,               ! in seconds
!
/
&CTRL_EVENT
!# trigger restart at this time interval
!IO_RERUN_EV = 1,'months','last',0,
IO_RERUN_EV = ${RESTART_INTERVAL},'${RESTART_UNIT}','first',0,
NO_CYCLES   = ${NO_CYCLES},            ! restart cycles without break
/
!-------------------------------------------------------------------------------------------------------------------
```

Figure S5: Example `&CTRL`-namelist of the TIMER namelist file (`timer.nml`) used to setup dates and time for a DWARF simulation.

would result in hourly written restart files, but only after 12 hours the simulation would be interrupted and automatically restarted.

`LRESUME` indicates whether a simulation was restarted, i.e. it is `.FALSE.` at cold-start and `.TRUE.` at a restart. This switch is automatically set by the MESSy runscript.

`delta_time` is the integration time step length in seconds applied during the simulation.

S3.5.3.2 Define grid and decomposition

The MESSy infrastructure submodel GRID (more precisely GRID_DEF) fulfills an additional tasks for DWARF (compared to the legacy models): a grid consisting of 3 dimensions (2 horizontal / 1 vertical dimension) and its geolocation is defined via namelist in GRID_DEF. Figure S6 displays an example namelist file for GRID (`grid.nml`).

```
!-------------------------------------------------------------------------------------------------------------------
! -*- f90 -*-
&CTRL_GRID_DEF

mgpcol = 43   ! number of grid boxes simulated first horizontal dimension (x-axis)
mgprow = 63   ! number of grid boxes simulated second horizontal dimension (y-axis)
nlev   = 40   ! number of vertical grid boxes

startlon_tot = -15.0  ! longitude of mid-point of lower left corner of domain
startlat_tot = -10.0  ! latitude of mid-point of lower left corner of domain

dlon = 0.5  ! grid spacing of grid boxes in 1st hor. dimension in degree
dlat = 0.5  ! grid spacing of grid boxes in 2nd hor. dimension in degree

!-----------------------------------------------------------------------------
! height above ground for 40 levels:
vc_heighti = "22700.0, 20800.0, 19100.0, 17550.0, 16150.0, 14900.0, 13800.0, 12785.0, 11875.0, 11020.0, 10205.0,
9440.0, 8710.0, 8015.0, 7355.0, 6725.0, 6130.0, 5565.0, 5035.0, 4530.0, 4060.0, 3615.0, 3200.0, 2815.0, 2455.0,
2125.0, 1820.0, 1545.0, 1295.0, 1070.0, 870.0, 695.0, 542.0, 412.0, 303.0, 214.0, 143.0, 89.0, 49.0, 20.0, 0.0"
!-----------------------------------------------------------------------------
/

&CPL_GRID_DEF
! lignore_mass = T,
! press is required as long as lignore_mass = .FALSE. (default)
inp_press_3d   = 'import_grid', 'inp3d_press',
inp_topoheight = 'import_grid', 'inp2d_topoheight',  ! optional
/
!-------------------------------------------------------------------------------------------------------------------
```

Figure S6: Example `&CTRL_GRID_DEF` and `&CPL_GRID_DEF` namelists of the GRID namelist file (`grid.nml`) as used to setup a DWARF.

In the currently implemented simple solution the domain is rectangular with number of grid boxes `mgpcol` and `mgprow`, respectively, in the two horizontal dimensions. The geographical location of the domain is determined by the longitude and latitude (`startlon_tot` and `startlat_tot`) of the grid mid-point of the grid box in the lower left corner of the domain. The grid spacing is defined (in degree) by `dlon` and `dlat` and can be chosen for each horizontal dimension independently. The definition of the domain and its parallel decomposition, as well as the routines for the parallel data exchange have been inspired by the COSMO model (`http://www.cosmo-model.org`). However, to simplify the usage, neither halos, nor a rotation of the model domain are implemented.

The number of vertical layers is set by `nlev`. The respective height coordinate (in *'meter above ground'*) can be defined by the namelist parameter `vc_heighti`, which requires exactly `nlev+1` entries, as it is defined at the interfaces of the vertical layers. This height grid is static and horizontally uniform, however, the height of the topography could vary horizontally. At the beginning of the initialisation phase a topography height of zero is assumed. However, if `inp_topoheight` is set in the &CPL_GRID_DEF in the grid namelist in `grid.nml`, this topography height (H_{topo}) is used after the initialisation phase to calculate the actual height at a grid point (i, j, k) with i, j horizontal indices and [k] vertical index. The height of the vertical grid box interface (H^I) is defined by

$$H^I(i, j, k) = H_{topo}(i, j) + vc\_heighti(k) \tag{1}$$

Additionally, the humid air mass (`grmass`) and the dry air mass (`grmassdry`) contained in each grid box need to be provided by GRID_DEF. For the calculation of `grmass`, the pressure [Pa] and the temperature [K] are required. For calculation of `grmassdry`, additionally the specific humidity [kg/kg] is needed. The pressure is available as channel object, which name has to be given in the &CPL_GRID_DEF namelist (`inp_press_3d`). Temperature and specific humidity are prognostic variables and are directly accessible via TENDENCY.

However, there might be DWARF configurations, which do not require `grmass` or `grmassdry`. Therefore, to avoid unnecessary importing of unused fields, the calculation can be skipped with the logical namelist parameters `lignore_mass`.

```
!-------------------------------------------------------------------------------------------------------------------
! -*- f90 -*-
&CTRL
!npx = 6   ! number of tasks used in first horizontal dimension
!npy = 4   ! number of tasks used in second horizontal dimension

npx = ${NPX[1]} ! number of tasks used in first horizontal dimension replaced by the runscript
npy = ${NPY[1]} ! number of tasks used in second horizontal dimension replaced by the runscript
/
!-------------------------------------------------------------------------------------------------------------------
```

Figure S7: Example `&CTRL`namelist of DECOMP (`decomp.nml`) as used to setup a DWARF.

The decomposition of the domain is performed by the MESSy infrastructure submodel DECOMP. The `&CTRL` namelist in `decomp.nml` (see Fig. S7) contains only two parameters, defining the decomposition of the horizontal domain. The decomposition is chosen by setting `npx` and `npy`, where the product needs to equal the number of tasks requested in the scheduler settings in the runscript (Sect. S3.5.2). Usually, they are set via the runscript parameters `NPX` and `NPY` as shown by the active part of the example namelist. As an example, let's assume they are set to 6 and 4, respectively (commented lines in Fig. S7 show how the entries would look like after `${NPX[1]}` and `${NPY[1]}` have been replaced by the runscript). In this case the x-axis will be cut into 6 individual chunks and the y-axis into 4 chunks, leading to 24 patches for 24 tasks.

S3.5.4 Input data

In simulations with 3-D legacy models the initialisation and change of the most important variables (especially the prognostic variables) is driven by the basemodel. However, in the DWARF this needs to be replaced by input procedures provided by the MESSy infrastructure. These are specifically:

- the submodel IMPORT, which provides import of
  - gridded data (IMPORT_GRID, Sect. S3.5.4.1.1), where the grid of the imported data and the actual DWARF grid are independent of each other and IMPORT_GRID remaps the input data to the DWARF grid, and

`NML=`	followed by the name (including the path) of the file containing the `&REGRID` namelist for import of gridded data. If this keyword is omitted (or empty), the IMPORT namelist file itself (`import.nml`) is used.
`FILE=`	followed by the name (including the path) of the input file. IMPORT_GRID loops over all `&REGRID` namelists in the specific namelist file (`NML=...`), until the first namelist with matching netCDF filename is found. If this keyword is omitted (or empty), the first namelist in the specified namelist file (`NML=...`) is used.
`VAR=`	followed by the name of the variable that is imported by IMPORT_GRID. IMPORT_GRID loops over all IMPORT_GRID namelists in a specific namelist file (the result of `NML=`), until the first namelist with matching variable name is found (in this case, the `FILE` specifier is ignored). If this keyword is omitted, IMPORT_GRID imports all variables from `FILE`, if specified, or all variables from the first `&REGRID` namelist in `NML`.
`Z=`	followed by a comma separated list of geometric heights (in meter above ground). This is only applicable to multilevel (Nx2D) data. The number of heights must match the number of levels (N) in the input file.
`P=`	followed by a comma separated list of pressure levels (in Pa). This is only applicable to multilevel (Nx2D) data. The number of pressure levels must match the number of levels (N) in the input file.
`IPOL=`	identifies the interpolation method. It can be one of `SCRP` for SCRIP or `NRGT` for NREGRID remapping or `NONE` for raw data import.

Table 1: Table of action string options in the parameters RG_TRIG and RG_INIT being part of the &RGTINIT and &RGTEVENT namelists, respectively, in the `import.nml` namelist file.

  - time series data (IMPORT_TS, Sect. S3.5.4.1.2), where a vector of 1-D timely varying data can be read in.

- the tracer initialisation (TRACER, Sect. S3.5.4.1.3).

S3.5.4.1 How to read / import data?

The MESSy infrastructure submodel IMPORT provides dedicated interfaces for data import of gridded data from netcdf files, time series data in ASCII or netcdf format, and reading of lookup tables. Lookup tables are not required for the replacement of basemodel specific data. Thus, only IMPORT_GRID (Sect. S3.5.4.1.1) and IMPORT_TS (Sect. S3.5.4.1.2) are shortly described in the following. A documentation, which is currently not fully up-to-date, is available within the MESSy code distribution (Kerkweg and Jöckel, 20XYb). After compiling with `gmake docu`, a documentation of the IMPORT submodel can be found at `messy/docu/pdf`. For the specific task of tracer initialisation, the TRACER submodel provides an interface for tracer initialisation (Sect. S3.5.4.1.3). A documentation for the submodel TRACER is also available in the MESSy code distribution (Jöckel et al., 2008, 20XYa).

S3.5.4.1.1 IMPORT_GRID

IMPORT_GRID reads gridded netcdf data fields and interpolates them to the target (=model) grid. For the transformation / interpolation IMPORT_GRID uses the MESSy infrastructure submodel GRID (Kerkweg et al., 2018; Kerkweg and Jöckel, 20XYa).

Importing one or more gridded variables via IMPORT_GRID and their interpolation requires two namelist files:

1. The IMPORT namelist file (`import.nml`): It includes lists of all required "regrid trigger events" and "regrid initialisations" in the `&RGTEVENTS` and `&RGTINIT` namelists, respectively (see Fig. S8).

2. The namelist for the event itself (see Fig. S9).

Figure S8 shows an example of import trigger events (`RG_TRIG(.)`) and regrid initialisations (`RG_INIT(.)`).

```
!-------------------------------------------------------------------------------------------------------------------------------
! -*- f90 -*-
!
...
!           {--------------------------------------- STEPPER ---------------------------------------}
!           {- counter -} {---------------------------- action string ----------------------------}
&RGTINIT
RG_INIT(1) = 'inp3d',  1,  'NML=./import/MBM/dwarf/init/FZJ_1.0_example_X_ECHAM5_20000101_1000_3d.nml;',
RG_INIT(2) = 'inp2d',  1,  'NML=./import/MBM/dwarf/init/FZJ_1.0_example_X_ECHAM5_20000101_1000_2d.nml;',
RG_INIT(3) = 'inp2',   1,  'NML=./import/MBM/dwarf/init/FZJ_1.0_example_X_cloud_20000101_1000.nml;',
/
!
&RGTEVENTS
! ### SYNTAX:
! ###         NML=   '' (DEFAULT)   : this namelist-file (import.nml)
!                    <namelist file>: other namelist file
!
! ###         FILE=  '' (DEFAULT)   : - ONLY first netCDF-file in NML
!                    <netCDF-file>  : - this file in NML
!
! ###         VAR=   '' (DEFAULT)   : - all variables from FILE
!                                     - all variables in first namelist in NML
!                    <tracer name>  : - this variable from namelist in NML
!                                       (FILE specifier ignored !!!)
! ###         Z=     <z1,z2,...>    : - list of emission heights [m]
!                                       (above GND) for multi level emissions
!                                       (Nx2D)
! ###         P=     <p1,p2,...>    : - list of pressure levels [Pa] (Nx2D)
!
!           {--------EVENT--------}{---------------------------- STEPPER ----------------------------------------------------}
!                                  {----counter----}{----------------------------action string----------------------------------}
RG_TRIG(20) = 1,'months','first',0, 'reg3d', 1,1,1,1, 'NML=./import/MBM/dwarf/init/FZJ_1.0_example_X_ECHAM5_20000101_1000_3d.nml;',
RG_TRIG(22) = 1,'months','first',0, 'reg2d', 1,1,1,1, 'NML=./import/MBM/dwarf/init/FZJ_1.0_example_X_ECHAM5_20000101_1000_2d.nml;',
RG_TRIG(23) = 1,'years', 'first',0, 'reg2',  5,1,10,3, 'NML=./import/MBM/dwarf/init/FZJ_1.0_example_X_cloud_20000101_1000.nml;',
/
!
!-------------------------------------------------------------------------------------------------------------------------------
```

Figure S8: Example `&RGTINIT` and `&RGTEVENTS` namelists of the IMPORT namelist file (`import.nml`).

Each `RG_TRIG` entry consists of an *event* and a *stepper*. The *event* defines a periodically occuring event[5], e.g., for the regridding of an emission field every month. The *stepper* consists of a *counter* and an *action string*. The *counter* is defined as:

counter = name, min, step, max, start

It defines the cyclic stepping through the time steps of the netCDF input file. *name* is a string defining a name by which the *counter* can be identified. *start* is the initial value of the *counter* at the very first model time step. During the simulation the *counter* is incremented by *step* until *max* is reached. Afterwards, the *counter* is reset to *min*. In the example namelist, the *counter* with the name 'reg2' starts with 3, is incremented by 1 at the beginning of each month until 10 is reached, then the *counter* is reset to 5.

Finally, the *action string* controls the remapping. Table 1 lists the keywords, which, separated by semicolons, are recognised within the the IMPORT_GRID *stepper action string*.

Note that, if `VAR=` is not specified in the action string, a list of variables can be imported within one trigger event from one file. However, the fields imported within one trigger event have to have the same geometry (i.e., channel representation). Importing 2-D and 3-D data from the same file consequently requires two separate trigger events.

In addition to the regularly processed regrid trigger events, IMPORT_GRID facilitates the regridding of fields just once during the startup phase of the model. This functionality is envoked by the regrid initialisations (`RG_INIT(.)`). Their syntax is very similar to the regrid trigger events. The (`RG_INIT(.)`) do not require events, as they are called only in the initialisation phase of the model. Furthermore, the stepper function is reduced to the name of the initialisation trigger and one integer indicating the time step which should be read from the netcdf file.

Figure S9 displays, as example, the regrid namelist for `RG_TRIG(20)` in Fig. S8. The most important parameters of the namelist are

- the path and name of the netCDF file from which the imported data should be read (`infile`),
- the names of the horizontal dimension variables in the input file (in this example `i_latm` and `i_lonm`) and their respective ranges (`i_latr` and `i_lonr`),

[5]For the exact definition of an event see the User Manual of the generic MESSy submodel TIMER, Jöckel et al. (2010); Kerkweg et al. (20XY);

- the list of variables to be imported (`var`), and
- if 3-D data is imported and the data should also be interpolated vertically, the definition of the vertical grid of the imported data (in this example `i_hyam`, `i_hybm`, `i_ps` and `i_p0`).

More detailed information including a list of (all) namelist parameters can be found in Kerkweg and Jöckel (20XYb).

S3.5.4.1.2 IMPORT_TS

The IMPORT submodel IMPORT_TS reads standardised time series data from ASCII or netCDF files. Time series data generally consist of a time axis and a parameter axis. The time axis can be arbitrarily chosen, i.e., it must not be equidistant. The time axis covers data defined annually, monthly, daily, hourly, every minute or every second. The parameter axis can be freely chosen. It may consist of a number of vertical levels or be just a collection of different data. For example, the parameter axis of a radio sonde measurement could be
`lat[deg] lon[deg] height[hPa] O3[ppb] temperature[K] .`

At the beginning of a simulation, the file is read (i.e., all time steps and parameters). During the simulation the data is processed according to the namelist entries: for simulation dates, which do not exactly match the times provided by the input data, the available data next to the current date is selected by using the previous or the next point in time, or by interpolationg linearly between the two nearest points in time. For more details see Kerkweg and Jöckel (20XYb).

Figure S10 shows an example for an ASCII input data file, while Fig. S11 displays an example for the `&CTRL_TS` namelist contained in the namelist file `import.nml`. A detailed description of these examples is included in Kerkweg and Jöckel (20XYb).

S3.5.4.1.3 TRACER initialisation

The method for the initialisation of the tracers uses the same IMPORT_GRID core routines as IMPORT_GRID. However, as the regridding takes place only once in the initialisation phase and the assignment to the tracers can not be performed by IMPORT_GRID, it is conducted by the TRACER submodel. To trigger the initialistion of tracers from gridded data,

- the `L_TRACER_INIT` (a parameter of the TRACER `&CPL` namelist) needs to be .TRUE. (default is .TRUE.), and
- the required `&regrid` namelists need to be included in the `tracer.nml` namelist file.

Note, that while in IMPORT_GRID the `&regrid` namelists are placed in independent files for each regrid trigger event, for tracer the `&regrid` namelists are collected in one namelist file (namely `tracer.nml`).

```
!-------------------------------------------------------------------------------
! -*- f90 -*-
&regrid
infile    = "$INPUTDIR_MESSY/MBM/dwarf/init/FZJ_1.0_example_X_ECHAM5_20000101_1000.nc"
i_latm    = "lat",      ! name of latitude dimension in input file
i_latr    = -90.0,90.0,
i_lonm    = "lon",      ! name of longitude dimension in input file
i_hyam    = "hyam",     ! name of hybrid A coefficients in input file
i_hybm    = "hybm",     ! name of hybrid B coefficients in input file
i_ps      = "101325.0 Pa",
i_p0      = "1. Pa",    ! value of reference pressure in input file
var       = "tm1;qm1;rhum;xlm1;xim1;"
/
!-------------------------------------------------------------------------------
```

Figure S9: Example `&regrid` namelist for the regrid event RG_TRIG(20) in Fig. S8.

```
### EXAMPLE DATA SET CREATED BY A. KERKWEG
### showing mean number of trajectories
### -9999.9 is flag for undefined value
### flag(1=annualy,2=monthly,3=daily,4=hourly), 1st year, last year, columns
1 2010 2014 4
### Parameter Axis: levels [hPa]
1000.0   850.0   550.0   200.0
### data (YYYY  4 parameters)
2010  -9999.9     10.0    100.2     33.0
2011    200.0    113.2     53.7     99.1
2012     17.2  -9999.9     13.4    520.9
2013    965.9      0.0     34.2     13.8
2014     78.9     23.6    343.5  -9999.9
```

Figure S10: Example for an ASCII data file for IMPORT_TS.

```
&CTRL_TS
! ### SYNTAX:
!     - name of time series
!     - [var@] name (incl. path) of data file
!           .nc -> netCDF, e.g., "var@my_path_to_my_file/my_file.nc"
!               -> ASCII,  e.g., "my_path_to_my_file/my_file.txt"
!     - valid range ( default: -HUGE(0._dp), HUGE(0._dp) )
!     - out of time interval policy: 0: stop; 1: continue with nearest ...
!       ... (before time interval, after time interval)
!     - interpolation method:  -1: previous; 0: linear interpolation; 1: next
!     - yr,mo,dy,hr,mi,se : pick out always this date/time
!       (example: 2000, , , , , , will cycle through the year 2000 etc.)
!     - offset (in days)
!
! ### EXAMPLE netCDF ###
TS(1) = 'exnc', 'EXNC@/DATA/exnc/EXNC_1950_2012.nc',-99.90,99.90, 0, 0, 0, , , , , , , 0.0,
!
! ### EXAMPLE  ASCII ###
TS(2) = 'exascii','/DATA/example/misc/ex_1985-1990.txt', , , 0, 1, -1, 1989, , , , , , 0.0,
!
/
```

Figure S11: Example for the CTRL_TS namelist of IMPORT_TS.

S3.5.4.2 How to generate initial / input data

In principle, two types of initial or input data can be distinguished:

1. Data used in MESSy submodels independent of the driving model, as
   - gridded input data used for MESSy regular submodels (independent of the basemodel), e.g., emissions,
   - time series data, e.g., the incoming solar radiation at the top of the atmosphere, and
   - tracer initial files.
2. Data especially required for DWARF to replace the variables usually calculated by the legacy basemodel or by other switched off MESSy submodels, but required by the activated MESSy submodels.

For a simple (technical) DWARF test setup, it is sufficient to just provide these fields, i.e., to provide physically and chemically consistent fields (which dependence on the setup) in order to execute the regular MESSy submodels without problems. This section is about how to produce this second type of initial or input data.

S3.5.4.2.1 Gridded data

The most frequently required variables are the prognostic variables temperature, water vapour, liquid water and ice.In the DWARF these fields are always defined by TENDENCY and should be initialised by DATA. However, if a DWARF namelist setup is inconsistent and a required variable is not initialised, the variable is zero. This might cause technically problems (division by zero), as zero might not be a physically meaningful value (at least not for temperature).

An easy way to produce input data for DWARF is to run another MESSy legacy model (e.g. the global EMAC model) writing out the required input fields at the desired time interval. The input fields for the example DWARF configurations in Sect. S4 have been created in this way.

S3.5.4.2.2 Time series input data

In order to define some boundary conditions, it might be helpful to prescribe specific, time dependent scalar variables. As illustrated above (Sect. S3.5.4.1.2) IMPORT_TS input files (at least the ASCII input files) can be easily generated manually, as long as the number of time steps is reasonable.

S4 Example DWARFs

This section describes three example DWARF configurations, which are tested and run at the DKRZ machine levante. The idea is that these setups provide a basis for the users own development of a DWARF configuration. For further development of DWARF setups please note that all submodels, which are completely smil-ified (i.e., they are equipped with macros for the rank flip and they are not using basemodel specific routines/variables, etc.) compile with DWARF, but that only a minority of submodels was checked to really run with DWARF.

In the following the configuration, i.e., the namelists, of the two examples are described at some length, in order to provide at least a small introduction of capabilities of the MESSy software to new MESSy users starting directly with DWARF. Sect. S4.1 provides an overview of the Basemodel Interface Layer (BMIL) or more precise about the MESSy infrastructure submodels (also called generic submodels). In contrast to this, Sect. S4.2 shortly introduces a relatively simple atmospheric chemistry setup.

S4.1 The prototype dwarf

The prototype DWARF, is named as such, as it is intended to work as a prototype for all DWARF configuration developments. This setup uses only the MESSy infrastructure submodels. The only active regular submodels is DWARFDCD, which is required to define the prognostic variables. This configuration is still sufficient to test basic functionalities of IMPORT_GRID, CHANNEL output, or the correct scheduling of events (TIMER). The list of currently available (status 06.2024) generic submodels is:

- BLATHER: output of information / debug output
- BMLUSE: only used for ongoing legacy basemodel/MESSy developments (currently ICON/MESSy only)
- CHANNEL: memory and output management and check-pointing data output
- CONTROL: control of the simulation flow, call of submodels, if switched on
- DATA: exchange of data between basemodel and MESSy submodels
- DECOMP: parallel decomposition definition of a model domain
- GRID: grid transformation capabilities: DWARF special: define model grid / domain
  - GRID_DEF
  - GRID_NETCDF
- IMPORT: import data from file system into the model
  - IMPORT_GRID: import time series of gridded netCDF data

  - IMPORT_TS: import ASCII or netCDF time series data
  - IMPORT_LT: import lookup table data
- MPI: MPI functionalities, e.g., communication between tasks
- PLANET: module for simulating other planets than the Earth (only functional in EMAC)
- QTIMER: scheduler time management and check-pointing, time measurement methods
- RND: reproducible random number series
- SWITCH: switch the regular MESSy submodels
- TENDENCY: tendency diagnostic for prognostic variables, e.g. tracers. DWARF special: allocation of memory of prognostic variables, except for tracers.
- TIMER: time and event management
- TOOLS: utility functions, e.g.string to number conversion etc.
- TRACER: tracer and tracer meta-data management
  - TRACER_FAMILY: definition of families of tracers
  - TRACER_PDEF: mass diagnostics for tracers, ensurance of positive definiteness
- TRANSFORM: transformation between different spaces (e.g. Fourier and grid point space), localisation in specific decomposition.

The brown coloured submodels do not require a namelist file, while for the orange coloured submodels the namelist is either optional (e.g., BLATHER), or a namelist file is only required for specific driving models, i.e., namelist files for DATA and DECOMP are only require for DWARF setups. Those submodels without a namelist file mostly contain routines, which are called from other MESSy submodels. For example the submodel RND generating reproducible random number series is utilized by a submodel requiring random numbers.

S4.1.1 PT-DWARF namelists

In the following the most important namelist switches available for the infrastructure submodels in alphabetically order are shortly discussed:

S4.1.1.1 BLATHER

BLATHER is the submodel providing subroutines for logfile output management. Basically, four types of output are distinguished:

- information,
- warning,
- debug and
- error.

While `info`, `warning` and `error` are always dumped to the logfile, additional information useful for debugging can be triggered submodel-wise by nameing the submodels of interest in the namelist (`sms` list) and setting `i_debug` respectively.

```
!-------------------------------------------------------------------------------------
! -*- f90 -*-
&CTRL
!i_debug = 0, ! default 0 = off; 1 = selected SMs (sms); 2 = on (all SMs)
!
! sms ='sm1;sm2; ...;smX;' ! semi-colon seperated list of submodels
!sms = 'mecca;jval;tnudge;'
/
!-------------------------------------------------------------------------------------
```

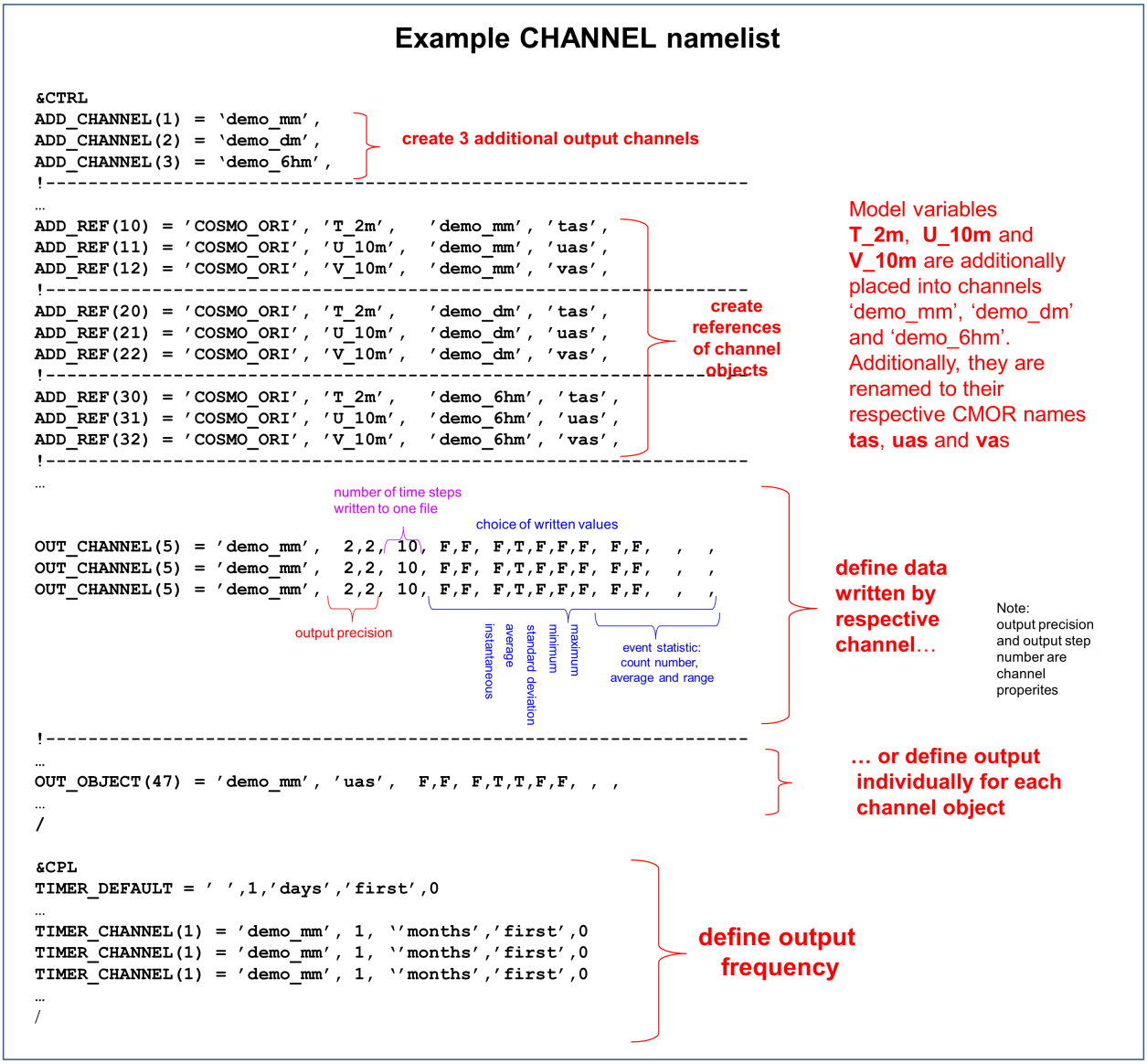


Figure S12: Excerpt from the CHANNEL namelist, illustrating the most important namelist parameters.

S4.1.1.2 CHANNEL

The generic submodel CHANNEL provides a powerful application programming interface (API) for the flexible and efficient data exchange / sharing between different processes (submodels). It is written in Fortran95 (ISO/IEC-1539-1) following an object-oriented approach to the extent possible. The basic entities of CHANNEL are

- attributes, representing time independent, scalar characteristics, e.g., the measuring unit,
- dimension variables, representing specific coordinate axes, e.g., the latitude in degrees north, the zonal wave number, the trajectory number,
- dimensions, representing the basic geometry in one dimension, e.g., the number of latitude points, the number of trajectories,
- representations, describing multidimensional geometric structures (based on dimensions), e.g., Eulerian (or gridpoint), spectral, Lagrangian,
- channel objects, representing data fields including their meta information (attributes) and their underlying geometric structure (representation), e.g., the 3-D vorticity in spectral representation, the ozone mixing ratio in Eulerian representation, the pressure altitude of trajectories in Lagrangian representation,

- channels, representing sets of "related" channel objects with additional meta information. The "relation" can be, for instance, the simple fact that the channel objects are defined by the same submodel.

CHANNEL further serves the output into data files and input/output (IO) from/into check-point (restart) files. The implemented IO features comprise

- a complete control (user interface) via two Fortran95 namelists,
- a powerful check-pointing facility for simulation chains,
- output redirection to create tailormade output files,
- a flexible choice of the output file format, of the output method, of the output precision, of the output frequency, and
- the capability to conduct basic statistical analyses w.r.t. time on-line, i.e., to output in addition (or alternative) to the instantaneous data (i.e., at a specific model time step) the average, standard deviation, minimum, maximum, event counts, and event averages for the output time interval.

Chapter 2 of the MESSy CHANNEL User Manual (Jöckel et al., 20XYb) explains the CHANNEL namelist in detail. Figure S12 illustrates the most important namelist parameters. However, it is recommended to read the full namelist description provided by the CHANNEL documentation.

S4.1.1.3 DATA

Variables, which are usually provided by the dynamical model are required by some process submodels. If these can be calculated from the basic model setup and the prognostic variables, they are calculated within DATA for the DWARF. Currently, these variables are:

- the coriolis parameter ([1/s]),
- the vorticity ([1/s]),
- the density of dry air ([kg/m^3]).

Even more important, the 3-dimensional pressure fields defined at the box centres or the vertical interfaces are required by many calculations in the MESSy submodels. These can be prescribed by imported data (in this case the imported data needs to be coupled to DATA via the `&CPL` namelist of DATA:

```
!-------------------------------------------------------------------------------
! -*- f90 -*-

&CPL
inp_press  = 'import_grid', 'inp3d_press'
inp_pressi = 'import_grid', 'inp3d_pressi'
/
!-------------------------------------------------------------------------------
```

Alternatively, the pressure fields are allocated and calculated in DATA itself following the US standard atmosphere 1976, using the height fields as determined in GRID (see Sect. S3.5.3.2).

S4.1.1.4 DECOMP

DECOMP defines the decomposition of the DWARF domain. The `decomp.nml` namelist is only required for DWARF. It is described in Sect. S3.5.3.2 and Fig. S7 displays an example namelist.

S4.1.1.5 DWARFDCD

The function of DWARFDCD and its namelists are discussed in Sect. S3.5.1 including an example `dwarfdcd.nml` namelist file (Fig. S4).

S4.1.1.6 GRID

GRID itself defines the geo-hybrid grid of the basemodel, which is required for the grid transformation routines provided within the submodel GRID. Additionally, the GRID submodel GRID_DEF contains all important grid dimension variables and all fields describing the domain (i.e., the information about the geo-location). In legacy basemodels, parts of these information are generated within the legacy model. However, some variable fields are defined for all basemodels. Additionally, for the DWARF, also the grid itself is defined in GRID_DEF . Therefore, all basemodels require a `&CPL_GRID_DEF` namelist, while the DWARF additionally requires a `&CTRL_GRID_DEF` namelist. The parameters of `grid.nml` are described in Sect. S3.5.3.2 and Fig. S6.

S4.1.1.7 IMPORT

The namelists and capabilites of IMPORT are discussed in Sect. S3.5.4.

S4.1.1.8 QTIMER

QTIMER enables the triggering of simulation restarts dependent on the consumed queue time in a scheduler.

```
!-------------------------------------------------------------------------------
! -*- f90 -*-
&CTRL
QTIME  =  $QWCH,0,0, ! QUEUE TIME LIMIT (hh,mi,se) (NOTE: 0,0,0 TO SWITCH OFF)
QCLOCK = 'wall',     ! QUEUE CLOCK TYPE ('wall','cpu ','user','sys ')
QFRAC  = 0.95        ! USABLE FRACTION OF QUEUE TIME LIMIT
/
&CPL
QMETHOD = 'max'      ! METHOD FOR PARALLEL PROCESING ('max', 'ave', 'sum')
L_DIAG  = F          ! DIAGNOSTIC OUTPUT TO LOG-FILE ?
/
!-------------------------------------------------------------------------------
```

`QTIME` should be set to the time requested in the run-script settings for the scheduler. Its syntax is integer numbers of hours, minutes, seconds. QTIMER can be effectively switched off by setting all three integers to zero. In the above example the number of integer hours is set in the runscript and the placeholder `$QWCH` is replaced by the runscript during the setup of the simulation. `QFRAC` states the fraction of the above listed queue time after which the restart is triggered. `QFRAC` should be distinctive less than 1, as after the triggering of the restart (1) at least one more time step of the integration time loop of the model is calculated, (2) the restart files need to be written to disc and (3) the run-script requires additional time to move all the files in the respective directories and setup the working directory for the next simulation cycle. `QFRAC` should be chosen depending on the complexity of the setup: the larger the domain and the more complex the configuration, the smaller `QFRAC` needs to be, as the above listed 3 steps require more time. For debugging purposes additional output can be triggered by setting `L_DIAG=T`. It is recommended to keep the other switches as they are.

MESSy comprises two options to trigger a restart, on the one hand side by QTIMER, and on the otherhand side by the restart cycle as defined in the TIMER namelist (see Sect. S4.1.1.11 and S3.5.2). The settings by TIMER are always valid, i.e., if TIMER triggers a restart before the QTIMER reaches its trigger point, the restart will be triggered as demanded by TIMER. However, if the restart interval requested in TIMER is too long to be reached within the available queue time, QTIMER will trigger a restart, independent of the restart frequency demanded in TIMER.

S4.1.1.9 SWITCH / CONTROL

SWITCH and CONTROL interact closely and thus build a working unit. While SWITCH contains and handles the switches to activate the regular MESSy submodels, CONTROL organises the calling of the respective submodels at the required entry points, if a submodel is switched on. As only the regular submodels can be switched, the `switch.nml` for the prototype dwarf is nearly empty:

```
!-------------------------------------------------------------------------------
! -*- f90 -*-
```

```
&CTRL
!
!#############################################################################
!## SWITCH MESSy-SUBMODELS ON / OFF (comment out switch USE_*)
!#############################################################################
!
L_TIME_INFO=F,
!
! L_DEBUG=F,
!
!## ------------------------------------------------------------TECHNICAL-PI---
!## -------------------------- SM1: does this and that  ----------------(AP)--
!USE_SM1=.TRUE.
!## -------------------------- additional SM2 --------------------------(PJ)--
!USE_SM2=.TRUE.
! ....
!-------------------------------------------------------------------------------
```

Note, that all but one line are comments, listing examples for the switches for the regular submodels, which are always named `USE_submodelname`. The only additional parameters are

- `L_TIME_INFO` which triggeres printing of the current model date and time into the logfile from `messy_physc` entry point every simulation step.
- `L_DEBUG`, which triggeres the output of begin and end of a MESSy entry point into the logfile. This might be helpful for debugging to narrow down the place where a model failure happens.

S4.1.1.10 TENDENCY

The TENDENCY submodel provides namelist driven tendency diagnostics. In addition, for DWARF, it allocates the memory for the tendency fields of the prognostic varibales (apart from tracers). A detailed description can be found in Eichinger and Jöckel (2014). Moreover, a short description is already provided in the `tendency.nml` namelist file itself.

```
!-------------------------------------------------------------------------------
! -*- f90 -*-

&CPL
!
l_full_diag = F  ! set TRUE to define individual channel objects for
                 ! each process and each prognostic variable or tracer
!
l_closure = F    ! set TRUE to define internal channel objects
                 ! for closure analysis
                 ! (I_HANDLE_SUM and I_HANDLE_DIFF)
!
l_clos_diag = F  ! set TRUE to get additional diagnostic log-file
                 ! output (about the closure);
                 ! this option is useful while integrating new processes
                 ! into the tendency-submodel
!
! USER DEFINED TENDENCY ANALYSES:
! SYNTAX:
! TDIAG(.) = 'variable name or tracer name',
!            'semicolon separated list of (sums of) processes'
! EXAMPLE: TDIAG(1) = 'X', 'p1;p2+p3;...;pn',
!
! Variables are: t, q, xl, xi, u, v, all GP-tracers
! Processes are:
!  - MESSy-submodels: e5vdiff, cloud, surface, convect, rad, gwave,
!                     orogw, qbo, cvtrans, dradon, ...
!
! ------------------------------------------------------
TDIAG(1)  = 'V1' ,'mecca; ddep;sedi;scav;',
TDIAG(2)  = 'V2' ,'mecca; ddep+sedi+scav;',
!
/
!-------------------------------------------------------------------------------
```

For each diagnostic variable, an entry `TDIAG` can be defined, providing a list, for which processes (`TDIAG(1)`) or combination of processes (`TDIAG(2)`) the tendency should be diagnosed (i.e., channel objects are created and tendencies are stored in these objects). In the example, the tendencies of the tracer V1 as calculated by the MESSy submodels MECCA, DDEP, SEDI and SCAV would be diagnosed, while for the tracer V2 the tendencies for the submodel MECCA and the sum of the tendencies of the sink processes calculated by DDEP, SEDI and SCAV would be stored. Additionally, the switch `l_full_diag` enables to switch on the diagnostic of all tendencies for all prognostic variables at once. This makes sense for a simple setup but requires an enormous amount of memory for global 3-D chemistry climate applications. Last but not least, a closure analysis can be triggered (`l_closure`). This is once more especially helpful during development phases to test if an implementation is complete.

S4.1.1.11 TIMER

Most functionalities of the generic MESSy submodel TIMER are discussed in the section about the runscript (Sect. S3.5.2).

S4.1.1.12 TRACER

The generic MESSy submodel TRACER handles the data and meta-data for chemical species (tracers). It is able to handle different tracer-sets, i.e. tracers defined in different representations (e.g. grid-point or lagrange), in the same simulation. Furthermore it contains two sub-submodels

- TRACER_FAMILY enables the transport of user-defined tracer-subsets as tracer families (e.g., all NO_x species). This is mostly used to ensure consistent advective transport for such tracers groups.
- TRACER_PDEF forces positive definiteness of tracer mixing ratios including a diagnostics of numerically created negative (optionally removed) tracer mass.

Jöckel et al. (20XYa) provides a full description of the namelist file `tracer.nml`. Here only a short overview is provided:

```
!-------------------------------------------------------------------------------------------------------------------
! -*- f90 -*-
&CTRL
L_FAMILY     = F, !# USE TRACER FAMILIES (SEE CTRL_FAMILY BELOW)
L_PDEF       = F, !# SWITCH TRACER SUB-SUBMODEL PDEF (SEE CPL/CTRL_PDEF BELOW)
/

&CPL
L_TRACER_INIT = T, !# SWITCH ON(T)/OFF(F) TRACER INITIALISATION
/

&CTRL_FAMILY
/

&CTRL_PDEF
!# diagnostic output ?
L_DIAGOUT = T,
!# SYNTAX: set-name, tracer-name, subname, reset to zero?, stop on exceed?, relative tolerance
TPD_DEFAULT(1) = 'gp',   '',          '',       T,             T,               0.01,
TPD(1)         = 'gp',   'HNO3',   'nat',       T,             T,                 ,
!
/

!!! ######################################################################
!!! ### TRACER INITIALISATION
!!! ######################################################################

!!! ### H2O
&regrid
infile    = "$INPUTDIR_MESSY/tracer/M2E24_exp05_0007_restart_0005_tracer_gp.nc"
i_latm    = "lat",      ! name of latitude dimension in input file
i_latr    = -90.0,90.0,
i_lonm    = "lon",      ! name of longitude dimension in input file
i_hyam    = "hyam",     ! name of hybrid A coefficients in input file
i_hybm    = "hybm",     ! name of hybrid B coefficients in input file
i_ps      = "101325.0 Pa",
i_p0      = "1. Pa",        ! value of reference pressure in input file
var       = "H2O;",
/

!!! ### MECCA + SCAV
&regrid
infile    = "$INPUTDIR_MESSY/tracer/M2E24_exp05_0007_restart_0005_tracer_gp.nc"
i_latm    = "lat",      ! name of latitude dimension in input file
i_latr    = -90.0,90.0,
i_lonm    = "lon",      ! name of longitude dimension in input file
i_hyam    = "hyam",     ! name of hybrid A coefficients in input file
i_hybm    = "hybm",     ! name of hybrid B coefficients in input file
i_ps      = "101325.0 Pa",
i_p0      = "1. Pa",        ! value of reference pressure in input file
var       = "BrNO2;H2SO4;CH3SO3H;NO3m_cs;Hp_cs;Cl2O2;CH3SO3;NH3;OClO;ClNO2;SO2;N;NH2OH;ISOOH;DMSO;CH3SO2;NH2O;HNO;
             NH2;BrCl;Br2;HOCl;DMS;C2H5O2;HOBr;CH3CO3;H;O1D;HBr;HO2;OH;O3P;Cl;Br;ClO;CH3O2;NH4pres_cs;Clmres_cs;",
/
!-------------------------------------------------------------------------------------------------------------------
```

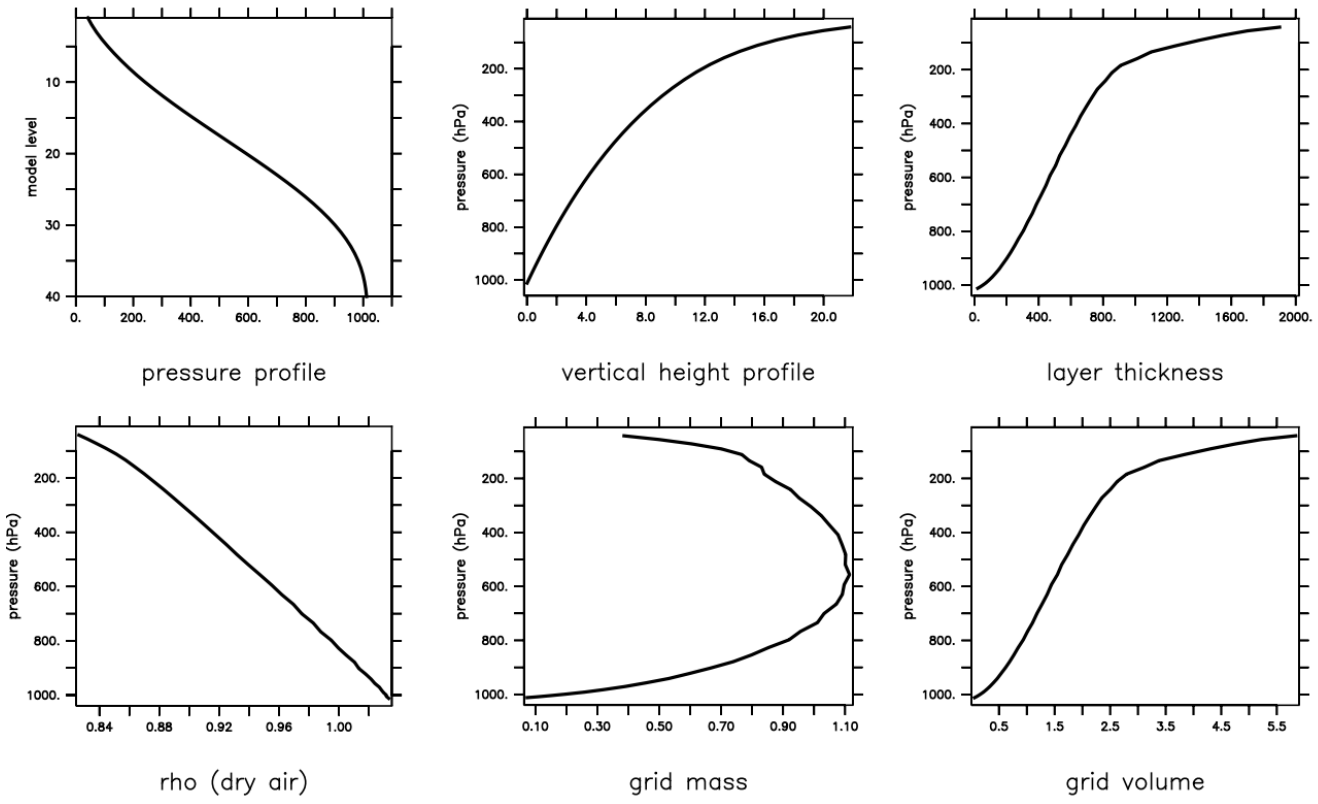


Figure S13: Illustration of the vertical structure initialised in the PT DWARF. Upper row: left: Vertical profiles of pressure [hPA] versus model levels, middle: height [km] / pressure [hPa] dependency; right: vertical profile of the layer thickness [m]. Lower row; vertical profile of (left) the density of dry air [kg/m^3], (middle) the mass of the grid box [10^{12}kg] and (right) the volume of the grid box [10^{12}m^3].

The TRACER sub-submodels are switched in the `&CTRL` namelist. Whether tracer initialisation via regridded files should take place, is controlled in the `&CPL` namelist by the namelist parameter `L_TRACER_INIT`. The required `&regrid` namelists are provided in the lower part of the namelist file. Additionally, the submodel TRACER_PDEF determines tracer set wise, whether negative tracer mixing ratios should be reset to zero, and whether the simulation should be terminated, if the (globally integrated) negative tracer mass exceeds the provided tolerance. In the example, the simulation is terminated, if the globally integrated negative mass produced within one time step is larger than 1% of the globally integrated positive tracer mass. Only for the tracer `HNO3_nat` the negative mass can be as high as the positive mass (as the default for the relative tolerance is 1. for this tracer).

S4.1.2 PT-DWARF output/results

There is not much to show for the PT DWARF, as simply the infrastructure is run. Figure S13 pictures the vertical profiles of pressure [hPA] versus model levels, the height [m]/pressure dependency [hPa], the layer thickness [m], the density of dry air ([kg/m^3]), the mass of the grid box ([10^{12}kg]) and the volume of the grid box ([10^{12}m^3]).

S4.2 The simple Atmospheric Chemistry DWARF

The simple atmospheric chemistry DWARF operates on a 3-D grid and uses the MESSy submodels MECCA (Module Efficiently Calculating the Chemistry of the Atmosphere) (Sander et al., 2005, 2011), JVAL (Sander et al., 2014) and ORBIT (Dietmüller et al., 2016). This is a basic setup for gas phase chemistry, as the kinetic chemistry calculations in MECCA require photolysis rates as input. Of course these could be prescribed, but it is more consistent to calculate them online. The photolysis code (JVAL) itself requires the solar zenith angle as input, which is calculated by ORBIT based on orbital parameters.

Optionally, if heterogeneous reaction rates in the stratosphere should be taken into account, the submodels MSBM (Multiphase Stratospheric Box Model) and TROPOP could be switched on. Where TROPOP provides the tropopause index required in MSBM to switch between tropospheric and stratospheric regimes for calculation of heterogeneous reaction coefficients in the stratosphere.

The namelist setup for the AC DWARF is available in the directory `messy/nml/MBM/dwarf_ac` of the MESSy source code distribution. Figure S14 displays the `switch.nml` for the smaller AC DWARF configuration discussed here. Un-commenting the `USE_` switches for MSBM and TROPOP would activate the expanded configuration.

Two different configurations can be chosen in the runscript generated from the runscript header (`xmessy_mmd.header`) in the `dwarf_ac` namelist setup, by setting `ZSIM` in the runscript.

- the simpler setup is `ZSIM=JVAL`. This setup runs MECCA, JVAL and ORBIT as regular submodels, while
- the setup `ZSIM=MSBM` runs MECCA, JVAL, ORBIT, MSBM and TROPOP.

Note that the runscript automatically uses the `ZSIM` specific namelist files for SWITCH, IMPORT and MECCA.

```
!-----------------------------------------------------------------------------------------------------------------
! -*- f90 -*-

&CTRL
!
!############################################################################
!## SWITCH MESSy-SUBMODELS ON / OFF (comment out switch USE_*)
!############################################################################
!
L_TIME_INFO=F,
!
!## ------------------------------------------------------------TECHNICAL-PI---
!## --------------------------- JVALues ---------------------------------(RS)--
USE_JVAL=.TRUE.
!## --------------------------- MECCA(_AERO) chemistry ---------------(RS/AK)--
USE_MECCA=.TRUE.
!## --------------------------- Multi-phase Stratospheric Box Model -----(RS)--
!USE_MSBM=.TRUE.
!## --------------------------- ORBIT -----------------------------------(PJ)--
USE_ORBIT=.TRUE.
!## --------------------------- TROPOPause ------------------------------(PJ)--
!USE_TROPOP=.TRUE.
!############################################################################
/
!-----------------------------------------------------------------------------------------------------------------
```

Figure S14: Example `switch.nml` namelist file for the AC DWARF configuration.

```
!-------------------------------------------------------------------------------------------------------------------
&CTRL_TS
! ############ ######################
! ### JVAL ### ### RAD4ALL_FUBRAD ###
! ############ ######################
!
!   NOTE:
!    The data file must either contain
!     * JVAL and/or FUBRAD:
!       - the F10.7cm in sfu (adjusted to 1 AU !!!)
!     * JVAL
!       - phi_la,SR_toa_flux,flux(7),f0(7) (adjusted to 1 AU !!!)
!     * FUBRAD
!       - TSI, Flya, Fsch1, Fs, Fl, soscale, Fherz(7:21),
!         Fhart(22:31), Fhug(32:49), Fchap
!         (units: Wm-2, except: Fsch1, Fs, Fl (erg cm-2 s-1),
!         soscale (dimensionless))
!
TS(2) = 'solact','$INPUTDIR_MESSY/jval/misc/NRLSSI_FUB1.0_hist_X_solar1AU_19500101_20111231.txt',,,0,0,0,,,,,,,0.0,
!
/
[...]
&RGTINIT
RG_INIT(1) = 'inp3d',   1, 'NML=./import/MBM/dwarf/init/FZJ_1.0_example_X_ECHAM5_20000101_1000_3d.nml;',
RG_INIT(2) = 'inp2d',   1, 'NML=./import/MBM/dwarf/init/FZJ_1.0_example_X_ECHAM5_20000101_1000_2d.nml;',
RG_INIT(3) = 'inp2',    1, 'NML=./import/MBM/dwarf/init/FZJ_1.0_example_X_cloud_20000101_1000.nml;',
/

&RGTEVENTS
! ### SYNTAX:
[...]
!
! ###########################################
! MSBM
! ###########################################
!
! H2SO4
!
! TIMER: JAN 1950 =  1        DEC 1950 = 12
!        JAN 1951 = 13        DEC 1951 = 24
!        JAN 1960 = 121       DEC 1960 = 132
!        JAN 2010 = 721       DEC 2010 = 732
!        JAN 2011 = 733       DEC 2011 = 744
!
RG_TRIG(180) = 1,'months','first',0,'H2SO4_clim',733,1,744,$START_MONTH,'VAR=H2SO4;
                           NML=./import/msbm/CCMI-ETH_MPIC1.1_hist_X_H2SO4_195001-201112.nml',
!
!###############################################
! JVAL: use upper boundary condition of ozone
!###############################################
RG_TRIG(300) = 1,'months','first',0,  'O3ubc', 1, 1, 12, $START_MONTH,
    'NML=./import/jval/HALOE_MPIC1.0_clim_X_O3_01-12.nml; VAR=O3_H;
    P=0.5,1,1.3,1.8,2.3,3.1,4.1,5.5,7.4,9.8,13,18,23,31,41,55,74,98,131,175,233,311,414,552,737,
      982,1310,1750,2330,3100,4140,5520,7360,9810,13100,17400;',
/
!-------------------------------------------------------------------------------------------------------------------
```

Figure S15: Example `import.nml` namelist file for the AC DWARF configuration.

S4.2.1 AC DWARF generic submodels and their namelists

The layout of the generic submodel namelists has already been discussed for the PT DWARF (Sect. S4.1). The only actual difference is that now the above listed variable fields need to be provided via IMPORT. Figure S15 displays the respective `import.nml` namelist file. The namelist contains

- the import of one time series (`TS(2)`) which provides the solar activity required by JVAL (see Sect. S4.2.2.2).
- three `RG_INIT` namelist entries, which trigger the reading of the required initial data:
  - `RG_INIT(1)` reads the 3-D data fields required to initialise the prognostic variables and `rhum`,
  - `RG_INIT(2)` reads the 2-D fields required to initialise the sea-land fraction (`slf`) and albedo (`alb`) required by JVAL, and
  - `RG_INIT(3)` reads the 2-D cloud cover field (`aclc`) also required by JVAL.
- two `RG_TRIG` entries, which trigger the reading of the transient input data:
  - `RG_TRIG(180)` reads an H_2SO_4 climatology required by MSBM, and
  - `RG_TRIG(300)` reads an upper boundary Ozone climatology required by JVAL.

S4.2.2 AC DWARF regular submodels and their namelists

In the following, the regular submodels used in the AC DWARF and their namelists are discussed.

S4.2.2.1 MECCA

MECCA (Module Efficiently Calculating the Chemistry of the Atmosphere) "is a kinetic chemistry submodel that contains a comprehensive atmospheric reaction mechanism. In addition to the basic $HO_x(OH + HO_2)$, $NO_x(NO + NO_2)$, and CH_4 chemistry, it also includes non-methane volatile organic compounds (NMVOC), halogens (Cl, Br, I), sulfur (S), and mercury (Hg) chemistry" (Sander et al., 2019). For the numerical integration, MECCA uses the KPP software (Sandu and Sander, 2006). An elaborate labelling and replacement system enables a high flexibility for defining scientific problem specific subsets of the available reactions.

MECCA is described in a series of articles about MECCA and/or CAABA/MECCA (Sander et al., 2005, 2011, 2019)[6].

Chapter 4 of the CAABA/MECCA User Manual (Sander, 20XYb) contains a description of how to select a mechanism or modify it using the `xmecca` script. Figure S16 provides parts of the `mecca.nml` namelist file.

`&CTRL_KPP` allows to explicitly choose the solver which is used. Usually, `icntrl(3) = 2` should be used, but for more complex liquid and aerosol phase chemistry `icntrl(3) = 5` should be used (Rosanka et al., 2023).

The `&CTRL` namelist includes switches to

- switch on more debug output within KPP (`l_kpp_debug`) and
- force the calculation of heterogeneous reaction rate (`l_force_khet`). Default is to only calculate those rates required by the chosen chemical mechanism.

The `&CPL` namelist is required to tell MECCA where the various input data should be taken from:

- `photrat_channel_gp` names the channel from which the photolysis rates are taken. In a coupled setup these are taken from the JVAL submodel. In principle they could also be read in via IMPORT_GRID. However, the naming convention for these channel objects needs to be taken into account. The object are always named `J_SPEC`, where `SPEC` stands for the name of the photolysed species[7].

[6] While MECCA is the MESSy submodel solving the kinetic reaction equations, CAABA/MECCA denotes a 0-D boxmodel, which comprises a lot of additional features specificially for boxmodel applications. CAABA/MECCA uses some other MESSy submodels, e.g. JVAL. But it does not take advantage of the MESSy infrastructure and therefore coupling to other MESSy submodel in CAABA/MECCA is hardcoded.

[7] This means, that if a `jval_gp` output from a previous simulation is imported by IMPORT_GRID, the respective trigger event `RG_TRIG()` needs to be named `'J'` and the '`var`'-string in the &regrid namelist must be something like `var ="NO2=J_NO2;OClO=J_OClO; ..` as only in this way, the final channel objects are again named `J_NO2`, `J_OClO` etc. . Alternatively, the naming convention required by JVAL might also be met, by creating correctly named channel object references via the channel namelist (`ADD_REF(.) = 'import_grid','J_NO2', 'myJVAL', 'J_NO2'`)

```
!-------------------------------------------------------------------------------------------------------
! -*- f90 -*-
&CTRL_KPP
! icntrl(3) = solver-specific method:
 icntrl(3) = 2 ! ros3: L-stable method, 3 stages, order 3 (recommended)
!icntrl(3) = 4 ! rodas3: stiffly-stable method, 4 stages, order 3
!icntrl(3) = 5 ! rodas4: stiffly-stable method, 6 stages, order 4
/
&CTRL
!l_force_khet = T ! switch on khet subsubmodel even if REQ_HET=F
!l_kpp_debug  = T ! switch on kpp debugging
/
!*****************************************************************************
! coupling namelist for MECCA
!*****************************************************************************
&CPL
! NOTE: If photolysis reactions are considered MECCA requires a submodel that
!       calculates photolysis rate coeff., e.g., JVAL.
! Choose a channel that contains J-values for photolysis reactions:
photrat_channel_gp  = 'jval_gp'
l_gp = T   ! GRIDPOINT
l_lg = F   ! LAGRANGIAN
!l_skipkpp_gp = T    ! skip call to kpp chemistry integration (GRIDPOINT)
!
c_pa_asm = 'ptrac' ! submodel for pseudo aerosol tracer properties
i_pa_amode = 4     ! corresponding mode of pseudo aerosol properties
!
! input from other MESSy submodels
inp_press    = '${MINSTANCE[$i]}','press'
inp_philon   = 'grid_def',        'philon_2d'
inp_philat   = 'grid_def',        'philat_2d'
/
!*****************************************************************************
! control namelist for MECCA_KHET subsubmodel
!*****************************************************************************
&CTRL_KHET
l_troposphere  = T
l_stratosphere = T
/
!*****************************************************************************
! coupling namelist for MECCA_KHET subsubmodel
!*****************************************************************************
&CPL_KHET
! channel object for aerosol surface climatology:
aerosurf_clim = 'import_grid', 'aerosurf_clim_A_CLIM'
! aerosol submodel and modes:
asm(2) = 'gmxe',   '1,2,3,4,5,6,7'
! aerosol chemistry coupling (submodel to calculate rate coefficients):
! (0 = aerosol surface climatology)
asm_cpl = 0
! stratosphere
strat_channel = 'msbm'
! input from other MESSy submodels
inp_press    = '${MINSTANCE[$i]}','press'
/
&CPL_AERO
/
!-------------------------------------------------------------------------------------------------------
```

Figure S16: Example `mecca.nml` namelist file for the AC DWARF configuration.

- L_GP and L_LG switch on the MECCA calculations for grid point and lagrangian space, respectively. As DWARF operates in grid point space and does not contain a lagrangian model, L_GP = T and L_LG = F is the correct setting for the DWARF.
- For some specific reaction rates, aerosol properties need to be defined. c_pa_asm and i_pa_amode enable the user to determine where this information is taken from.
- l_skipkpp_gp allows to skip the integration of MECCA. In this case also the coupled channels (i.e., jval_gp and msbm) need not to be available. This is only helpful for debugging purposes. As in this case, the full set of tracers defined by MECCA is created and available for all other submodels.
- inp_press, inp_philon and inp_philat provide the informtation, which channel object MECCA should couple to access the pressure on grid box mids, the geographical longitude and the geographical latitude.

The MECCA submodel MECCA_KHET calculates / collects heterogeneous reaction rates. In the control namelist &CTRL_KHET the user can chose, whether heterogeneous reaction rates shell be provided for the troposphere (l_troposphere) and / or the stratosphere (l_stratosphere). In the &CPL_KHET namelist it is determined, where the required input data comes from. For tropospheric heterogeneous reactions rates, either an aerosol surface climatology (aerosurf_clim) is required, or the aerosol surface is calculated from an active aerosol submodel (the entry asm(2) = 'gmxe', '1,2,3,4,5,6,7' names the aerosol submodel and which modes of the aerosol submodel shall be used). Note, that more than one asm entry can exist. The switch asm_cpl determines, which of the listed inputs is used. asm_cpl=0 triggers the usage of the aerosol surface climatology, otherwise the respective asm entry is used. In this example asm_cpl=2 would calculate the aerosol surface from modes 1 to 7 of the aerosol microphysical model GMXe.

Stratospheric heterogeneous reaction rates are taken from the submodel requested by the namelist entry strat_channel. In Fig. S16, the Multiphase Stratospheric BoxModel (MSBM) has been chosen as input channel. Additionally, MECCA_KHET requires the knowedge of the pressure at the box interfaces (inp_pressi).

In addition to the namelists shown in Fig. S16, MECCA contains also an &CPL_AERO namelist in case aerosol phase chemistry is calculated within MECCA. A description of this namelist can be found in Sander (20XYb).

S4.2.2.2 JVAL

The MESSy submodel JVAL calculates photolysis rates required in MECCA. JVAL is described in detail by Sander et al. (2014) and Sander (20XYa). Figure S17 provides an example jval.nml namelist file.

The &CTRL namelist provides a choice of the parameterisation of the CH_3CHCH_3 photolysis rate. In this example a parameterisation following Blitz et al. (2004) is chosen.

The meaning of the &CPL namelist parameters is as follows:

- l_force: Usually, JVAL analyses the list of tracers and calculates photolysis rates only for the available tracers. However, for testing (or other) purposes, it might be desirable to calculate all possible photolysis rates. This can be triggered by setting l_force = T.
- l_heating: This switch enables the calculation of the UV heating rates by oxygen and ozone as additional diagnostic quantitites, default is .FALSE..
- inp_O3: As ozone is important for the photolysis rate calculation, its distribution is required as input to JVAL. This could be either the prognostic ozone distribution (i.e., the ozone tracer 'tracer_gp','O3') or an ozone climatology e.g., imported by IMPORT_GRID ('import_grid','RADO3_O3').
- inp_O3h: not only the ozone in the model domain (as available by inp_O3) is important but also the ozone column above the model top. This is usually imported by IMPORT_GRID (inp_O3h = 'import_grid', 'O3ubc_O3_H').
- inp_cossza / inp_cdisse: Most important for the efficiency of the photolysis is the incoming solar radiation. Therefore, the cosine of the solar zenith angle inp_cossza and the distance sun - earth inp_cdisse are required. These are usually provided by the submodel ORBIT (see next paragraph).
- inp_solar: This refers to the channel object providing information on the solar activity (solar cycle). Mostly this is read in as time series by IMPORT_TS. More detailed information on this parameter is provided by Sander (20XYa).

```
!-------------------------------------------------------------------------------------------------------------------
! -*- f90 -*-
!*****************************************************************************
! control namelist for JVAL
!*****************************************************************************
&CTRL
!
!# QUANTUM YIELD FOR CH3COCH3:
!qy_ch3coch3 = 1 ! Gierzack & ECHAM5 (old IUPAC) (default)
qy_ch3coch3 = 2 ! BLITZ 2004
!qy_ch3coch3 = 3 ! IUPAC
!
/
!*****************************************************************************
! coupling namelist for JVAL
!*****************************************************************************
&CPL
!
!l_force = T,               ! calculate all species (not only for tracers)
!l_heating = T,             ! calculate UV heating rates by O2 and O3 ?
!
 inp_O3 = 'tracer_gp',  'O3',
!inp_O3 = 'import_grid', 'RADO3_O3', ! for ozone climatology (see import.nml
!
inp_cossza = 'orbit', 'cossza', ! cos(zenith angle)
inp_cdisse = 'orbit', 'cdisse', ! distance Sun-Earth (in AU)
!
! use imported ozone distribution above model top
inp_O3h    = 'import_grid', 'O3ubc_O3_H',
!
! # SOLAR CYCLE TIME SERIES; if commented, r_sol in CTRL is used instead
inp_solar = 'import_ts','solact',  ! F10.7 cm (1 parameter), or 16 parameters
!                                   ! (see &CTRL_TS in import.nml)
inp_press  = 'dwarf', 'press',      ! 3-D pressure on mid levels
inp_pressi = 'dwarf', 'pressi',    ! 3-D pressure on interface levels
inp_rhum   = 'import_grid', 'inp3d_rhum', ! 3-D relative humidity on mid levels
inp_lsclc  = 'import_grid', 'inp2_aclc',  ! 3-D large scale cloud cover
inp_slf    = 'import_grid', 'inp2d_slf ', ! 2-D land-sea fraction
inp_alb    = 'import_grid', 'inp2d_alb'   ! 2-D albedo
!
! # couple to external aerosol
! in case coupling is set to .FALSE., the internal climatology from JVAL is used
l_aero_inp = F,
/
!*****************************************************************************
!-------------------------------------------------------------------------------------------------------------------
```

Figure S17: Example `jval.nml` namelist file for the AC DWARF configuration.

- `l_aero_inp`: In addition to ozone, information about the optical properties of the aerosol in the air is required. Either, an internal climatology is used (`l_aero_inp = F`), or the following input fields need to be provided:
  - `jv_aer_sca`: the aerosol scattering extinction (per layer) ,
  - `jv_aer_abs`: the aerosol absorbing extinction (per layer), and
  - `jv_aer_ga`: the aerosol asymmetry factor for JVAL (per layer).
- Last but not least, the following variables are required from other MESSy submodels:
  - `inp_press` : the 3-D pressure on mid levels,
  - `inp_pressi` : the 3-D pressure on interface levels,
  - `inp_rhum` : the 3-D relative humidity on mid levels,
  - `inp_lsclc` : the 3-D large scale cloud cover,
  - `inp_slf` : the 2-D land-sea fraction, and
  - `inp_alb` : the 2-D albedo.

```
!-------------------------------------------------------------------------------------------------------------------
! -*- f90 -*-

&CTRL
!cecc   =  0.016715
!cobld  =  23.441
!clonp  =  282.7
!l_orbvsop87 = T    ! T for annual cycle, F for perpetual month experiments
/

&CPL
!c_rad_offset = 'rad','dt_offset',
!
inp_sinlon = 'grid_def','sinlon'
inp_sinlat = 'grid_def','sinlat'
inp_coslon = 'grid_def','coslon'
inp_coslat = 'grid_def','coslat'
!
/
!-------------------------------------------------------------------------------------------------------------------
```

Figure S18: Example `orbit.nml` namelist file for the AC DWARF configuration.

S4.2.2.3 ORBIT

The MESSy submodel ORBIT is responsible for Earth orbit calculations. The most important objects provided by ORBIT are

- the distance sun - earth (`cdisse`),
- the declination of the sun (`dec`),
- the right ascension of the sun (`ra`),
- the cosine of the solar zenith angle (`cossza`) and
- the relative day length (`rdayl`).

The `&CTRL` namelist of ORBIT (Fig. S18) provides some switching probabilities. However, for the AC DWARF configuration the `&CTRL` namelist in `orbit.nml` is essentially empty (i.e., all default values are used). The `&CPL` namelist establishes the connection to the field providing (co)sines of the longitude and latitudes, respectively (see Fig. S18). More detailed information about the ORBIT submodel are provided by Dietmüller et al. (2016).

S4.2.2.4 MSBM

The Multiphase Stratospheric Box Model (MSBM) consistently calculates heterogeneous reaction rates on polar stratospheric cloud (PSC) particles and on stratospheric background aerosol. As described in Sect. 7.2 by Jöckel et al. (2010), MSBM is a combination of the earlier MESSy1 submodels PSC and HETCHEM.

A detailed description of the current state of the polar stratospheric clouds (PSCs) parameterisations is beyond the scope of this DWARF documentation and has been published by Kirner et al. (2011) for the submodel PSC. Figure S19 lists the AC DWARF `msbm.nml` namelist file. Since the MSBM namelists are essentially identical to those of PSC, please see Kirner et al. (2011) for a detailed namelist description. For the coupling within the AC DWARF the following parameters are important:

- `LCalcChem` needs to be `.TRUE.` to trigger the creation and calculation of the channel objects for the heterogeneous reactions rates.
- `inp_Tropop_Index` is the channel object name for the tropopause index, i.e., the index of the vertical layer which includes the tropopause. The tropopause and the index of its location are usually calculated by the submodel TROPOP.
- `l_feedback` switches the dynamical-chemical feedback. `l_feedback = F` switches off the feedback by using a climatology for the sum of `HNO3_gas + HNO3_liq + HNO3_nat` for several calculations (details see Kirner et al., 2011).

```
!-------------------------------------------------------------------------------------------------------------------
! -*- f90 -*-
&CTRL
!
KinPar = T                 ! microphysical kinetic NAT scheme
!
LAdvectIceNat = F          ! advection influence on ice/NAT formation? (yes/no)
LHomNucNAT = F             ! homogeneous NAT nucleation? (yes/no)
NatFormThreshold = -3.0    ! supercooling required for NAT formation / K
minKhet = 0.0              ! minimum reaction rate / (cm**3/s)
maxKhet = 1.0e-13          ! maximum reaction rate / (cm**3/s)
!SupSatIce = 1.5            ! supersaturation required for ice formation
SupSatIce = 1.2            ! (reducd for Arctic)
r_min = 1.0e-7             ! minimum radius of solid aerosol particles / m
!                          ! with KinPar = T:
!                          !    minimum radius of ice particles / m
N_solid_max = 0.042e6      ! max. solid particle number concentration / (1/m**3)
!                          ! with KinPar = T:
!                          !    max. ice particle number concetnration /(1/m**3)
SedScheme = 3              ! switch for sedimentation scheme:
!                          !    1 = simple upwind scheme
!                          !    2 = Walcek (2000) advection scheme
!                          !    3 = trapezoid scheme
!                          ! else = no sedimentation
/
&CPL
!
LCalcChem = T             ! calculate reaction rates? (yes/no)
TempShift = 0.0           ! internal temperature shift in PSC submodel / K
!
inp_Tropop_Index = 'tropop','tp_i'  ! channel object containing tropopause index
!
r_lat     = -55.0, 45.0   ! latitude limit of PSC region (SH, NH)
!                         ! default: (-55.0, 45.0)
r_lb      = 18000.0, 18000.0  ! lower boundary of PSC region [Pa] (SH, NH)
!                             ! default: (18000., 18000.0)
r_mb      = 14000.0, 10000.0  ! middle boundary of PSC region [Pa] (SH, NH)
!                             ! default: (14000., 10000.0)
r_ub      = 500.0, 500.0    ! upper boundary of PSC region [Pa] (SH, NH)
!                             ! default: (2000., 2000.0)
!
l_feedback = T            ! feedback on dynamics
                          ! NOTE: for l_feedback=F PSCoffl_predef_HNO3
!                         !       needs to be provided via IMPORT
!
!inp_predef_HNO3_tot = 'import_grid', 'QCTM_HNO3'    ! if l_feedback = F
inp_H2SO4clim       = 'import_grid', 'H2SO4_clim_H2SO4'
!
inp_philat     = 'grid_def',    'philat_2d'   ! latitude
inp_press      = 'dwarf',       'press',      ! 3-D pressure on mid levels
inp_pressi     = 'dwarf',       'pressi',     ! 3-D pressure on interface levels
inp_aclc       = 'import_grid', 'inp2_aclc',  ! 3-D large scale cloud cover
!
/
!-------------------------------------------------------------------------------------------------------------------
```

Figure S19: Example `msbm.nml` namelist file for the AC DWARF configuration.

- `inp_predef_HNO3_tot` carries the channel and channel object name of the `HNO3_gas + HNO3_liq + HNO3_nat` climatology required if `l_feedback = F`.
- `inp_H2SO4clim` gives the channel and channel object name of an H_2SO_4 climatology. This is always required.
- additionally, MSBM requires access to the following variables provided by other MESSy submodels:
  - `inp_philat:` geographical latitude,
  - `inp_press:` pressure add mid levels,
  - `inp_pressi:` pressure at interface levels, and
  - `inp_aclc:` cloud cover

As MSBM requires the tropopause index in order to define the stratosphere region, the submodel TROPOP, diagnosing (among others) the tropopause index, is required to be switched on.

S4.2.2.5 TROPOP

TROPOP is a MESSy submodel for tropopause and other diagnostics. As the additional "other diagnostics" part is growing, this submodel will probably be renamed in the future.

Figure S20 displays the `tropop.nml` namelist file of the AC DWARF configuration. The `&CTRL` namelist parameter `l_tropop` switches the calculation of tropopause heights and level indices (i.e., in which vertical layer the tropopause is located) on. TROPOP calculates these measures for three different tropopause parameterisations:

- the climatological tropopause (calculated by `tp_clim = r_climtp(1)-r_climtp(2)*(cos(latitude))^2`),
- the WMO tropopause (either pure or corrected by the climatological tropopause `l_wmo_clim_corr = T`),
- the dynamical PV tropopause as defined by a potential vorticity level. `r_dyntp_PV` provides the value of the potential vorticity which is assumed to define the tropopause, here 3.5 PVU. `r_press_range_PV` defines the pressure range in which the PV iso-surface is searched for.

The WMO and the PV tropopause are combined to one tropopause field (which is e.g., used by MSBM in the AC DWARF setup): in the region between `-r_lat` and `r_lat` the (optinally corrected) WMO tropopause is used, while in the polar regions the dynamical PV tropopause is used.

```
!-------------------------------------------------------------------------------
! -*- f90 -*-
&CTRL
! CLIMATOLOGICAL TROPOPAUSE
r_climtp          = 300., 215., !  climatolog. tropop.: a - b * cos^2(latitude)
! WMO-TROPOPAUSE
l_wmo_clim_corr   = T   ! correct WMO tropopause with climatolog.
! PV-TROPOPAUSE
r_dyntp_PV        = 3.5 ! [PVU] |PV| at dynamical tropopause
! look for PV-iso-line in this pressure interval [Pa]
r_press_range_PV  = 5000.,80000.,
! WHERE TO COMBINE PV AND WMO TO DIAGNOSED TROPOPAUSE ?
r_lat             = 30. ! [deg] |latitude| intersect PV-WMO
! MISC. DATA
l_tropop          = T   ! calculate tropopause height and index
l_O3_PV           = F   ! calculate O3(PV) ?
l_N2O             = F   ! calculate N2O(O3)
l_NOy             = F   ! calculate NOy(N2O)
l_pblh            = F   ! calculate planetary boundary layer height
l_slp             = F   ! calculate sea level pressure
l_cpt             = F   ! cold point diagnostics
/
&CPL
 inp_press  = '${MINSTANCE[$i]}','press',       ! always required
 inp_philat = 'grid_def','philat_2d',           ! always required

 inp_zust   = 'import_grid','vdiff2d_zust',     ! req. for l_tropop
 inp_zlatkf = 'import_grid','vdiff2d_qflx',     ! req. for l_tropop
 inp_zsenkf = 'import_grid','vdiff2d_heat',     ! req. for l_tropop
 inp_vom1   = '${MINSTANCE[$i]}','vom1',        ! req. for l_tropop
 inp_coriol = '${MINSTANCE[$i]}','coriol_2d',   ! req. for l_tropop
 inp_tpot   = 'import_grid','vdiff3d_tpot',     ! req.. for l_pblh and l_tropop, l_N2
/
!-------------------------------------------------------------------------------
```

Figure S20: Example `tropop.nml` namelist file for the AC DWARF configuration.

For the diagnostic of the tropopause, the following input (in the `&CPL` namelist of TROPOP) is required:

- `inp_press:` pressure add mid-levels [hPa],
- `inp_philat:` geographical latitude [°N],
- `inp_zust:` surface friction velocity [m/s],
- `inp_zlatkf:` surface kinematic moisture flux [m/s],
- `inp_zsenkf:` surface kinematic heat flux [K m/s],
- `inp_vom1:` vorticity [1/s],
- `inp_coriol:` coriolis parameter [1/s] and
- `inp_tpot:` potential temperature [K].

Further optional diagnostics available from TROPOP are

- parameterised stratospheric ozone based on PV (`l_O3_PV`),
- parameterised stratospheric N_2O based on O_3 (`l_N2O`),
- parameterised stratospheric NO_y based on N_2O (`l_NOy`),
- a boundary layer height analysis (`l_pblh`, currently only available for ECHAM5 and COSMO),
- diagnostic of the sea-level pressure (`l_slp`),
- the calculation of the windspeed (`l_windspeed`), and
- a cold point diagnostics (`l_cpt`).

None of these are used in the current setup.

S4.2.3 AC-DWARF output/results

Example results for the AC DWARF (`ZSIM=JVAL` setup) are presented by Kerkweg et al. (2024).

S5 The traject mode: a special mode for a box model following trajectories

For the special need of a 1 grid box model following a trajectory (i.e., for the analysis of flight campaign measurement data), a so-called "traject mode" was implemented as a very special case of a MESSy DWARF. It is activated by setting the namelist parameter `l_traject = T` in the &CTRL_GRID_DEF namelist (see Fig. S21). Implementing a full 3-D grid which changes with time is a very complicated task, e.g., the data import would be required every time step as the target grid is changing. Thus, for running the DWARF in traject mode, some limitations had to be introduced:

- the traject mode is only possible for a 1 grid-box model (meaning $mgprow = mgpcol = 1$)
- IMPORT_GRID does not work for the traject mode. All input data needs to be imported via IMPORT_TS. In other words, the data imported by IMPORT_TS is the data along the respective simulated trajectory. Thus, the interpolation or the reduction of the data to the trajectory needs to be performed before the actual simulation.

```
 ! -*- f90 -*-
&CTRL_GRID_DEF
l_traject=T,
[...]
/
 &CPL_GRID_DEF
!inp_topoheight = 'import_grid', 'inp2d_topoheight',
inp_geolon  = 'import_ts', 'geolon'
inp_geolat  = 'import_ts', 'geolat'
inp_dlon    = 'import_ts', 'dlon'
inp_dlat    = 'import_ts', 'dlat'
inp_heighti = 'import_ts', 'heighti'
[...}
/
```

Figure S21: &CTRL_GRID_DEF and &CPL_GRID_DEF namelist entries specifically required for the traject mode.

If the box should be moving in time, the new coordinates need to be specified. More specifically these are

1. the geographical longitude of the grid box's mid-point (`inp_geolon`) in degrees East,
2. the geographical latitude of the grid box's mid-point (`inp_geolat`) in degrees North,
3. (optional) the vertical height of the interfaces of the grid box (`inp_heighti`) in m above topography,
4. (optional) the horizontal width of the grid box in longitudinal direction (`inp_dlon`) in degrees East,
5. (optional) the horizontal width of the grid box in latitudinal direction (`inp_dlat`) in degrees North.

The keywords in the parantheses are the respective namelist entries in the `&CPL_GRID_DEF` namelist (see Fig. S21).

Note that in the trajectory mode, the geographical longitude and latitude channel objects are mandatory. Changing the location of the grid box in height is optional. Note that the height requires the interfaces in m as input (not the box mid-point!). As a consequence, changing the height also allows for changing the vertical extend of the parcel. Additionally, the horizontal extend of the grid box can be changed, be providing data including the width of the grid box in the longitudinal (`dlon`) and latitudinal direction (`dlat`).

The initialisation of the grid box location still proceeds via the constants provided in the `&CTRL_GRID_DEF` namelist. If no channel object names are provided in the `&CPL_GRID_DEF` namelist, the values of the `&CTRL_GRID_DEF` namelist are used throughout the simulation.

S6 Some special notes

This section is just a collection of thoughts, what might have to be taken into account or might cause problems when setting up a special kind of DWARF configuration.

1. Only those regular submodels are compiled within DWARF, which interface files are located in `messy/smil`. But still this directory contains some submodels, which are only available for a single legacy basemodel and are specifically coded for this basemodel (e.g., (a) the submodel uses a hard-coded (and thus basemodel specific) rank order, i.e., rank-identifier or rank macros have not yet or not fully been introduced in the interface, or (b) basemodel specific routines are called, which are only available from that specific basemodel.) As these submodels are enclosed in the respective basemodel preprocessor directives, they are essentially empty for DWARF.
2. When new submodels are introduced in the MESSy distribution, it might have been overlooked to add calls to their entry points to `messy_main_control_dwarf.inc`. Thus the only problem in using this new submodel might be, that it is simply not called from the respective MESSy entry points. In this case, the missing calls need to be added to `messy_main_control_dwarf.inc`.
3. GRID interpolation requires always full geographical information. Thus, IMPORT_GRID might not work correctly if a 2-D model domain with 1 horizontal and 1 vertical axis is defined without providing full lon-lat information.

References

Blitz, M. A., Heard, D. E., Pilling, M. J., Arnold, S. R., and Chipperfield, M. P.: Pressure and temperature-dependent quantum yields for the photodissociation of acetone between 279 and 327.5 nm, Geophysical Research Letters, 31, doi:https://doi.org/10.1029/2003GL018793, URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2003GL018793, 2004.

Dietmüller, S., Jöckel, P., Tost, H., Kunze, M., Gellhorn, C., Brinkop, S., Frömming, C., Ponater, M., Steil, B., Lauer, A., and Hendricks, J.: A new radiation infrastructure for the Modular Earth Submodel System (MESSy, based on version 2.51), Geoscientific Model Development, 9, 2209–2222, doi:10.5194/gmd-9-2209-2016, URL https://gmd.copernicus.org/articles/9/2209/2016/, 2016.

Doms, G. and Baldauf, M.: A Description of the Nonhydrostatic Regional COSMO-Model: Part I - Dynamics and Numerics, Deutscher Wetterdienst, Frankfurter Straße 1335, 63067 Offenbach (Available at: www.cosmo-model.org), doi:10.5676/DWD_pub/nwv/cosmo-doc_6.00_I, 2021.

Eichinger, R. and Jöckel, P.: The generic MESSy submodel TENDENCY (v1.0) for process-based analyses in Earth system models, Geoscientific Model Development, 7, 1573–1582, doi:10.5194/gmd-7-1573-2014, URL https://gmd.copernicus.org/articles/7/1573/2014/, 2014.

Jöckel, P., Sander, R., Kerkweg, A., Tost, H., and Lelieveld, J.: Technical Note: The Modular Earth Submodel System (MESSy) - a new approach towards Earth System Modeling, Atmospheric Chemistry and Physics, 5, 433–444, doi:10.5194/acp-5-433-2005, URL https://acp.copernicus.org/articles/5/433/2005/, 2005.

Jöckel, P., Kerkweg, A., Buchholz-Dietsch, J., Tost, H., Sander, R., and Pozzer, A.: Technical Note: Coupling of chemical processes with the Modular Earth Submodel System (MESSy) submodel TRACER, Atmospheric Chemistry and Physics, 8, 1677–1687, doi:10.5194/acp-8-1677-2008, URL https://acp.copernicus.org/articles/8/1677/2008/, 2008.

Jöckel, P., Kerkweg, A., Pozzer, A., Sander, R., Tost, H., Riede, H., Baumgaertner, A., Gromov, S., and Kern, B.: Development cycle 2 of the Modular Earth Submodel System (MESSy2), Geoscientific Model Development, 3, 717–752, doi:10.5194/gmd-3-717-2010, URL https://gmd.copernicus.org/articles/3/717/2010/, 2010.

Jöckel, P., Kerkweg, A., Buchholz, J., Tost, H., Sander, R., and Pozzer, A.: MESSy TRACER User Manual, MESSy source code distribution: messy/docu/pdf/main_tracer.pdf, 20XYa.

Jöckel, P., Kerkweg, A., Pozzer, A., Sander, R., Tost, H., Riede, H., Baumgaertner, A., Gromov, S., and Kern, B.: MESSy CHANNEL User Manual, MESSy source code distribution: messy/docu/pdf/main_channel.pdf, 20XYb.

Kerkweg, A. and Jöckel, P.: GRID User Manual, MESSy source code distribution: messy/docu/pdf/main_grid.pdf, 20XYa.

Kerkweg, A. and Jöckel, P.: IMPORT User Manual, MESSy source code distribution: messy/docu/pdf/main_import.pdf, 20XYb.

Kerkweg, A., Hofmann, C., Jöckel, P., Mertens, M., and Pante, G.: The on-line coupled atmospheric chemistry model system MECO(n) – Part 5: Expanding the Multi-Model-Driver (MMD v2.0) for 2-way data exchange including data interpolation via GRID (v1.0), Geoscientific Model Development, 11, 1059–1076, doi:10.5194/gmd-11-1059-2018, URL https://gmd.copernicus.org/articles/11/1059/2018/, 2018.

Kerkweg, A., Kirfel, T., and ...???: The MESSy DWARF (based on MESSy 2.55.02), submitted to Geoscientific Model Development, ??, 2024.

Kerkweg, A., Riede, H., and Jöckel, P.: MESSy TIMER User Manual, MESSy source code distribution: messy/docu/pdf/main_timer.pdf, 20XY.

Kirner, O., Ruhnke, R., Buchholz-Dietsch, J., Jöckel, P., Brühl, C., and Steil, B.: Simulation of polar stratospheric clouds in the chemistry-climate-model EMAC via the submodel PSC, Geoscientific Model Development, 4, 169–182, doi:10.5194/gmd-4-169-2011, URL https://gmd.copernicus.org/articles/4/169/2011/, 2011.

Müller, A., Deconinck, W., Kühnlein, C., Mengaldo, G., Lange, M., Wedi, N., Bauer, P., Smolarkiewicz, P. K., Diamantakis, M., Lock, S.-J., Hamrud, M., Saarinen, S., Mozdzynski, G., Thiemert, D., Glinton, M., Bénard, P., Voitus, F., Colavolpe, C., Marguinaud, P., Zheng, Y., Van Bever, J., Degrauwe, D., Smet, G., Termonia, P., Nielsen, K. P., Sass, B. H., Poulsen, J. W., Berg, P., Osuna, C., Fuhrer, O., Clement, V., Baldauf, M., Gillard, M., Szmelter, J., O'Brien, E., McKinstry, A., Robinson, O., Shukla, P., Lysaght, M., Kulczewski, M., Ciznicki, M., Piątek, W., Ciesielski, S., Błażewicz, M., Kurowski, K., Procyk, M., Spychala, P., Bosak, B., Piotrowski, Z. P., Wyszogrodzki, A., Raffin, E., Mazauric, C., Guibert, D., Douriez, L., Vigouroux, X., Gray, A., Messmer, P., Macfaden, A. J., and New, N.: The ESCAPE project: Energy-efficient Scalable Algorithms for Weather Prediction at Exascale, Geoscientific Model Development, 12, 4425–4441, doi:10.5194/gmd-12-4425-2019, URL `https://gmd.copernicus.org/articles/12/4425/2019/`, 2019.

Roeckner, E., Bäuml, G., Bonaventura, L., Brokopf, R., Esch, M., Giorgetta, M., Hagemann, S., Kirchner, I., Kornblueh, L., Manzini, E., Rhodin, A., Schlese, U., Schulzweida, U., and Tompkins, A.: The atmospheric general circulation model ECHAM5, Tech. rep., Max Planck-Institute for Meteorology, 2003.

Rosanka, S., Tost, H., Sander, R., Jöckel, P., Kerkweg, A., and Taraborrelli, D.: How non-equilibrium aerosol chemistry impacts particle acidity: the GMXe AERosol CHEMistry (GMXe–AERCHEM, v1.0) sub-submodel of MESSy, EGUsphere, 2023, 1–29, doi:10.5194/egusphere-2023-2587, URL `https://egusphere.copernicus.org/preprints/2023/egusphere-2023-2587/`, 2023.

Sander, R., Kerkweg, A., Jöckel, P., and Lelieveld, J.: Technical note: The new comprehensive atmospheric chemistry module MECCA, Atmospheric Chemistry and Physics, 5, 445–450, doi:10.5194/acp-5-445-2005, URL `https://acp.copernicus.org/articles/5/445/2005/`, 2005.

Sander, R., Baumgaertner, A., Gromov, S., Harder, H., Jöckel, P., Kerkweg, A., Kubistin, D., Regelin, E., Riede, H., Sandu, A., Taraborrelli, D., Tost, H., and Xie, Z.-Q.: The atmospheric chemistry box model CAABA/MECCA-3.0, Geoscientific Model Development, 4, 373–380, doi:10.5194/gmd-4-373-2011, URL `https://gmd.copernicus.org/articles/4/373/2011/`, 2011.

Sander, R., Jöckel, P., Kirner, O., Kunert, A. T., Landgraf, J., and Pozzer, A.: The photolysis module JVAL-14, compatible with the MESSy standard, and the JVal PreProcessor (JVPP), Geoscientific Model Development, 7, 2653–2662, doi:10.5194/gmd-7-2653-2014, URL `https://gmd.copernicus.org/articles/7/2653/2014/`, 2014.

Sander, R., Baumgaertner, A., Cabrera-Perez, D., Frank, F., Gromov, S., Grooß, J.-U., Harder, H., Huijnen, V., Jöckel, P., Karydis, V. A., Niemeyer, K. E., Pozzer, A., Riede, H., Schultz, M. G., Taraborrelli, D., and Tauer, S.: The community atmospheric chemistry box model CAABA/MECCA-4.0, Geoscientific Model Development, 12, 1365–1385, doi:10.5194/gmd-12-1365-2019, URL `https://gmd.copernicus.org/articles/12/1365/2019/`, 2019.

Sander, R. e. a.: JVAL and JVPP User Manual, MESSy source code distribution: messy/mbm/jval/manual/jval_jvpp_manual.pdf, 20XYa.

Sander, R. e. a.: CAABA/MECCA User Manual, MESSy source code distribution: messy/mbm/caaba/manual/caaba_mecca_manual.pdf, 20XYb.

Sandu, A. and Sander, R.: Technical note: Simulating chemical systems in Fortran90 and Matlab with the Kinetic PreProcessor KPP-2.1, Atmospheric Chemistry and Physics, 6, 187–195, doi:10.5194/acp-6-187-2006, URL `https://acp.copernicus.org/articles/6/187/2006/`, 2006.