Introduction to DALES v3.1

authors:
Thijs Heus
Chiel van Heerwaarden
Johan van der Dussen

Department of Multi-Scale Physics
Faculty of Applied Sciences
Delft University of Technology

March 3, 2009

Contents

Chapter 1

Introduction

This is the introduction of the manual of DALES version 3.1.

Chapter 2

Change Log

Major Changes - Version 3.1

- Non-equidistant grid for 5th order advection scheme
- Mixed buoyancy mixing over the wet/dry interface
- Bulk microphysics implemented
- Adaptive Timestepping

Major Changes - Version 3.0

- 3rd order Runge Kutta integration
- Several advection schemes: 2nd, 4th and 6th order central as well as 3rd and 5th order schemes that are slightly diffusive and therefore nearly positive. The kappa scheme is also implemented.
- Allocatable arrays, meaning that new compilation is only necessary for real code changes, and not anymore for changes in resolution or number of processors. The number of processors is now not an input parameter anymore; it is read from MPI.
- New setup of the namelists; only one namelist (namoptions) file is now necessary.
- Major cleanup of the main program; most subroutines are now sorted into modules.
- The modular setup also allows for statistical routines and add-ons.
- To keep bookkeeping of open files easy, files are always directly closed after use; as an added bonus, results are directly flushed into the files upon closing.
- guard constructions are replaced; thermodynamical 'constants' 0.622, 0.61, 1.61 and 0.378 are replaced with combinations of 1, rv and rd.

Roadmap / Wish List

- Chemistry addon
- Land surface scheme
- Non-periodic boundary conditions
- Deep convection / anelastic Navier Stokes solver
- Block instead of stripe parallelization

Chapter 3

Getting Started

3.1 Installation

...

3.2 Namoptions

The model starts by reading options from a file. This file should be included before running the code. The name of the file is `namoptions`. However, you are free to choose any other name, by changing the command line options at the invocation of DALES:

```
dales3 namoptions.000
```

Some of the options in `namoptions` are required, while most have default values and are thus optional. The options are divided into different so-called namelists. The standard package contains four of them: run, domain, physics and dynamics. The options belonging to a namelist must be preceded by an ampersand (&) followed directly by the name of the namelist in capitals. There should be a backslash on the line below the last option of a namelist to close it, e.g.:

```
&RUN
option1 = value1
option2 = value2
/
```

The order of the options is not important. In table 3.1 the options required by the model and the most used options are shown. In the examples directory, a namoptions file with this minimalistic set of options is included (`namoptions.000`). The option $isurf$ requires some extra attention. This option has four possible values, one to four. Each of these values indicates a different method of calculating surface conditions. Therefore, each requires different options to be set correctly:

Table 3.1: Namelists with the most used/required options. The default value gives the value that is incorporated in the model itself. When the options is not included in the namelist, this default value is used. A '-' indicates that the model will not run without setting this value.

Option	Default	Possible values	Description	Unit
RUN				
iexpnr	000	any positive integer	Experiment number; every output filename ends with [.iexpnr]	-
dtmax	20	any positive real	Maximum timestep that is used by the model	s
runtime	300	any positive real	Total simulation (or: run) time	s
DOMAIN				
xsize	-	any positive real	Horizontal size of the simulated domain	m
ysize	-	any positive real	Horizontal size of the simulated domain	m
PHYSICS				
thls	-	any positive real	Liquid water potential temperature at the surface	K
ps	-	any positive real	Pressure at the surface	Pa
isurf	-	1,2,3,4	Flag for surface parametrization	-
z0	-	any positive real	Surface roughness	m
ustin	-	any real	Prescribed friction velocity	m s^{-1}
wtsurf	-	any positive real	Flux of liq. water pot. temp. at the surface	K m s^{-1}
wqsurf	-	any positive real	Flux of total water content	kg kg^{-1}

- isurf = 1: only z0 is required, thls is constant
- isurf = 2: only z0 is required, thls is constant
- isurf = 3: ustin, wqsurf and wtsurf are required
- isurf = 4: wqsurf and wtsurf are required

Regarding *dt_max*, it has to be noted that this is a maximum timestep. If necessary for convergence (usually at the start of the simulation) the timestep is significantly smaller. In this version of the model there is also the possibility of adaptive timestepping. The timestep is constantly calculated by means of stability criteria on the basis of the Courant and the Peclet number, $\frac{u\Delta t}{\Delta x}$ and $\mathrm{Re}_L\mathrm{Pr}$ respectively.
Table 3.2, at the end of this chapter, contains an extensive list of all available options in the standard package. Including any of these is optional: when not included in namoptions, the default values are used.

3.3 Input files

The DALES model not only needs the namoptions file. For a proper run it also requires initial vertical profiles of different quantities. These profiles are contained in several different input files. At default, only two files are used, namely `prof.inp.[iexpnr]` and `lscale.inp.[iexpnr]` (here [iexpnr] denotes the value of iexpnr contained in `namoptions`. In the rest of this chapter, this extension is omitted). Examples of input files can be found in the examples folder.

prof.inp

This file contains six columns separated by white spaces and contains the initial vertical profiles of different quantities. The first (leftmost) column contains the height. The information in this column is used to see whether the grid is equidistant. In case of an equidistant grid, the column should start with $dz/2$ and has $kmax$ entries (see figure 3.1). These levels are the half levels.

Figure 3.1: An example of the first part of a `prof.inp` input file. In this case $kmax$ = 64 and the domain height is 1600 meters. This results in a vertical spacing dz of 25 meters. The first level is at $dz/2$ = 12.5m.

```
ASTEX Flight A209 exp nr 000
      height         th_l                qt              u              v       tke
        12.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
        37.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
        62.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
        87.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       112.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       137.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       162.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       187.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       212.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       237.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       262.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       287.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       312.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       337.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       362.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       387.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       412.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       437.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       462.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       487.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       512.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       537.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       562.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       587.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       612.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       637.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       662.5000      288.000000      1.020000e-02    -0.700000      -10.000000      1.000000
       687.5000      290.750000      9.650000e-03    -1.350000      -10.000000      0.000000
       712.5000      293.500000      9.100000e-03    -2.000000      -10.000000      0.000000
       737.5000      293.650000      9.030000e-03    -2.000000      -10.000000      0.000000
```

The following columns contain the profiles at *height* of the following quantities (from left to right):

- liquid water potential temperature in K, θ_l
- total humidity in kg kg^{-1}, q_t
- horizontal wind velocity (in the x-direction) in m s^{-1}, u
- horizontal wind velocity (in the y-direction) in m s^{-1}, v
- subfilter scale turbulent kinetic energy in m^2 s^2, TKE

The first two lines of the file are unimportant and can be used as headers containing experiment information or column descripitions, for instance.

lscale.inp

This file is made up of eight columns. It contains information about large scale forcings of the simulation. Like the profiles file, it starts with two unimportant header lines. Furthermore, the leftmost column is again the height, the vertical half levels of the domain, with *kmax* entries. For an example, see figure 3.2.

Figure 3.2: An example of the first few lines of a `lscale.inp` input file. Here, $kmax$ = 64 and the domain height is 1600 meters. This results in a vertical spacing dz of 25 meters. The first level is at $dz/2$ = 12.5m. Most forcings are set to zero.

```
ASTEX Flight A209 exp nr 000
height          ug          vg          wfls          dqtdx          dqtdy       dqtdtls       thlpcart
  12.5000    -2.000     -10.000   -6.250000e-05   0.00e+00   0.00e+00   0.00e+00   0.00e+00
  37.5000    -2.000     -10.000   -1.875000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
  62.5000    -2.000     -10.000   -3.125000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
  87.5000    -2.000     -10.000   -4.375000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 112.5000    -2.000     -10.000   -5.625000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 137.5000    -2.000     -10.000   -6.875000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 162.5000    -2.000     -10.000   -8.125000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 187.5000    -2.000     -10.000   -9.375000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 212.5000    -2.000     -10.000   -1.062500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 237.5000    -2.000     -10.000   -1.187500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 262.5000    -2.000     -10.000   -1.312500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 287.5000    -2.000     -10.000   -1.437500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 312.5000    -2.000     -10.000   -1.562500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 337.5000    -2.000     -10.000   -1.687500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 362.5000    -2.000     -10.000   -1.812500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 387.5000    -2.000     -10.000   -1.937500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 412.5000    -2.000     -10.000   -2.062500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 437.5000    -2.000     -10.000   -2.187500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 462.5000    -2.000     -10.000   -2.312500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 487.5000    -2.000     -10.000   -2.437500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 512.5000    -2.000     -10.000   -2.562500e-03   0.00e+00   0.00e+00   0.00e+00   0.00e+00
```

Columns two to eight contain the following quantities (from left to right):

- geostrophic wind in the x-direction in m s^{-1}, u_g
- geostrophic wind in the y-direction in m s^{-1}, v_g
- large scale subsidence in m s^{-1}, $w_{f,ls}$
- large scale inflow of moisture in the x-direction in kg kg^{-1} s^{-1}, $\frac{dq_t}{dx}$
- large scale inflow of moisture in the y-direction in kg kg^{-1} s^{-1}, $\frac{dq_t}{dy}$
- additional large scale moisture change in time in kg kg^{-1} s^{-1}, $\frac{dq_{t,ls}}{dt}$
- prescribed temperature tendency due to radiation in K s^{-1}

Normally, most of these forcings can be set to zero.

ls_flux.inp

In some cases, the large scale forcings need to be varied over time, e.g. when the subsidence at the start of a simulation is much larger than in the end. This is possible, but of course not only the initial forcing must be prescribed. Additional profiles can be 'fed' to the model by using the `ls_flux.inp` file. Furthermore, timeseries of different variables

can be prescribed. This file is only read (and thus required!) when *ltimedep* is set to .true. in `namoptions`.

The file has a somewhat different lay-out than the others. Like the input files above, it starts with header lines, three this time. The first block of input is made up of five columns separated by white spaces. These columns contain (from left to right):

- time in s, t
- liquid water potential temperature flux at time t at the surface in K m s^{-1}, $w\theta_{l,s}$
- total humidity flux at time t at the surface in kg m s^{-1}, $wq_{t,s}$
- liquid water potential temperature at the surface in K, $\theta_{l,s}$
- total humidity at the surface in kg kg^{-1}, $q_{t,s}$

The routine reads these values into arrays. Then, it interpolates linearly between these given values. Therefore, if a linearly increasing total humidity flux is required, it is sufficient to include only two lines in the first block. One line at time $t = 0$ and one at time $t = runtime$. By using more entries, complicated changes in time can be made. The last line that is read is the line at time $t \geq runtime$.
The second part of the file contains blocks of vertical profiles of large scale forcings. Each block is exactly equal in lay-out to the `lscale.inp` file. It contains the same information, but not necessarily at time $t = 0$. A profile can be given for every desired t. The model starts reading at a header line directly above the columns, which contains a # followed by a white space and a time in seconds.

Figure 3.3: An example of the first few lines of a `ls_flux.inp` input file. Here, the vertical spacing dz is 25 meters. The first level is at $dz/2 = 12.5$m. To keep the figure small, all vertical levels above 137.50m are truncated off.

```
ASTEX Flight A209 exp nr 000
time    wtsurf          wqsurf          thls            qts
[s]     [K m/s]         [kg m/s]        [K]             [kg/kg]
     0  0.010000        1.0000e-05      287.996900      0.00e+00
 36000  0.013000        1.8000e-05      291.104000      0.00e+00

Large scale forcing terms
height          ug      vg        wfls          dqtdx       dqtdy    dqtdtls    thlpcart
#         0
  12.5000   -2.000   -10.000   -6.250000e-05   0.00e+00   0.00e+00   0.00e+00   0.00e+00
  37.5000   -2.000   -10.000   -1.875000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
  62.5000   -2.000   -10.000   -3.125000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
  87.5000   -2.000   -10.000   -4.375000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 112.5000   -2.000   -10.000   -5.625000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 137.5000   -2.000   -10.000   -6.875000e-04   0.00e+00   0.00e+00   0.00e+00   0.00e+00
-------------------------------------------------------------------------------------------
height          ug      vg        wfls          dqtdx       dqtdy    dqtdtls    thlpcart
#        36000
  12.5000   -3.000   -10.000   0.000000e+00   0.00e+00   0.00e+00   0.00e+00   0.00e+00
  37.5000   -3.000   -10.000   0.000000e+00   0.00e+00   0.00e+00   0.00e+00   0.00e+00
  62.5000   -3.000   -10.000   0.000000e+00   0.00e+00   0.00e+00   0.00e+00   0.00e+00
  87.5000   -3.000   -10.000   0.000000e+00   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 112.5000   -3.000   -10.000   0.000000e+00   0.00e+00   0.00e+00   0.00e+00   0.00e+00
 137.5000   -3.000   -10.000   0.000000e+00   0.00e+00   0.00e+00   0.00e+00   0.00e+00
```

The model interpolates the profiles between the different times. So for a linear increase/decrease in time, only two blocks need to be present, one block at time $t = 0$ and

one at $t = runtime$. The lines between the end of a block and the line containing the #, are not used. These lines can contain column information for instance. Again, the last block the model reads is that at time $t \geq runtime$.

scalar.inp

It is possible to include scalars in the model, such as concentrations of CO_2 or pollution. These scalars are so-called passive scalars, i.e. they do not influence the model. There is only one-way coupling between the model and the scalars. Initial vertical profiles of these quantities must be included in the input. This is done by using the `scalar.inp` file. This file is only read (and thus required!) if, in `namoptions`, $nsv \neq 0$ (see table 3.1).
The file consists of two unimportant header lines, followed by $nsv + 1$ columns of $kmax$ entries. The first of these columns is as usual the height, the vertical half levels of the domain. The next columns are the vertical profiles of the scalars.

Figure 3.4: An example of the first few lines of a `scalar.inp` input file. Here, the vertical spacing dz is 25 meters. The first level is at $dz/2 = 12.5$m. This example gives the necessary scalar fields for the microphysics routine, the initial raindrop (number) concentration N_r and the rainwater specific humidity q_r (section 4.1).

```
ASTEX Flight AZ09 exp nr000
height          Nr              qr
  12.5000       0.00e+00        0.00e+00
  37.5000       0.00e+00        0.00e+00
  62.5000       0.00e+00        0.00e+00
  87.5000       0.00e+00        0.00e+00
 112.5000       0.00e+00        0.00e+00
 137.5000       0.00e+00        0.00e+00
 162.5000       0.00e+00        0.00e+00
 187.5000       0.00e+00        0.00e+00
 212.5000       0.00e+00        0.00e+00
```

ls_fluxsv.inp

When both time dependence is used ($ltimedep = .true.$) and scalars are included ($nsv \neq 0$), the model requires extra information about the timedependence of the scalar profiles. This information is supplied by the ls_flux.inp file. This file has a similar build up as the ls_flux.inp file. It starts with three unimportant header lines.
The first block of data contains in the first column again *time*. The next nsv columns contain the surface fluxes of the different scalars at the specified time. The model interpolates these surface fluxes linearly between these times, therefore, when a linear time profile is needed, only two lines have to be included. One for the initial surface fluxes and one for the fluxes at *runtime*. The last line the model reads is where $t \geq runtime$. The next block starts with a #, followed by white space and a time. It contains the forcings for each scalar species at the specified time. Only one profile is read per scalar. That means that one block consists of $nsv+1$ columns, starting with *height*, the vertical

half levels of the domain. As many blocks can be included as required. The last block the model reads is the one at $t \geq runtime$.
NB. Up untill now, the large scale forcings of the scalar concentrations are not used. Including these profiles (these can be equal to zero everywhere) is required however, the model will stop running otherwise.

3.4 Output files

The filenames of the files generated by the model all end with a dot followed by the number of the experiment, *iexpnr* (table 3.1). In the standard model, no output data is generated. Only one file is created, namely `output`, which displays the options and initial profiles that are used by the model in that run. Therefore, it can be used as a check, to see whether or not the used input (`namoptions`, `prof.inp` etc.) is read correctly. Also, input/code related warnings are shown in this file.
A large part of the DALES code is devoted to the calculation of useful data and writing it to different output files. The code is divided over several so-called statistical routines and can be activated by adding options in `namoptions` (see chapter 4).

3.5 Run script

Of course, just compiling the source code of DALES is not enough to make it run. A run file (script) has to be used to call the model en perform some other tasks. In this section, a short overview of run scripts is given for Nemo, the computational grid of the TUDelft and for Huygens on the Dutch supercomputer SARA in Amsterdam.

Nemo

Figure 3.5 gives the general setup for the script that is needed to run the model on Nemo. Lines 5-8 are not necessary when the input files (here `prof.inp`, `lscale.inp` and `namoptions`), the model (`dales3_1`) and the run script are all located in the same

Figure 3.5: An example of a run script for Nemo, the computational grid of the TUDelft. The numbers at the beginning of each line are line numbers and should not be included in the real script.

```
1  #!/bin/bash
2  #
3  #$-pe mpi 4
4
5  cp ... prof.inp.001
6  cp ...  lscale.inp.001
7  cp ... namoptions
8  cp ...  dales3_1
9
10 . /etc/profile
11
12 mpiexec -n $NSLOTS ./dales3_1
13
14 exit 0
```

directory. Usually, it is more convenient to locate the input files in another directory and possibly with a different name. The copy commands copy all necessary files to the same directory and at the same time give them the appropriate name. Relative paths can be used in these copy commands (for instance `./Input/E001/namoptions.001`) or absolute paths (`~/Les/Input/E001/namoptions.001`). Of course, when extra input files are needed, these need also be copied in a similar way to the active directory. It is important that the destination file has the correct name, ending with a dot and the experiment number.

Huygens

Figure 3.6 gives the general setup for the script that is needed to run the model on Huygens. Lines 44 to 47 give the copy commands. These are exactly equal to the commands in the run file for Nemo. In the lines below, a temporary directory is created, where all output files are stored for the duration of the run. The actual model is called in line 50. Afterwards, a directory is created whereto all init files are copied.

3.6 Warm start

Sometimes, it can be convenient to continue a certain simulation after it has already ended. This is possible with DALES. There are some options in `namoptions` that are of importance in this context (see section 3.7).
First of all, apropriate initial conditions have to be selected. Luckily, DALES saves profiles once in a certain time. The time interval at which DALES does this is *trestart*. By default, it happens once in a simulated hour. The paralellization of the model requires a restart file for each CPU.
NB. The restart files can take up quite some disk space, up to several megabytes for each file. Therefore, care should be taken in setting a small value for *trestart*, especially when using many CPUs.
DALES automatically assigns names to the restart files. They are as follows: `initd00-hxxmxxx.[iexpnr]`. The extension is as with all output files the experiment number. The double x in the file name is the number of the restart file, the triple x gives the CPU ID. For example, if $trestart = 3600$ and four CPUs are being used, then after two simulated hours, eight restart files will have been written, namely:

`initd00h01m001.000`	`initd00h02m001.000`
`initd00h01m002.000`	`initd00h02m002.000`
`initd00h01m003.000`	`initd00h02m003.000`
`initd00h01m004.000`	`initd00h02m004.000`

To continue the simulation, starting after two hours, the appropriate restart files must be copied to the directory of the executable (or, at least, the run script should copy them to the right directory). The $startfile$ option should be: `initd00h02mxxx.000` and

lwarmstart = *.true.* The model automatically changes the triple x after the 'm' (places 12 to 14) to the CPU ID.
All output now starts at $t = 7200s$. Therefore special care should be taken with *runtime*. This is, as the name already says, not the final time of the simulation, but the time the simulation will run. If a simulation is continued after 7200 seconds, and the runtime

Figure 3.6: An example of a run script for Huygens, the computational grid on the SARA, the supercomputer in Amsterdam.

```
#!/bin/bash
# Loadleveler directives start with # @
#
# In this job an MPI program is started.
# This job will run on two nodes, 16 processes on each node,
# making a total of 32 processes.
# @ job_name = 000.
# @ node = 1
# @ tasks_per_node = 32
#
# Loadleveler can send email, but for this job, we ask Loadleveler not
#      to send any email:
# @ notification = never
#
# Define the standard input, output and error for the job:
# @ input = /dev/null
# @ output = errorfile.$(job_name)
# @ error = errorfile.$(job_name)
#
# Job will take at most 10 hour wallclock time
# @ wall_clock_limit = 10:00:00
#
# Tell Loadleveler that this is a parallel job. Loadleveler will take
# care of defining the proper environment for the job
# @ job_type = parallel
#
# The following line ensures that the communication between nodes will
# use the infiniband hardware:
# @ network.MPI = sn_all,not_shared,US
#
# This ends the instructions for Loadleveler:
# @ queue
#
# Here the shell script starts.
# go to the working directory:
/bin/bash
cd $HOME/Les/Output/H000
workdir=`pwd`
partition=`uname -n`
TMPDIR=`echo $TMPDIR | sed "s/scratch\/local/gpfs\/mcn1\/${partition}/g"`
rm -f TMPDIR.$LOADL_JOB_NAME
ln -fs $TMPDIR TMPDIR.$LOADL_JOB_NAME

cp .../dales3_1 $TMPDIR/dales3_1
cp ... $TMPDIR/namoptions
cp ... $TMPDIR/prof.inp.000
cp ... $TMPDIR/lscale.inp.000

cd $TMPDIR
./dales3_astex >output.000
mv * $workdir
cd $workdir
mkdir INIT
mv init* INIT/.
```

is also 7200 seconds, the end time of the simulation will be 14400 seconds, or, 4 hours. This is particularly important when also working with time dependent input.

3.7 List of options

This section contains a list of all options that can be set in the standard model. There are other options in other namelists that can also be usefull. These can be found in chapter 4.

Table 3.2: All options with their default values, possible values, a description and the unit, for all namelists in `namoptions`. Namelists that are specific for statistical routines or add-ons are not included (see chapter 4).

Option	Default	Possible values	Description	Unit
RUN				
iexpnr	000	any positive integer	Experiment number; every output filename ends with [.iexpnr]	-
dtmax	20	any positive real	Maximum timestep that is used by the model	s
runtime	300	any positive real	Total simulation (or: run) time	s
lwarmstart	.false.	.false./.true.	Flag for a "cold" or a "warm" start	-
startfile	-	see section 3.6	Basis for the name of the restartfiles	-
trestart	3600	any positive real	Each *trestart* seconds, a restart file is written to disk	s
dtav_glob	60	any positive real	Global value for sampling interval of statistical routines	s
timeav_glob	3600	integer multiple of *dtav_glob*	Global value for writing interval of statistical routines	s
irandom	0	any integer	Number to feed randomnizer with	-
krand	kmax	any positive integer	Top vertical full level of randomnization	-
randqt	1e-5	any positive real	Amplitude of randomnization of qt	kg kg^{-1}
randthl	0.1	any positive real	Amplitude of randomnization of thl	K

continued on next page

nsv	0	0-100, integer	Number of additional passive scalars	-
ladaptive	.false.	.false./.true.	If .true., this allows the model to vary time step, depending on numerical stability criteria	-
courant	1.4 or 3	any positive real	Courant number	-
peclet	0.2	any positive real	Peclet number	-
DOMAIN				
imax	64	any positive integer	Number of horizontal grid points in x-direction	-
jtot	64	any integer multiple of number of processors	Number of horizontal grid points in y-direction	-
kmax	96	any positive integer	Number of vertical grid points	-
xsize	-	any positive real	Horizontal size of the simulated domain	m
ysize	-	any positive real	Horizontal size of the simulated domain	m
xlat	52	-90 - +90	Latitude	$^\circ$
xlon	0	0 - 360	Longitude	$^\circ$
xday	1	1 - 365	Number of the day	-
xtime	0	0:00 - 24:00	UTC time of the day	h
ksp	$\min(\frac{3}{4}\text{kmax}, \text{kmax - 15})$	0 - (kmax - 15)	Lower height of sponge layer	-
PHYSICS				
thls	-	any positive real	Liquid water potential temperature at the surface	K
ps	-	any positive real	Pressure at the surface	Pa
isurf	-	1,2,3,4	Flag for surface parametrization	-
z0	-	any positive real	Surface roughness	m
ustin	-	any real	Prescribed friction velocity	m s^{-1}
wtsurf	-	any positive real	Flux of liq. water pot. temp. at the surface	K m s^{-1}
wqsurf	-	any positive real	Flux of total water content	kg kg^{-1}
wsvsurf(1:nsv)	0	any real	Flux of scalar n at the surface	-

continued on next page

ltimedep	.false.	.false./.true.	Switch for timedependent fluxes and large scale forcings	-
lcoriol	.true.	.false./.true.	Switch for coriolis force	-
lmoist	.true.	.false./.true.	Switch for calculation of moisture fields	-
lneutraldrag	.false.	.false./.true.	Switch for stability correction	-
chi_half	0.5	0 - 1	Fraction determining if a dry or a moist θ_v gradient is used to compute the subgrid bouyancy flux at a cloud interface	-
iradiation	2	0, 1, 2, 10	Flag for radiation calculations 0 = No radiation 1 = Full radiation (not implemented yet) 2 = Parametrized radiation 10 = User defined radiation (use `rad_user.f90`)	-
timerad	0	any positive real	Value for sampling interval of radiation scheme	s
rad_ls	.true.	.false./.true.	Switch for prescribed radiative forcing	-
rad_longw	.true.	.false./.true.	Switch for parametrized longwave radiative forcing	-
rad_shortw	.true.	.false./.true.	Switch for parametrized shortwave radiative forcing	-
rad_smoke	.false.	.false./.true.	Switch for longwave divergence for smoke cloud	-
rka	130	any positive real	Extinction coefficient (used if $iradiation = 2$)	$m^2\ kg^{-1}$
dlwbot	0	any positive real	Longwave radiative flux jump at cloud bottom	$W\ m^{-2}$
dlwtop	74	any positive real	Longwave radiative flux jump at cloud top	$W\ m^{-2}$

continued on next page

sw0	1100	any positive real	Direct solar radiative component cloud top (assumes zero diffusive contribution)	W m^{-2}
gc	0.85	0 - 1	Asymmetry factor of droplet scattering angle distribution	-
sfc_albedo	0.05	0 - 1	Surface albedo	-
reff	1e-5	any positive real	Cloud drop effective radius	m
isvsmoke	1	positive integer $\leq$ nsv	Number of passive scalar fields to be used for optical depth calculation (not used when $rad\_smoke = .false.$)	-
DYNAMICS				
cu	0	any real	Transformation velocity of the Galilei transformation in x-direction	m s^{-1}
cv	0	any real	Transformation velocity of the Galilei transformation in y-direction	m s^{-1}
llsadv	.false.	.false./.true.	Switch for large scale forcings	-
lqlnr	.false.	.false./.true.	Switch for Newton-Raphson approximation of the liquid water content	-
iadv_mom iadv_tke iadv_thl iadv_qt iadv_sv	5	1,2,4,5,6,7	Advection scheme for momentum, TKE, θ_l, q_t and scalars: 1 = 1st order upwind 2 = 2nd order central difference 4 = 4th order central difference 5 = 5th order central difference 6 = 6th order central difference 7 = kappa scheme	-
SUBGRID				
ldelta	.false.	.false./.true.	Switch for diminished sfs in stable flow	-
lmason	.false.	.false./.true.	Switch for decreased length scale near the surface	-

continued on next page

cf	2.5	any positive real	Filter constant	-
cn	0.76	any positive real	Subfilter scale parameter	-
Rigc	0.25	any positive real	Critical Richardson number	-
Prandtl	3	any positive real	Prandtl number	-

Chapter 4

Statistical Routines and Add-Ons

The DALES code is highly modular. It consists of only a few modules, which in turn consist of a few subroutines per module. The core program makes calls to the subroutines to execute them. This helps to keep the model fast and tidy. Parts of the model can easily be deactivated. This way, everyone can adjust it to his or her purposes. This deactivating can be done by removing the calls inside the main time loop in the code.
An easier way, without having to change the code, is by using namelists. Some modules have their own namelist. These modules can be quite useful to some, while being completely unnecessary to others. The namelist of a seperate module contains the options specific for that module, including a switch to activate it. By default, these switches are false. Therefore, namelists of unused modules need not be included in the namoptions file.
A distiction can be made in the modules that are not part of the core. There are modules that have a one way interaction with the model. These modules use information of the model, but do not influence the simulation. These modules are called Statistical Routines. What these routines usually do is create data files with time-averaged output. They have a time sampling interval and a time averaging/writing interval. After every sampling interval, the instantaneous values of certain quantities are taken from the model and, if necessary, calculations are done with them. After a writing interval, these values are averaged and written to a data file. Most of the statistical routines do not write entire 3D fields, but average over the domain, which results in average vertical profiles:

$$\overline{\psi}(z) = \frac{\sum_{n=1}^{N} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \psi(x,y,z,t)}{N N_x N_y} \tag{4.1}$$

where

$$N = \frac{time\_av}{dtav}. \tag{4.2}$$

Preferably, the sampling and averaging time intervals are by default equal to the global values, which can be set in `namoptions` (see table 3.2). The namelist of the module may contain options to change these intervals to different values.

Other modules incorporate physical processes into the model. This means that there can be two-way coupling between the module and the model. This does not necessarily be the case however. The particles add-on, for instance, that is discussed below does not influence fields of q_t or temperature, but it still is an add-on because it incorporates the physical process of particle advection. A list of all add-ons that are already available is given in the next section.
The general idea is that the code required to include an extra module into the model is kept at a minimum. Those present now only require a use statement in `program.f90` together with three calls: the first call is to the init subroutine in the module, the second calls the execution subroutine while the third finalizes the module by calling the exit subroutine of the module (if necessary). This way, the original code is hardly altered by the addition of the extra module. To ensure that variable or subroutine names are not interfering, a 'private' statement should be included.

4.1 Add-Ons

Below, a short description of the available add-on modules is given, together with a list of the associated options.

modmicrophysics

This module adds microphysical schemes to the code. The available options are summarized in table 4.1.
Up until now the only microphysics scheme is the bulk microphysics scheme. On the list for later versions is a (simpler) drizzle routine and an option for user specified microphysics. The options below the microphysics flag in table 4.1 are only used in the bulk microphysics routine. The bulk microphysics scheme uses two scalars, so when using this scheme, nsv should be at least two. The first scalar represents the particle number concentration N_r, while the second is the rainwater specific humidity q_r. The output that is produced by this module are the files associated with the scalar fields. For each scalar a field and a flux file are generated. More output can be generated by activating the special bulk microphysics statistical routine. More information on this scheme can be found under modbulkmicrostat in the next section.
The shape parameter μ_r is used to describe the rain drop number density distribution in the microphysics parameterization in the following way (Stevens and Seifert, 2008):

$$f_r(D) = n_r \left(\frac{D}{D_p}\right)_r^{\mu} \frac{exp(-D/D_p)}{D_p \Gamma(\mu_r + 1)}. \tag{4.3}$$

In this equation n_r is the rain-drop number mixing ratio, μ is the shape parameter, D_p is a size parameter and Γ is the Gamma function.

Table 4.1: Options associated with the microphysics add-on with the default value, possible values, a short description and the unit of the given value.

NAMMICROPHYSICS				
imicro	0	0,2	Flag for the microphysical scheme: 0 = No microphysics (all-or-nothing scheme) 2 = Bulk microphysics	-
l_sb	.true.	.false./.true.	Switch for KK00 (Khairoutdinov and Kogan, 2000) or SB (Seifert and Beheng, 2001, 2006) scheme resp.	-
l_sedc	.true.	.false./.true.	Switch for cloud droplet sedimentation	-
l_rain	.true.	.false./.true.	Switch for rain formation and evolution	-
l_mur_cst	.false.	.false./.true.	Switch for a constant value of μ_r (in raindrop gamma distribution)	-
mur_cst	5	any positive real	Value for μ_r, a shape parameter for the rain drop number density distribution (used only if l_mur_cst = .true.)	-
Nc_0	-	any positive real	Initial number of cloud droplets	-
sig_g	-	any positive real	Geometric standard deviation of the cloud droplet drop size distribution	-
sig_gr	-	any positive real	Geometric standard deviation of the rain droplet drop size distribution	-

modnudge

With this module, θ_l en q_t can be kept at their initial profiles. This is achieved by exerting a source term: $-\frac{\overline{\phi}-\overline{\phi}_{init}}{\tau}$. The only options available for this module is *tnudge* (τ, the timescale of the source term), which has a default value of 0. This causes the module not to be used. Setting a different value under the namelist NAMNUDGE automatically activates the module.

modparticles

The particles module enables one to add massless tracer particles to the simulation. It calculates the Lagrangian tracks of these particles according to a user-specifiable time integration scheme. Unlike the bulk microphysics module, which has a separate statistics module, the particles module has only one namelist where statistics can be switched on and off. All available options for this module are located in table 4.2.
To use the particles routine, an extra input file needs to be included. The name of this input file has to be `partstartpos.[iexpnr]`. The total number of particles must be on the first line of the file. The rest of the file has the initial conditions of each particle, one line per particle. The first column is the time t in seconds, the second to fourth column are the physical coordinates of the particles: x, y, z respectively.

Table 4.2: Options associated with the particles add-on with the default value, possible values, a short description and the unit of the given value.

NAMPARTICLES				
lpartic	.false.	.false./.true.	Switch to enable/disable this routine	-
lpartsgs	.true.	.false./.true.	Switch for subgrid diffusion	-
intmeth	3	0,3	Flag for time integration scheme 0 = particles stand still 3 = Adams-Bashfort second order scheme	-
lstat	.false.	.false./.true.	Switch for particle statistics	-
dtav	60	any positive real	Time interval for sampling of statistics	s
timeav	3600	integer multiple of dtav	Time interval for writing statistics	s
ldump	.false.	.false./.true.	Switch for dump of particle field	-
timedump	3600	any positive real	Time interval for particle field dump	s
npartdump	10	0-10	Number of variables written at *timedump*, in order: x, y, z, u, v, w, θ_l, θ_v, q_t, q_l	-

The output of this routine can consist of two files: `partstat` and `particles`. The first of these is only created if *lstat* = .true. and it contains the statistics on the particles, like averaged vertical profiles of resolved and subgrid scale velocities and turbulent kinetic energy. The latter file is only created if *ldump* = .true. and it contains instantanious information of the particles.

modtilt

By activating this module, one can create a tilted boundary layer. This can be used to simulate katabatic flow. The options that are available in the namelist associated with this add-on are located in table 4.3.

Table 4.3: Options associated with the tilt add-on with the default value, possible values, a short description and the unit of the given value.

NAMTILT				
ltilted	.false.	.false./.true.	Switch for a tilted boundary layer	-
alfa	0	$-\frac{\pi}{2}$-$+\frac{\pi}{2}$	Tilt angle	rad
lstat	.true.	.false./.true.	Switch for statistics	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
timeav	timeav_glob	integer multiple of dtav	Time interval for writing statistics	s

4.2 Statistical Routines

In this section, a short description of all available statistical routines is given, with their associated namelist.

modbudget

The budget module calculates the terms of the TKE equation: buoyancy, shear, transport etc. terms. Two output files are created, namely `budget` and `sbbudget`. The first contains vertical profiles of the resolved TKE terms, while the latter contains the subgrid TKE terms. The options that are available for this routine are given in table 4.4.

Table 4.4: Options associated with the budget statistical routine with the default value, possible values, a short description and the unit of the given value.

NAMBUDGET				
lbudget	.false.	.false./.true.	Switch for turbulent TKE budget calculation	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
timeav	timeav_glob	integer multiple of dtav	Time interval for writing statistics	s

modbulkmicrostat

This routine can only be activated if the bulk microphysics add-on is used. It calculates various tendencies and statistics regarding the number of rain and cloud drops, the amount of liquid water in cloud or rain droplets, etc. The output is located in four different files: `precep`, `nptend`, `qltend` and `qttend`. Table 4.5 contains the options of the namelist associated with this routine.

Table 4.5: Options associated with the bulkmicrostat routine with the default value, possible values, a short description and the unit of the given value.

NAMBULKMICROSTAT				
lmicrostat	.false.	.false./.true.	Switch for microphysics statistics calculation	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
timeav	timeav_glob	integer multiple of dtav	Time interval for writing statistics	s

modchecksim

This routine calculates the divergence $\overline{\nabla \overline{u}}$ and the Courant numbers for the different directions $u_i \frac{\Delta t}{\Delta x}$. Some of these values are written to the `output` file (the maximum and

the total divergence, the maximum values of the Courant number in all three directions). The associated namelist is NAMCHECKSIM, the only option contained in this namelist is *tcheck*, which gives the interval between checks. The default value is 3600 s.

modcloudfield

The cloudfield statistical routine dumps all the wet points (points were $q_l >$ 1e-5 kg kg^{-1} of the domain into the `clouds` file. An additional switch controls the output of q_l and w values. All available options are summarized in table 4.6.

Table 4.6: Options associated with the cloud field statistical routine with the default value, possible values, a short description and the unit of the given value.

NAMCLOUDFIELD				
lcloudfield	.false.	.false./.true.	Switch for cloud field calculations	-
laddinfo	.false.	.false./.true.	Switch to enable writing of q_l and w values	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s

modcrosssection

This statistical routine dumps instantaneous cross sections of the field. For every variable (u, v, w, θ_l, θ_v, q_t, q_l), a new file is created for the horizontal and for the vertical crosssection. The available options are given in table 4.7.

Table 4.7: Options associated with the crosssection routine with the default value, possible values, a short description and the unit of the given value.

NAMCROSSSECTION				
lcross	.false.	.false./.true.	Switch for dumping of crosssections of the field	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
crossheight	2	1 - $kmax$, integer	Height of the horizontal crosssection	-
crossplane	2	1- $xmax$, integer	Location of the vertical (yz) plane on every processor	-

modfielddump

When this routine is enabled, at regular time intervals, a dump is made of the entire 3d-fields of u, v, w, θ_l, θ_v, q_t, q_l in 2-byte integers. This module's options and the name of the namelist is given in table 4.8

Table 4.8: Options associated with the field dump routine with the default value, possible values, a short description and the unit of the given value.

NAMFIELDDUMP				
lfielddump	.false.	.false./.true.	Switch for dumping of 3d-fields	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
ldiracc	.false.	.false./.true.	Switch to dump into direct access files instead of Fortran unformatted files	-
klow	1	1 - *khigh*	Lowest level of the 3d-field output	-
khigh	*kmax*	*klow* - kmax	Highest level of the 3d-field output	-

modgenstat

This statistical routine calculates generic slabaveraged statistics, i.e. time averaged fields of the most used variables (T, θ, q_t etc.), time averaged tendencies of most of these variables, time average turbulent fluxes, variances and skewness of w and q_t. This output is divided over four files: `field`, `flux1`, `flux2` and `moments`. Furthermore, if $nsv \neq 0$, for each scalar a field and a flux file is created. The options are the usual and can be found in table 4.9

Table 4.9: Options associated with the genstat routine with the default value, possible values, a short description and the unit of the given value.

NAMGENSTAT				
lstat	.false.	.false./.true.	Switch for calculating generic slabaveraged statistics	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
timeav	timeav_glob	integer multiple of dtav	Time interval for writing statistics	s

modradstat

This statistical routine calculates slabaveraged vertical profiles of variables concerning radiation, e.g. longwave radiation going up and going down and tendencies of θ_l due to short- and longwave radiation. The file that is created to contain the output is `radstat`. The available options for this module are given in table 4.10.

modsampling

With this module, conditionally sampled fields can be calculated. The conditions are imposed by switches. For each of these switches that is enabled, two files are created: a field file and a flux file. The names of the available options for this module is given in table 4.11.

Table 4.10: Options associated with the radstat routine with the default value, possible values, a short description and the unit of the given value.

NAMRADSTAT				
lradstat	.false.	.false./.true.	Switch for calculating slabaveraged radiation statistics	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
timeav	timeav_glob	integer multiple of dtav	Time interval for writing statistics	s

Table 4.11: Options associated with the sampling statistics routine with the default value, possible values, a short description and the unit of the given value.

NAMSAMPLING				
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
timeav	timeav_glob	integer multiple of dtav	Time interval for writing statistics	s
lsampcl	.false.	.false./.true.	Switch for conditional sampling cloud ($q_l > 0$)	-
lsampco	.false.	.false./.true.	Switch for conditional sampling cloud core ($q_l > 0, \theta_v' > 0$)	-
lsampup	.false.	.false./.true.	Switch for conditional sampling updrafts ($w > 0$)	-
lsampbuup	.false.	.false./.true.	Switch for conditional sampling of buoyant updrafts ($w > 0, \theta_v' > 0$)	-

modstattend

The modstattend calculates tendencies of various prognostic variables. It generates five different files: `utend`, `vtend`, `wtend`, `tltend` and `qttend`. The options are the usual and can be found in table 4.12

Table 4.12: Options associated with the sampling statistics routine with the default value, possible values, a short description and the unit of the given value.

NAMSTATTEND				
lstattend	.false.	.false./.true.	Switch for calculation of tendencies of prognostic variables	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
timeav	timeav_glob	integer multiple of dtav	Time interval for writing statistics	s

modtimestat

This module calculates timeseries of several variables, among others: cloud cover, cloud bottom and top, inversion height and mean liquid water path. These timeseries can be found in the output file `tmser1`. The module also calculates several surface variables, like surface temperature and surface fluxes. These timeseries are found in a second file: `tmsurf`. Several methods are present in the code to calculate the inversion height z_i, which can be selected by means of options. For all options, see table 4.13.

Table 4.13: Options associated with the time series statistics routine with the default value, possible values, a short description and the unit of the given value.

NAMTIMESTAT				
ltimestat	.false.	.false./.true.	Switch for calculation of time series	-
dtav	dtav_glob	any positive real	Time interval for sampling of statistics	s
timeav	timeav_glob	integer multiple of dtav	Time interval for writing statistics	s
iblh_var	iblh_thv	iblh_thv iblh_thl iblh_qt	Flag for the variable used to calculate boundary layer height iblh_thv = virtual pot. temp. θ_v iblh_thl = liquid water pot. temp. θ_l iblh_qt = total humidity q_t	-
iblh_meth	iblh_grad	iblh_flux iblh_grad iblh_thres	Flag for the method used to calculate boundary layer height iblh_flux = use flux of selected variable iblh_grad = use gradient of selected variable iblh_thres = use a threshold (auto or user specified)	-
blh_thres	automatic	any positive real	Threshold for the selected variable, used only for iblh_thres method	K or kg kg^{-1}
blh_nsamp	32	any positive integer	Number of levels to integrate over	-

Chapter 5

Model Theory

5.1 Governing Equations

After replacement of the conventional temperature in the Navier-Stokes equation with the virtual potential temperature θ_v, the Boussinesq approximation can be assumed within the ABL. This results (after application of the LES filter) in:

$$\frac{\partial \widetilde{u_i}}{\partial x_i} = 0, \tag{5.1}$$

$$\frac{\partial \widetilde{u_i}}{\partial t} = -\frac{\partial \widetilde{u_i}\widetilde{u_j}}{\partial x_j} - \frac{\partial \pi}{\partial x_i} + \frac{g}{\theta_0}(\widetilde{\theta_v} - \theta_0)\delta_{i3} - 2\epsilon_{ijk}\Omega_j\widetilde{u_k} + F_i - \frac{\partial \tau_{ij}}{\partial x_j}, \tag{5.2}$$

$$\frac{\partial \widetilde{\theta_l}}{\partial t} = -\frac{\partial \widetilde{u_j}\widetilde{\theta_l}}{\partial x_j} + S_{\theta_l} - \frac{\partial R_{u_j,\theta_l}}{\partial x_j}, \tag{5.3}$$

$$\frac{\partial \widetilde{q_t}}{\partial t} = -\frac{\partial \widetilde{u_j}\widetilde{q_t}}{\partial x_j} + S_{q_t} - \frac{\partial R_{u_j,q_t}}{\partial x_j}, \tag{5.4}$$

as basic equations, where the tildes denotes the filtered mean variables. The $\hat{z}$ direction is taken to be vertical. θ_0 is the reference state potential temperature and $\vec{\Omega}$ is the earth's angular velocity. Viscous transport terms are neglected. F_i represents large scale forcings. With φ one of the scalars, the large scale source terms are S_φ and the residuals are denoted by $R_{u_j,\varphi} = \widetilde{u_j\varphi} - \widetilde{u_j}\widetilde{\varphi}$. The anisotropic subfilter-stress tensor is defined by

$$\tau_{ij} \equiv \widetilde{u_iu_j} - \widetilde{u_i}\widetilde{u_j} - \frac{2}{3}\delta_{ij}\widetilde{e}, \tag{5.5}$$

where $\widetilde{e} = \frac{1}{2}(\widetilde{u_iu_i} - \widetilde{u_i}\widetilde{u_i})$ is the filtered subfilter-scale turbulent kinetic energy (SFS-TKE). The isotropic part of the subfilter-scale flux is included in the modified pressure

$$\pi = \frac{1}{\rho_0}(\widetilde{p} - p_0) + \frac{2}{3}e, \tag{5.6}$$

which is determined by solving a Poisson equation for π

$$\begin{aligned}\nabla^2\pi &= \nabla\left(-\frac{\partial\widetilde{u}_i\widetilde{u}_j}{\partial x_j}+\frac{g}{\theta_0}(\widetilde{\theta_v}-\theta_0)\delta_{i3}\right.\\ &\left.-2\epsilon_{ijk}\Omega_j\widetilde{u_k}+F_i-\frac{\partial\tau_{ij}}{\partial x_j}\right).\end{aligned} \tag{5.7}$$

Since computations are performed in a double periodic domain, the Poisson equation is solved with a Fast Fourier transform in the homogenous directions, while for the $\hat{z}$ direction a tridiagonal system is solved.

5.2 Subfilter-Scale Model

The subfilter-scale stresses and residuals are modeled using one-and-a-half order closure (Schmidt and Schumann, 1989). The stress tensor and the subfilter-scale fluxes are modeled by:

$$\tau_{ij}=-K_m\left(\frac{\partial\widetilde{u}_i}{\partial x_j}+\frac{\partial\widetilde{u}_j}{\partial x_i}\right), \tag{5.8}$$

$$R_{u_j,\varphi}=-K_h\frac{\partial\widetilde{\varphi}}{\partial x_j}, \tag{5.9}$$

with the eddy diffusivity coefficients K_m and K_h a function of $\widetilde{e}$. The prognostic equation for e:

$$\frac{\partial\widetilde{e}}{\partial t} = -\widetilde{u}_j\frac{\partial\widetilde{e}}{\partial x_j}-R_{u_i,u_j}\frac{\partial\widetilde{u}_i}{\partial x_j}+\frac{g}{\theta_0}R_{w,\theta_v}-\frac{\partial R_{u_j,e}}{\partial x_j}-\frac{1}{\rho_0}\frac{\partial R_{u_j,p}}{\partial x_j}-\varepsilon, \tag{5.10}$$

with ε the dissipation. To close Equation 5.10, we need to parameterize all right hand side terms except the first one. The SFS-TKE production by shear (the second term) is closed with Equation 5.8. Following Deardorff (1980), we use for the third term, the production due to buoyancy:

$$\frac{g}{\theta_0}R_{w,\theta_v}=\frac{g}{\theta_0}\left(AR_{w,\theta_l}+BR_{w,q_t}\right), \tag{5.11}$$

with coefficients A and B depending on the local thermodynamic state:

$$\begin{cases} A=1 & B=\left(1-\frac{R_v}{R_d}\right)\theta_0 \quad \text{for } \widetilde{q}_t<\widetilde{q}_s \\ A=\frac{1+\frac{R_v}{R_d}\left(\theta_0\frac{d\widetilde{q_s}}{dT}\right)}{1+\frac{L}{c_{pd}}\frac{d\widetilde{q_s}}{dT}} & B=\frac{L}{c_{pd}}A-\theta_0 \quad \text{for } \widetilde{q}_t>\widetilde{q}_s \end{cases} \tag{5.12}$$

The fourth and fifth term in Equation 5.10 are together parameterized as:

$$-\frac{\partial}{\partial x_j}\left(R_{u_j,e}+\frac{1}{\rho_0}R_{u_j,p}\right)=2K_m\frac{\partial\widetilde{e}}{\partial x_j}, \tag{5.13}$$

Under the assumption of 3D homogenous isotropic turbulence, the dissipation rate ε can be found by integration of the energy spectrum $E(k) = \alpha\varepsilon^{2/3}k^{-5/3}$ from a filter wavenumber k_f, that lies within the inertial subrange, to infinity

$$\varepsilon = \widetilde{e}^{\,3/2}k_f\left(\frac{3}{2}\alpha\right)^{-3/2} = \frac{c_\varepsilon}{\lambda}\widetilde{e}^{\,3/2}, \quad \text{with} \quad c_\varepsilon = \frac{2\pi}{c_f}\left(\frac{3}{2}\alpha\right)^{-3/2} \tag{5.14}$$

with $\alpha = 1.5$ the Kolmogorov constant and λ the dominant length-scale of the unresolved flow: $\lambda = \frac{2\pi}{c_f k_f}$, where c_f is chosen to be 2.5 (see Cuijpers 1990). In unstable flow, λ can be taken equal to grid size $\Delta = (\Delta x\Delta y\Delta z)^{1/3}$, but this no longer holds for stable flow, where $\frac{\partial\widetilde{\theta_v}}{\partial z} > 0$. In that case, the length scale is taken to be

$$\lambda = \min\left(\Delta, c_N\frac{\widetilde{e}^{\,1/2}}{N}\right), \tag{5.15}$$

with $N^2 = \frac{g}{\theta_0}\frac{\partial\theta_v}{\partial z}$ the Brunt-Väisälä frequency.
The eddy diffusivity can be found by equating the dissipation ε to the production of SFS-TKE:

$$\mathcal{P} = 2K_m\int_0^{k_f} k^2E(k)dk \tag{5.16}$$

$$= \frac{3}{2}K_m\alpha\varepsilon^{2/3}k_f^{4/3}, \tag{5.17}$$

which yields:

$$K_m = \frac{\widetilde{e}^{\,1/2}}{k_f(\frac{3}{2}\alpha)^{3/2}} = c_m\lambda\widetilde{e}^{\,1/2} \quad \text{with} \quad c_m = \frac{c_f}{2\pi}\left(\frac{3}{2}\alpha\right)^{-3/2} \tag{5.18}$$

Similarly, $K_h = c_h\lambda\widetilde{e}^{\,1/2}$. Like for λ, a stability correction is also applied on c_h and c_ε

$$c_h = \left(c_{h,1} + c_{h,2}\frac{\lambda}{\Delta}\right)c_m, \tag{5.19}$$

$$c_\varepsilon = c_{\varepsilon,1} + c_{\varepsilon,2}\frac{\lambda}{\Delta}. \tag{5.20}$$

Now all parameters of the subfilter-scale parameterization of DALES are defined; they are summarized in Equation 5.2.
Filling the closure relations and parameters into Equation 5.10, we get:

$$\begin{aligned}\frac{\partial\widetilde{e}^{\,1/2}}{\partial t} &= -\widetilde{u_j}\frac{\partial\widetilde{e}^{\,1/2}}{\partial x_j} + \frac{1}{2\widetilde{e}^{\,1/2}}\left[K_m\left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j}\right)\frac{\partial u_i}{\partial x_j}\right.\\ &+ \left.-K_h\frac{g}{\theta_0}\frac{\partial A\widetilde{\theta_l} + Bq_t}{\partial z}\right] + \frac{\partial}{\partial x_j}\left(2K_m\frac{\partial\widetilde{e}^{\,1/2}}{\partial x_j}\right) - \frac{c_\varepsilon\widetilde{e}}{2\lambda},\end{aligned} \tag{5.21}$$

which closes the system.

α	c_f	$c_{\varepsilon,1}$	$c_{\varepsilon,2}$	c_m	$c_{h,1}$	$c_{h,2}$	c_N
1.5	2.5	0.19	0.51	0.12	1	2	0.76

Table 5.1: An overview of the parameters used in the SFS-scheme of DALES model. Not all are independent.

5.3 Discretization and Numerical Scheme

A rectangular grid is used, with optional grid stretching in the $\hat{z}$ direction. For clarity, an equidistant grid is assumed in the discussion of the advection scheme. The grid is staggered in space; if the center of a grid cell of the pressure, the SFS-TKE or the scalars is located at $\vec{x} + \frac{1}{2}(\Delta x, \Delta y, \Delta z)$, the $\widetilde{u}$-grid is centered at $\vec{x} + \frac{1}{2}(0, \Delta y, \Delta z)$, and similar for $\widetilde{v}$ and $\widetilde{w}$.
Time integration is done by a third order Runge-Kutta scheme following Wicker and Skamarock (2002). With $f^n(\phi^n)$ the right-hand side of the appropriate equation of 5.2-5.4 for variable $\phi = \{\widetilde{u}, \widetilde{v}, \widetilde{w}, \widetilde{\theta_l}, \widetilde{q_t}\}$, ϕ^{n+1} at $t + \Delta t$ is calculated in three steps:

$$\begin{aligned} \phi^* &= \phi^n + \frac{\Delta t}{3} f^n(\phi^n) \\ \phi^{**} &= \phi^* + \frac{\Delta t}{2} f^*(\phi^*) \\ \phi^{n+1} &= \phi^{**} + \Delta t f^{**}(\phi^{**}), \end{aligned} \tag{5.22}$$

with the asterisks denoting intermediate time steps.
Depending on the desired properties (like high accuracy or monotonicity), several advection schemes are available. With advection in the $\hat{x}$ direction discretized as

$$\frac{\partial \widetilde{u}_i \phi_i}{\partial x} = \frac{F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}}}{\Delta x}, \tag{5.23}$$

with $F_{i-\frac{1}{2}}$ is the convective flux of variable ϕ through the $i - \frac{1}{2}$ plane. Since we are using a staggered grid the velocity is available at $i - \frac{1}{2}$ without interpolation. Second order central differencing can be used for variables where neither very high accuracy or strict monotonicity is necessary:

$$F^{2nd}_{i-\frac{1}{2}} = \widetilde{u}_{i-\frac{1}{2}} \frac{\phi_i + \phi_{i-1}}{2}, \tag{5.24}$$

Higher order accuracy is reached with a sixth order central differencing scheme (see Wicker and Skamarock, 2002):

$$\begin{aligned} F^{6th}_{i-\frac{1}{2}} = \frac{\widetilde{u}_{i-\frac{1}{2}}}{60} &[37(\phi_i + \phi_{i-1}) \\ &-8(\phi_{i+1} + \phi_{i-2}) + (\phi_{i+2} + \phi_{i-3})]. \end{aligned} \tag{5.25}$$

By adding a small dissipative term to $F^{6th}_{i-\frac{1}{2}}$, a fifth order scheme is created that is nearly monotone:

$$\begin{aligned} F^{5th}_{i-\frac{1}{2}} &= F^{6th}_{i-\frac{1}{2}} - \left|\frac{\widetilde{u}_{i-\frac{1}{2}}}{60}\right| [10(\phi_i + \phi_{i-1}) \\ &\quad - 5(\phi_{i+1} + \phi_{i-2}) + (\phi_{i+2} + \phi_{i-3})] . \end{aligned} \tag{5.26}$$

For advection of scalars that need to be strictly monotone (for example chemically reacting scalars) the κ scheme Hundsdorfer et al. (1995) is implemented:

$$F^{\kappa}_{i-\frac{1}{2}} = \widetilde{u}_{i-\frac{1}{2}} \left[\phi_{i-1} + \frac{1}{2}\kappa_{i-\frac{1}{2}} (\phi_{i-1} - \phi_{i-2})\right], \tag{5.27}$$

in case $\widetilde{u} > 0$. $\kappa_{i-1/2}$ serves as a switch between higher order advection and first order upwind in case of strong upwind gradients of ϕ. This makes the scheme monotone, but also rather dissipative.

5.4 Condensation Scheme

To calculate the liquid water content $\widetilde{q_l}$ from pressure, temperature and total water content, a condensation scheme has to be included. In the model, it is assumed that there is no liquid water present in a unsaturated grid box, while all moisture above saturation value $\widetilde{q_s}$ is liquid water. To calculate $\widetilde{q_s}$ following Sommeria and Deardorff (1977), first we need to know the liquid water temperature $\widetilde{T_l}$:

$$\widetilde{T_l} = \left(\frac{p_{ref}}{p_0}\right)^{cp/R_d} \widetilde{\theta_l}, \tag{5.28}$$

with

$$p_{ref}(z) = p_0 \left(\frac{T_{ref}(0) - (g/c_p)z}{T_{ref}(0)}\right)^{cp/R_d}. \tag{5.29}$$

The saturation vapor pressure e_s is an empirical function of $\widetilde{T_l}$:

$$e_s(\widetilde{T_l}) = e_{s0} \exp\left[a \frac{\widetilde{T_l} - T_{trip}}{\widetilde{T_l} - b}\right], \tag{5.30}$$

with constants $e_{s0} = 610.78\,\mathrm{Pa}$, $T_{trip} = 273.16\,\mathrm{K}$, $a = 17.27$ and $b = 35.86$. For ideal gases, the specific humidity and vapor pressure are related by:

$$q_{sl} = \frac{R_d}{R_v} \frac{e_s}{p_{ref} - (1 - \frac{R_d}{R_v})e_s}. \tag{5.31}$$

Using the Clausius-Clapeyron relation:

$$\frac{\mathrm{d}e_s}{\mathrm{d}T} = \frac{L e_s}{R_v T^2}, \tag{5.32}$$

with $R_v = 461.53\,\mathrm{J\ kg^{-1}\ K^{-1}}$ the gas constant for water vapor, a truncated Taylor expansion now gives a relation for the saturated specific humidity:

$$q_s = q_{sl}\left(1 + \frac{L^2}{R_v c_{p,d} \widetilde{T_l}^2} q_t\right)\left(1 + \frac{L^2}{R_v c_{p,d} \widetilde{T_l}^2} q_{sl}\right), \tag{5.33}$$

and finally we obtain for the liquid water content.

$$q_l = \begin{cases} q_t - q_s & \text{if } q_t > q_s \\ 0 & \text{otherwise.} \end{cases} \tag{5.34}$$

5.5 Boundary Conditions

The computational domain has periodic boundary conditions in the horizontal directions. At the top of the domain, we take:

$$\frac{\partial \widetilde{u}}{\partial z} = \frac{\partial \widetilde{v}}{\partial z} = 0; \qquad \widetilde{w} = 0; \qquad \frac{\partial \widetilde{\theta_l}}{\partial z} = \text{constant}; \qquad \frac{\partial \widetilde{q_t}}{\partial z} = \text{constant}. \tag{5.35}$$

Horizontal fluctuations at the top of the domain (e.g. gravity waves) are damped out by a sponge layer through an additional forcing/source term:

$$\frac{1}{\tau_r}\left(\overline{\phi}(z) - \phi\right) \tag{5.36}$$

with $\overline{\phi}$ the slab averaged value and τ_r a relaxation time scale that goes from 1 at the top of the domain to infinity at the bottom of the sponge layer.

Surface Model

In order to calculate the subgrid fluxes (Equation 5.9), the vertical gradients of the variables have to be determined. In the mixed layer, the profiles of variables are assumed to be linear, so that they can be simply approximated by finite difference, for instance $\frac{\partial <u>}{\partial z} = \frac{\Delta <u>}{\Delta z}$. This is not applicable in the surface layer, where the profiles are characterized by large gradients.
In the surface layer, therefore, the Monin-Obukhov similarity theory is used.

Monin-Obukhov similarity theory

By defining the horizontal wind speed $V = \sqrt{< u >^2 + < v >^2}$, the general expression for the wind shear in neutral conditions is:

$$\frac{\partial V}{\partial z} = \frac{u_*}{kz} \tag{5.37}$$

where k is the von Karman constant ($k = 0.4$) and u_* the friction velocity. Integrating equation 5.37 leads to the logarithmic wind profile

$$V(z) = \frac{u_*}{k} ln \frac{z}{z_0}. \tag{5.38}$$

The Monin-Obukhov similarity theory assumes that the wind shear is scaled by the dimensionless height

$$\zeta \equiv z/L \tag{5.39}$$

where $L = \frac{-\theta_0 u_*^3}{g\overline{w'\theta'}}$ is the Monin-Obukhov length.
Expression 5.37 can be generalized for non-neutral stability conditions by introducing a dimensionless wind shear defined as follows:

$$\Phi_m(\zeta) = \frac{kz}{u_*}\frac{\partial V}{\partial z} \tag{5.40}$$

which leads to a general form for the wind shear:

$$\frac{\partial V}{\partial z} = \frac{u_*}{kz} - \frac{1-\Phi_m}{kz}u_* \tag{5.41}$$

Integrating equation 5.41 between z_0 (height at which $V(z_0) = 0$) and a generic height z, leads to:

$$V(z) = \frac{u_*}{k}\{ln\frac{z}{z_0} - \Psi_m(\frac{z}{L})\} \tag{5.42}$$

where

$$\Psi_m(\frac{z}{L}) = \int_0^{z/L} \frac{1-\Phi_m(\zeta)}{\zeta} d\zeta \tag{5.43}$$

represents the correction to the logarithmic profile due to non-neutral conditions ($\Psi_m = 0$ in neutral conditions).
Expressions fo $\Phi_m(\zeta)$ based on field campaigns (Businger et al., 1971; Dyer, 1974; Hogstrom, 1988) are commonly used.
Similarly to momentum, the similarity theory is used to calculate the profiles of the other variables, namely $< \theta >$ and $< q >$. By rewriting the expressions for the subgrid fluxes:

$$\begin{aligned} &< \theta'' w'' > = -K_h \tfrac{\partial <\theta>}{\partial z} = -u_*\theta_* \\ &< q'' w'' > = -K_q \tfrac{\partial <q>}{\partial z} = -u_* q_* \end{aligned} \tag{5.44}$$

and assuming (Pielke, 1984) $K_h = K_q$ we obtain an expression similar to (5.40):

$$\Phi_h = \frac{kz}{\theta_*}\frac{\partial < \theta >}{\partial z} = \frac{kz}{q_*}\frac{\partial < q >}{\partial z}, \tag{5.45}$$

which integrated leads to:

$$< \theta(z) > = < \theta(z_0) > + \frac{\theta_*}{k}\{\ln\frac{z}{z_0} - \Psi_h(\frac{z}{L}) + \Psi_h(\frac{z_0}{L})\} \tag{5.46}$$

$$< q(z) >=< q(z_0) > + \frac{q_*}{k}\{\ln\frac{z}{z_0} - \Psi_h(\frac{z}{L}) + \Psi_h(\frac{z_0}{L})\} \tag{5.47}$$

where

$$\Psi_h(\frac{z}{L}) = \int_0^{z/L} \frac{1-\Phi_h(\zeta)}{\zeta} d\zeta \tag{5.48}$$

Surface Layer relationships

The stability functions are parameterized following (Businger et al., 1971; Dyer, 1974; Hogstrom, 1988) as follows:

$$\Phi_m = \begin{cases} (1-16\zeta)^{-1/4} & \zeta < 0 \\ \\ 1+5\zeta & \zeta > 0 \end{cases}$$

$$\Phi_h = \begin{cases} (1-16\zeta)^{-1/2} & \zeta < 0 \\ \\ 1+5\zeta & \zeta > 0 \end{cases}$$

$$\Psi_m(\frac{z}{L}) = \begin{cases} 2\ln\left(\frac{1+\Phi_m^{-1}}{2}\right) + \ln\left(\frac{1+\Phi_m^{-2}}{2}\right) - 2\tan^{-1}(\Phi_m^{-1}) + \frac{\pi}{2} & \zeta < 0 \\ \\ -5\zeta & \zeta > 0 \end{cases}$$

$$\Psi_h(\frac{z}{L}) = \begin{cases} 2\ln\left(\frac{1+\Phi_h^{-1}}{2}\right) & \zeta < 0 \\ \\ -5\zeta & \zeta > 0 \end{cases}$$

The friction velocity is calculated from equation 5.42:

$$u_* = kV\left(\ln\frac{z}{z_0} - \Psi_m(\frac{z}{L})\right), \tag{5.49}$$

whereas θ_* and q_* are calculated from the known values of the surface fluxes:

$$\theta_* = -\frac{< w''\theta' >_0}{u_*} \tag{5.50}$$

$$\tag{5.51}$$

$$q_* = -\frac{< w''q'' >_0}{u_*} \tag{5.52}$$

$$\tag{5.53}$$

The temperature and humidity surface values are calculated according to equations 5.46 and 5.47

$$< \theta(z_0) >=< \theta(z) > -\frac{\theta_*}{k}\{\ln \frac{z}{z_0} - \Psi_h(\frac{z}{L}) + \Psi_h(\frac{z_0}{L})\} \tag{5.54}$$

$$< q(z_0) >=< q(z) > -\frac{q_*}{k}\{\ln \frac{z}{z_0} - \Psi_h(\frac{z}{L}) + \Psi_h(\frac{z_0}{L})\} \tag{5.55}$$

using the values of temperature and moisture at the first computational level ($z = 1$). Finally, the vertical gradients are calculated from equations 5.40 and 5.45 according to:

$$\frac{\partial V}{\partial z} = \Phi_m(\zeta)\frac{u_*}{kz} \tag{5.56}$$

$$\frac{\partial < \theta >}{\partial z} = \Phi_h(\zeta)\frac{\theta_*}{kz} \tag{5.57}$$

$$\frac{\partial < q >}{\partial z} = \Phi_h(\zeta)\frac{q_*}{kz} \tag{5.58}$$

In the limit of free convection ($u_* = 0$) the following relationships are used to calculate the temperature and moisture profiles (Garrat):

$$\frac{\partial < \theta >}{\partial z} = -0.7 < w''\theta'' >_0^{2/3} \left(\frac{g}{\theta_0}\right) z^{-4/3} \tag{5.59}$$

$$\frac{\partial < q >}{\partial z} = -0.7 < w''q'' >_0^{2/3} \left(\frac{g}{\theta_0}\right) z^{-4/3} \tag{5.60}$$

Bibliography

J.A. Businger, J.C. Wyngaard, Y. Izumi, and E.F. Bradley. Flux-profile relationships in the atmospheric surface layer. *Journal of the Atmospheric Sciences*, 28:181–189, 1971.

J.W. Deardorff. Cloud top entrainment instability. *Journal of the Atmospheric Sciences*, 37:131–147, 1980.

A.J. Dyer. A review of flux-profile relationships. *Boundary-Layer Meteorology*, 7:363–372, 1974.

U. Hogstrom. Non-dimensional wind and temperature profiles in the atmospheric surface layer: a re-evaluation. *Boundary-Layer Meteorology*, 42:55–78, 1988.

W. Hundsdorfer, B. Koren, M. van Loon, and J.G. Verwer. A positive finite-difference advection scheme. *Journal of Computational Physics*, 117:35–46, 1995.

M. Khairoutdinov and Y. Kogan. A new cloud physics parametrization in a large-eddy simulation model of marine stratocumulus. *Monthly Weather Review*, 128:229–243, 2000.

R.A. Pielke. *Mesoscale Meteorological Modeling*. Elsevier, New York, 1984.

H. Schmidt and U. Schumann. Coherent structure of the convective boundary layer derived from large-eddy simulations. *Journal of Fluid Mechanics*, 200:511–562, 1989.

A. Seifert and K.D. Beheng. A double-moment parameterization for simulating autoconversion, accretion and selfcollection. *Atmospheric Research*, 59-60:265–281, 2001.

A. Seifert and K.D. Beheng. A two-moment cloud microphysics parameterization for mixed-phase clouds. part 1:model description. *Meteorology and Atmospheric Physics*, 92:45–66, 2006.

G. Sommeria and J.W. Deardorff. Subgrid-scale condensation in models of nonprecipitating clouds. *Journal of the Atmospheric Sciences*, 34:344–355, 1977.

B. Stevens and A. Seifert. Understanding macrophysical outcomes of microphysical choices in simulations of shallow cumulus convection. *Journal of the Meteorological Society of Japan*, 86A:143–162, 2008.

L.J. Wicker and W.C. Skamarock. Time-splitting methods for elastic models using forward time schemes. *Monthly Weather Review*, 130:2088–2097, 2002.

Appendix A: The Poisson Solver

In this appendix the Poisson solver is described. As explained in Section 2.4.1, this subroutine needed to be modified in order to use a non-equidistant numerical grid.

Introduction

The filtered Navier-Stokes equations can be written in the general form as (equation 5.2):

$$\frac{\partial u_i}{\partial t} = -\frac{\partial u_i u_j}{\partial x_j} + g\frac{\theta_l}{\theta_0}\delta_{i3} - 2\epsilon_{ijk}\Omega_j u_k - \frac{\partial \tau_{ij}}{\partial x_j} - \frac{\partial \pi}{\partial x_j}, \tag{5.61}$$

where the angular brackets <> (representing the filter operator) are omitted for sake of clarity.
As explained before, the modified pressure reads

$$\pi = \frac{p}{\rho_0} + 2/3e. \tag{5.62}$$

Equations (5.61) can be split in two contributions according to :

$$\frac{\partial u_i}{\partial t} = -\frac{\partial p}{\partial x_i} + Rem. \tag{5.63}$$

where *Rem* accounts for all the remaining terms in the r.h.s. of Equation (5.61) (advection, buoyancy, rotation and subgrid stress) except for the pressure gradient.
By introducing a fictitious velocity u^*, Equation (5.63) is decomposed as follows:

$$\begin{cases} \frac{\partial u^*}{\partial t} = Rem \\ \\ \frac{\partial u}{\partial t} - \frac{\partial u^*}{\partial t} = -\frac{\partial p}{\partial x_i} \end{cases} \tag{5.64}$$

Applying the divergence operator to the second equation of the system (5.64) we obtain:

$$\nabla\left(\frac{\partial u}{\partial t} - \frac{\partial u^*}{\partial t}\right) = -\nabla^2 p, \tag{5.65}$$

which becomes, using the continuity equation,

$$\nabla \left(\frac{\partial u^*}{\partial t} \right) = f(u, x, y, z, t) = \nabla^2 p, \tag{5.66}$$

that is a form of the Poisson's equation. Note that $\nabla u^* \neq 0$ because the fictitious velocity represents all the remaining terms of the Navier-Stokes equation, whose divergence is not necessarily zero.
Assuming that the *Rem* term is known (Equation 5.64), we first calculate the pressure by solving the Poisson's equation (5.66). Once the pressure is known, we can finally solve for $\frac{\partial u}{\partial t}$ in equation (5.64).

Time discretization

The discretized form of the l.h.s. of equation (5.63) reads (leap-frog method):

$$\frac{u_i^{n+1} - u_i^{n-1}}{2\Delta t} = -\frac{\partial p}{\partial x_i} + Rem, \tag{5.67}$$

Similarly as above, a fictitious velocity u^* can be introduced and the equation (5.67) can be split into:

$$\begin{cases} \frac{u_i^* - u_i^{n-1}}{2\Delta t} = Rem \\ \\ \frac{u_i^{n+1} - u_i^*}{2\Delta t} = -\frac{\partial p}{\partial x_i}, \end{cases} \tag{5.68}$$

which is the equivalent of the system (5.64).
By applying the divergence operator to (5.68) and making use of the continuity equation, we obtain

$$\nabla \left(\frac{u_i^*}{2\Delta t} \right) = \nabla^2 p, \tag{5.69}$$

which is the equation we need to solve, namely the discretized form of the Poisson's equation (5.66).

Spatial discretization

By defining the quantity

$$u_i^{**} \equiv \frac{u_i^*}{2\Delta t}, \tag{5.70}$$

the discretized form of the Poisson's equation (5.69) reads

$$\nabla \left(u_i^{**} \right) = \nabla^2 p. \tag{5.71}$$

Figure ... shows the numerical grid used in the code, in the vertical direction. The scalar quantities (temperature, humidity, scalar and pressure) are defined in the center of the cell, whereas the wind velocities are defined at the edges, so that $p(k)$ is not at the same level then $w(k)$.
Moreover, referring to Figure ..., it can be noticed that, if a non-equidistant grid is used in the vertical direction, the distance between two adjacent centers (Δz_h) does not coincide with the distance between two edges (Δz_f).
In discretizing equation (5.71), we have to compute the velocity gradient (l.h.s.) and the pressure Laplacian (r.h.s) at the same level, namely at the center of the cell. Therefore, the l.h.s. can be written in the discrete (centered) form as

$$\nabla u_i^{**} = \frac{u_{i+1}^{**} - u_i^{**}}{\Delta x} + \frac{v_{j+1}^{**} - v_j^{**}}{\Delta y} + \frac{w_{k+1}^{**} - w_k^{**}}{\Delta z_f(k)}, \tag{5.72}$$

whereas the pressure Laplacian (at the same level) is

$$\begin{aligned}\nabla^2 p = \frac{\partial}{\partial x_j}\left(\frac{\partial p}{\partial x_i}\right) = \frac{1}{\Delta x}\left[\left(\frac{\partial p}{\partial x}\right)_{i+\frac{1}{2}} - \left(\frac{\partial p}{\partial x}\right)_{i-\frac{1}{2}}\right] + \\ + \frac{1}{\Delta y}\left[\left(\frac{\partial p}{\partial y}\right)_{j+\frac{1}{2}} - \left(\frac{\partial p}{\partial y}\right)_{j-\frac{1}{2}}\right] + \\ + \frac{1}{\Delta z_f(k)}\left[\left(\frac{\partial p}{\partial z}\right)_{k+\frac{1}{2}} - \left(\frac{\partial p}{\partial z}\right)_{k-\frac{1}{2}}\right].\end{aligned} \tag{5.73}$$

and pressure gradients at the cell's edge (point $i \pm 1/2$) are defined as

$$\begin{cases} \frac{\partial p}{\partial x}|_{i+\frac{1}{2}} = \frac{p_{i+1} - p_i}{\Delta x} \\ \\ \frac{\partial p}{\partial x}|_{i-\frac{1}{2}} = \frac{p_i - p_{i-1}}{\Delta x}. \end{cases} \tag{5.74}$$

A similar formula holds for the y component.
The vertical pressure gradient reads:

$$\begin{cases} \frac{\partial p}{\partial z}|_{k+\frac{1}{2}} = \frac{p_{k+1} - p_k}{\Delta z_h(k+1)} \\ \\ \frac{\partial p}{\partial z}|_{k-\frac{1}{2}} = \frac{p_k - p_{k-1}}{\Delta z_h(k)}. \end{cases} \tag{5.75}$$

As a result, the discretized form of the Laplacian of the pressure reads

$$\begin{aligned}\nabla^2 p = \frac{p_{i+1} - 2p_i + p_{i-1}}{(\Delta x)^2} + \frac{p_{j+1} - 2p_j + p_{j-1}}{(\Delta y)^2} + \\ + \frac{1}{\Delta z_f(k)}\left(\frac{p_{k+1} - p_k}{\Delta z_h(k+1)} - \frac{p_k - p_{k-1}}{\Delta z_h(k)}\right),\end{aligned} \tag{5.76}$$

and the Poisson equation (5.71) becomes

$$\frac{u^{**}_{i+1} - u^{**}_i}{\Delta x} + \frac{v^{**}_{j+1} - v^{**}_j}{\Delta y} + \frac{w^{**}_{k+1} - w^{**}_k}{\Delta z_f(k)} =$$
$$= \frac{p_{i-1} - 2p_i + p_{i+1}}{(\Delta x)^2} + \frac{p_{j-1} - 2p_j + p_{j+1}}{(\Delta y)^2} + ap_{k-1} + bp_k + cp_{k+1}, \tag{5.77}$$

where

$$a = \frac{1}{\Delta z_f(k)\Delta z_h(k)} \tag{5.78}$$
$$c = \frac{1}{\Delta z_f(k)\Delta z_h(k+1)} \tag{5.79}$$
$$b = -(a + c). \tag{5.80}$$

Poisson Solver in the LES code

In the code used in this study, the routine FILLPS.F first computes, from Equation (5.68), the sum

$$u^{**}_i = \frac{u^*_i}{2\Delta t} = \frac{u^{n-1}_i}{2\Delta t} + Rem, \tag{5.81}$$

and then solves for the divergence ∇u^{**}_i (equation 5.72).
This is followed by the routine SOLMPJ.F that extracts the pressure p solving the Poisson's equation (5.77), given the value of ∇u^{**}_i calculated by FILLPS.F, applying a FFT in the two periodic directions (x and y), and solving a tridiagonal matrix in the vertical direction. The routine TDERIVE.F computes the velocity tendency $\frac{\partial u_i}{\partial t} \equiv u_p$, from Equation (5.63), namely

$$\frac{\partial u_i}{\partial t} \equiv u_p = Rem - \frac{\partial p}{\partial x_i} = u^{**}_i - \frac{u^{n-1}_i}{2\Delta t} - \frac{p_i - p_{i-1}}{\Delta x} - \frac{p_j - p_{j-1}}{\Delta y} - \frac{p_k - p_{k-1}}{\Delta z_h(k)}, \tag{5.82}$$

where equation (5.81) has been used, and the pressure gradient has been discretized at the same level of the velocity field.
Since by definition

$$\frac{\partial u_i}{\partial t} = \frac{u^{n+1}_i - u^{n-1}_i}{2\Delta t}, \tag{5.83}$$

the routine TSTEP.F can finally calculate the velocity at the time step $n + 1$

$$u^{n+1}_i = u_p 2\Delta t + u^{n-1}_i. \tag{5.84}$$