Manual for DALES
CIE4701: From Field Observations To Modeling

Stephan de Roode, Erwin de Beus, Pim van Dorp,
Fredrik Jansson, Pier Siebesma and Antoon van Hooft

March 12, 2021

Contents

1 Introduction **3**

2 Running DALES in the VirtualBox **4**
2.1 Make an executable of the code 4
2.2 Perform a simulation with DALES 4
2.2.1 Input files 4
2.2.2 Execute a simulation 5
2.3 File transfer 6
2.4 Known issues with the Virtualbox 6

3 Structure of the input files **8**
3.1 prof.inp 8
3.1.1 Header lines 8
3.1.2 Vertical grid resolution, half and full levels 8
3.1.3 Input variables 8
3.2 lscale.inp 9
3.3 namoptions 10

4 The Convective Boundary Layer **12**

5 Formulation of the LES model **14**
5.1 Prognostic equations 14
5.2 Formulations of the length scale 15
5.2.1 Constant length scale l_Δ 15
5.2.2 Stability dependent length scale l_Δ 16
5.2.3 Mason and Thomson length scale l_M 16
5.3 Model constants 16

6 Shallow Cumulus **18**
6.1 Conditional sampling 18
6.2 Precipitation 18

7 Instruction notes on the Stable Boundary Layer **20**
7.1 Introduction 20
7.2 The GABLS1 stable boundary layer model intercomparison case. . . 20
7.2.1 Adaptive time step 20
7.2.2 Input file for time-dependent large-scale forcings ls_flux.inp . . 21

8 Radiative transfer **23**
8.1 The atmospheric structure above the LES domain: backrad.inp 24

9 Warm start, visualization **26**
9.1 Introduction 26
9.2 How to perform a warm start 26
9.3 Creating 3D fields (Optional exercise) 27

10 Compiling Dales **28**
10.1 Compiling new code 28
10.2 Compilers 28

11 Model crashes **29**

A Linux tips **30**

B Installing Dales on a Mac **30**

C Installing Dales on Ubuntu 16.04 **30**

D Installing Dales on the VRLab system **31**

E Error messages **32**

1 Introduction

This guide starts by providing all essential steps to get started with the Dutch Atmospheric Large Eddy Simulation (DALES) model version 4.1 on your own computer, from compiling the source code to running your first case. This description has been adapted from an early description of DALES 3.1 by Thijs Heus, Chiel van Heerwaarden and Johan van der Dussen.
DALES is written in Fortran 90. This language is still frequently used in the scientific communities of geophysics and astronomy basically because in the 1950s many early scientific programs were developed in Fortran. For example, numerical weather prediction (NWP) was one of the first intensive computer applications, and many current state-of-the-art NWP models have their roots in these early Fortran versions. DALES can be run using multiple CPUs, and it has the option to write output in the so-called NCDF format. This however requires quite some software packages, which are often conveniently available in Linux environments. To perform numerical simulations you will have to install the VirtualBox[1] and the virtual Linux environment including DALES4.1[2]. Sometimes you want to install a package which requires the root password: in this application it has been set to `dales`. Some convenient Linux commands will be summarized in Appendix A. In case you happen to operate Ubuntu on your computer you can also find a recipe to install Dales on Ubuntu 16.04 in Appendix C. In case you want to install DALES on your Mac please follow the instructions in Appendix B. More information on installing DALES on systems that are not covered by this manual can be found on the DALES Wiki:`https://github.com/dalesteam/dales/wiki/Installation-notes`.

A nice resource for learning about Unix and the command line specificly for scientific computing: `https://software-carpentry.org/lessons/`. Especially the lessons "The Unix Shell", "Version Control with Git" are relevant for this course, and perhaps the two lessons on Python.

[1] available for downloading from `https://www.virtualbox.org`. In case you are already using Ubuntu or OpenSuSe you can install VirtualBox from `zypper install virtualbox` or `sudo apt-get install virtualbox`, respectively.

[2] available for downloading from
`http://www.srderoode.nl/Teaching/LES_course/Dales_4.1.5-2018.ova`

2 Running DALES in the VirtualBox

In the virtual Linux environment you will see eight icons at the top left of the screen. Open a terminal by clicking on the terminal icon. Type `ls` to see which files and directories are present. The source code of DALES is stored in `dales/src`. You can go to this directory by typing
`cd dales/src`[3]
If you want to read the contents of a file type `atom filename.f90`. If you open the file `program.f90`[4] you will learn about the skeleton of the DALES model in the sense that it calls many subroutines. Note that every `*f90` routine starts with a brief description. Since DALES is a collaboration between different institutions (Heus et al., 2010) it also states the authors of the subroutines, which can be handy in case one has specific questions about the code. For example, subroutines including the effect of plants on evaporation and atmospheric chemistry have been developed by our colleagues from Wageningen University. The bulk of the routines relevant to clouds have been written (e.g. precipitation, moist thermodynamics) or incorporated (radiative transfer) by team members from the TU Delft and KNMI.

2.1 Make an executable of the code

To make an executable of the model code create a new directory by typing

```
mkdir dales.build
```

Go to this directory and type

`cmake ~/dales`[5].

You will see that the directory contains some directories. In addition the script `Makefile` has been generated. It compiles the Fortran file and makes an executable. Execute this file by typing

`make`[6].

The `dales4` model has been put in the directory `~/dales.build/src`.

2.2 Perform a simulation with DALES

2.2.1 Input files

Some examples of input files have been posted on the internet. As you will often use the same input files for different experiments it is therefore convenient to put them in a separate directory. To this end create the directory `~/Cases/CBL`, go to this subdirectory by typing `cd ~/Cases/CBL` and download the three following files
`prof.inp.cbl_fixed_grad`,

[3] If you want to change directories you can type `cd ..` If you are in the directory `dir1/dir2`, this command will change your directory to `dir1`. If you want to go to back to your home directory simply type `cd`

[4] Linux tip: type `atom pr` followed by a click on the 'tab' of your keyboard. You will see that the system automatically finishes the filename.

[5] At some computers, like Macs, the $\sim$ symbol can be obtained from the Shift-§-button

[6] If your computer has multiple CPUs you can type `make -j n` to use a number of n CPUs, with $1 \leq n \leq N$ an integer that should be smaller than the number N of CPUs on your machine. If you type `make -j` the number of processors is automatically set to N. Be careful if you do this on a computer cluster as your task will use all CPUs which may lead to a significant speed reduction of the other running processes. You may loose friends if you do so.

`lscale.inp.cbl_fixed_grad`,
`namoptions.coarse`
by typing, as an example,

`wget www.srderoode.nl/Teaching/LES_course/CBL/namoptions.coarse`[7]

We will perform various experiments of the Convective Boundary Layer (CBL) and we will store the results in the directory `~/Experiments/CBL` (create it if it is not present). We wish to collect the output from our first experiment in the subdirectory `~/Experiments/CBL/H001`. Before we can do a run, we have to copy some obligatory input files to this subdirectory. Go to the subdirectory `H001` (`cd ~/Experiments/CBL/H001`) and copy the following files

```
cp ~/Cases/CBL/prof.inp.cbl_fixed_grad prof.inp.001
cp ~/Cases/CBL/lscale.inp.cbl_fixed_grad lscale.inp.001
cp ~/Cases/CBL/namoptions.coarse namoptions
cp ~/dales.build/src/dales4 .
```

In the last command the dot indicates that the name of the copied file remains the same. Always check if the experiment number `iexpnr` in the second line of `namoptions` is the same as the number of the subdirectory, otherwise change to `iexpnr = 001`, for example with aid of `atom namoptions`. The `namoptions` file tells the model to do a simulation on a horizontal domain size of `xsize=ysize`=1.6 km using `itot=jtot`=32 grid points that lasts only `runtime`= 300 s with a time step of `dtmax`=1 s . Except for some testing purposes such a small number of grid points in the horizontal plane (`itot=jtot`=32) is actually never used in practice. However, here we find it convenient to quickly see whether the model runs smoothly.

2.2.2 Execute a simulation

Start a model run by typing

`mpirun -np 2 dales4 |tee output.001`.

The command tells the computer to use 2 CPUs and to send information about the run both to screen and an output file. The last line presents the total duration of the simulation (W =`TOTAL wall time`). The model experiment can be sped up if you have the opportunity to use more than 2 CPUs.

[7] You can obtain the history of your previous commands by using the upwards pointing arrow at your keyboard.

Exercise 1: Applying LES to the entire Earth?

1a. Use the information given above to calculate the computation time for one grid point per computational time step per CPU. The total number of grid points N_p can be found in `namoptions` and is equal to
N_p = `itot`$\cdot$ `jtot`$\cdot$ `kmax`
The number of time steps is equal to the ratio N_t = `runtime`/`dtmax`. With aid of the wall clock time W the computational time per time step per grid point can be calculated from $\mathcal{T} = \frac{W N_\mathrm{CPU}}{N_t N_\mathrm{p}}$. We have to multiply by N_CPU as the wall clock time is related to its reciprocal[a].
1b. Use the radius of the Earth (6400 km) to compute the global surface area. Compute the total time needed to repeat simulation `H001` for the entire Earth and for a period of 24 hours using only one CPU.
1c. Suppose that you can make a network consisting of the same CPUs as you are using, and that the calculation time will be reduced inversely proportional to the number of CPUs. How many CPUs are needed to finish the calculation within 24 hours?

[a] here we implicitly assume that a number of n CPUs reduce the wall clock time by about a factor n. In reality, the speed up is less than n. This is due to the fact some time is spent for *communication* of data from one processor to the other.

We are now in a position to run the model and obtain model output. However, we have not yet explained what is actually included in the input files. So before digging into the details of the model output we will first explain the structure of the input files.

2.3 File transfer

It is possible to move files from the Linux environment to your laptop and vice versa. This allows to process LES output fields or to prepare LES input files in your own familiar laptop environment. In any case, take care that you have installed the latest version of VirtualBox (see in the VirtualBox VM menu → Check for updates). Then you have to go to Machine → Settings (see Fig. 1). Then go to Shared Folders and select an arbitrary directory on your laptop, for example `Desktop`. Take care that you switch on Auto-mount and Make Permanent. Next, in VirtualBox, go to Devices, then click "Insert guest additions CD image". You can now access your laptop folder via `cd /media`, in this directory you will see the contents of `Desktop`. In case it does not work rebooting the system may be necessary (type `reboot` in your terminal). Copying a file can be done by

```
cp file /media/your_laptop_folder/file
```

2.4 Known issues with the Virtualbox

Virtualbox cannot import the `*.ova` file "invalid argument"
Virtualbox apparently cannot handle special characters like "ë" in directory names. Move the file to another location.

Virtualbox does not start, gives an error about VTx.
Solution: enable VTx in the BIOS. Restart the computer. While it boots, press some

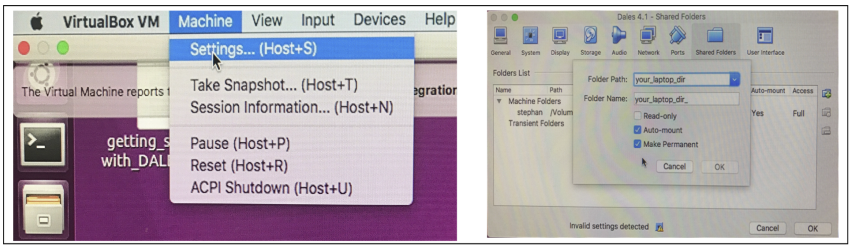

Figure 1: Connecting the virtual Linux environment to your laptop for file sharing.

key to get to a boot menu or the BIOS settings. Find a setting for VTx or virtualization, turn it on, save, restart.

On HP: the key to enter the BIOS is Esc or F10. One may have to quickly press the key many times while the computer starts. The VTx setting is under Advanced → Device options(?), then scroll down to find the VTx setting.

Shared folder in Virtualbox
There is a one-time procedure to install some addons inside the Virtualbox for the shared folder to work: in VirtualBox, go to Devices, then click "Insert guest additions CD image". This is supposed to automatically install something, but if it doesn't, open a terminal and type:

```
cd /media/dales/VBox_GAs_6.1.12 # or try with tab completion
./autorun.sh
```

When asked for a password, say "dales". When the installation completes, restart the virtualbox. Now, shared folders should be visible, under `/media` if you did not specify a different path.

3 Structure of the input files

3.1 prof.inp

3.1.1 Header lines

```
Dry Convective Boundary Layer LES Course Class 1
 height(m) thl(K) qt(kg/kg) u(m/s) v(m/s) tke(m2/s2)
 12.5000 293.075 0.00000 1.00000 0.00000 0.000800000
 37.5000 293.225 0.00000 1.00000 0.00000 0.000266667
 62.5000 293.375 0.00000 1.00000 0.00000 0.000160000
......
```

Table 1: Example of a `prof.inp` file.

Table 1 shows the first few lines of `prof.inp`[8]. The LES model expects two header lines. If you have only one, or more than two, the model will crash immediately. The first header line contains a brief description of the case, and the second one the meaning of the columns.

3.1.2 Vertical grid resolution, half and full levels

The example applies a constant vertical grid resolution $\Delta z = 25$ m. Note that the first level is located at $z_{\mathrm{f}}(1) = \Delta z/2 = 12.5$ m. This has to do with the fact that DALES uses a staggered grid, in which all variables are located at the so-called full levels (z_{f}) except for the vertical velocity which is defined at the half levels, $z_{\mathrm{h}}(1) = 0, z_{\mathrm{h}}(2) = \Delta z, z_{\mathrm{h}}(3) = 2\Delta z$, etc. The key point here is that the heights of the full vertical levels z_{f} are defined in `prof.inp`. The half levels are determined by DALES by

$$z_{\mathrm{h}}(k+1) = z_{\mathrm{h}}(k) + 2(z_{\mathrm{f}}(k) - z_{\mathrm{h}}(k)). \tag{1}$$

Note that a non-equidistant vertical grid structure is allowed, but at this point we recommend to use a vertically equidistant grid only.

3.1.3 Input variables

Dales uses the liquid water potential temperature as a prognostic variable. Its approximated form reads,

$$\theta_{\mathrm{l}} \approx \theta - \frac{L_{\mathrm{v}}}{c_{\mathrm{p}}\Pi} q_{\mathrm{l}}, \tag{2}$$

with the potential temperature

$$\theta = T \left(\frac{p_0}{p} \right)^{\frac{R_{\mathrm{d}}}{c_{\mathrm{p}}}}. \tag{3}$$

The Exner function Π is defined by

$$\Pi = \left(\frac{p}{p_0} \right)^{\frac{R_{\mathrm{d}}}{c_{\mathrm{p}}}}. \tag{4}$$

[8]The source file that produced the *.inp files can be obtained from `www.srderoode.nl/Teaching/LES_course/CBL/make_input.pro`. You can run it after installing `GDL` by `apt install gnudatalanguage`, then type `gdl` and write the *inp files with aid of the command `.r make_inp`

where L_v is the latent heat of vaporization, c_p is the specific heat of dry air at constant pressure, T is the absolute temperature, q_l the liquid water specific humidity, and p_0 is a reference pressure. Their values are summarized in Table 2.

c_p	1004	J/kg/K	Specific heat of dry air at constant pressure
g	9.81	$\mathrm{m}^2 s^{-2}$	Gravitational acceleration
L_v	$2.53 \cdot 10^6$	$\mathrm{J\ kg^{-1}K^{-1}}$	Latent heat of vaporization
p_0	1000	hPa	Reference pressure
R_d	287.04	$\mathrm{J\ kg^{-1}K^{-1}}$	Gas constant for dry air
R_v	461.5	$\mathrm{J\ kg^{-1}K^{-1}}$	Gas constant for water vapor

Table 2: Summary of constants used in dales (see `modglobal.f90`.)

In addition, Dales uses the total water specific humidity q_t as a second prognostic thermodynamic variable,

$$q_\mathrm{t} = q_\mathrm{v} + q_\mathrm{l}, \tag{5}$$

with q_v the water vapor specific humidity. Note that if the air is saturated with water vapor, q_v equals the saturation specific humidity q_sat. The values of the liquid water potential temperature (`thl`, θ_l), the total water specific humidity (`qt`, q_t), and the horizontal wind velocity components (`u`, u) and (`v`, v) represent the initial vertical profiles at $t = 0$. They need to be obtained from observations. The last quantity `tke` indicates the initial subgrid Turbulent Kinetic Energy (TKE). To promote the development of turbulence at the resolved scales it usually has a nonzero value in the lower part of the atmosphere.
The number of lines with input data should correspond to the number `kmax` in `namoptions`.

3.2 lscale.inp

```
Dry Convective Boundary Layer LES Course Class 1
 height(m) ugeo(m/s) vgeo(m/s) wfls(m/s) not_used not_used dqtdtls(kg/kg/s) dthldt(K/s)
 12.500 1.000 0.000 0.000000 0.0 0.0 0.0 0.0
 37.500 1.000 0.000 0.000000 0.0 0.0 0.0 0.0
 62.500 1.000 0.000 0.000000 0.0 0.0 0.0 0.0
......
```

Table 3: Example of a `lscale.inp` file.

Table 3 shows the first few lines of `lscale.inp`. Its values indicate large-scale forcing quantities that are assumed to be constant in time. If the forcings are time dependent another input file needs to be built which will be discussed at a later stage. The geostrophic wind components are indicated by `ugeo` and `vgeo`. Recall that the geostrophic wind is the wind velocity that would occur if the horizontal pressure force balances the Coriolis force, which implies that it ignores variations in time and turbulence. In fact, if we talk about prescribing a geostrophic wind, this is actually similar to stating that we define the horizontal pressure gradient since

$$u_\mathrm{geo} \equiv -\frac{1}{\rho f}\frac{\partial p}{\partial y} \quad , \quad v_\mathrm{geo} \equiv \frac{1}{\rho f}\frac{\partial p}{\partial x}. \tag{6}$$

The large-scale subsidence is given by `wfls`. There are two columns with the weird names `not_used` that should be present but are actually not used anymore in Dales. Because we wish to keep input files backwards compatible (meaning that files can be used in any arbitrary Dales version) such ghost data need to be included.

3.3 namoptions

Table 4 shows the `namoptions` file used for the second experiment. A complete description of the `namoptions` file is presented in `https://github.com/dalesteam/dales/blob/master/utils/doc/input/Namoptions.pdf`. The meaning of some quantities merit some attention, like `irandom`, `randthl` and `randqt`. We have seen that Dales reads vertical profiles of θ_l and q_t. Initially the same values are prescribed at every grid point in a horizontal plane. This results in model columns that are all the same. This implies that the horizontal advection terms are the same since $\partial\theta_\mathrm{l}/\partial x = 0$. Since at $t = 0$, $w = 0$, the vertical advection is also zero. Furthermore, if a horizontal plane is homogeneous, there will be no buoyancy force since this term depends on buoyancy differences with respect to the slab-mean value. In other words, turbulence will not develop. To avoid this situation any LES model starts with a random perturbation of the initial thermodynamic fields. This is achieved with a randomnize function that is present in the subroutine `modstartup.f90`. This function is fed with a first 'seeding' value, which is represented by `irandom`, to give a value r that varies between -1 and 1. The value of the perturbation at each grid point is computed by multiplying r with the amplitude of the fluctuation, `randthl` and `randqt`, respectively.

Dales contains several advection schemes. Its second-order advection scheme approximates gradients as

$$\frac{\partial \phi}{\partial x} \approx \frac{\phi(x + \Delta x) - \phi(x - \Delta x)}{2\Delta x}. \tag{7}$$

Dales produces ASCII output. However, if you know how to deal with NetCDF files you can obtain the results in this format by setting `lnetcdf=.true.` Scalar statistics like the boundary-layer height are saved at every `dtav` for `ltimestat=.true.`, and vertical mean profiles are written to a file at every time step `timeav`.

`&RUN`	
`iexpnr = 002`	experiment number
`runtime = 36000`	total simulation time (secs)
`dtmax = 10.`	numerical time step (secs), applied if `ladaptive=.false.`
`ladaptive = .true.`	max time step is determined by the CFL criterion
`irandom = 43`	number used in randomizer function
`randthl = 0.1`	amplitude of initial θ_l perturbations
`randqt = 0e-5`	amplitude of initial q_t perturbations
`/`	
`&DOMAIN`	
`itot = 64`	nr of points in the horizontal $x-$direction
`jtot = 64`	nr of points in the horizontal $y-$direction
`kmax = 128`	nr of points in the vertical $z-$direction
`xsize = 3200.`	domain size in the $x-$direction (m)
`ysize = 3200.`	domain size in the $y-$direction (m)
`xlat = 32.`	latitude, needed for Coriolis force
`xlon = 129.`	longitude, only relevant for solar radiation
`/`	
`&PHYSICS`	
`ps = 101900.00`	surface pressure (Pa)
`thls = 293.`	surface value of θ_l (K), not needed if fluxes are prescribed
`lcoriol = .true.`	include the Coriolis force
`iradiation = 0`	0=no radiation
`z0 = 0.1`	surface roughness length (m)
`/`	
`&NAMSURFACE`	
`wtsurf = 0.1`	surface value of $\overline{w'\theta'_\mathrm{l}}$ (mK/s)
`wqsurf = 0e-5`	surface value of $\overline{w'q'_\mathrm{t}}$ ((kg/kg)(m/s))
`isurf = 4`	4: use z_0 and prescribed surface values for $\overline{w'\theta'_\mathrm{l}}$, $\overline{w'q'_\mathrm{t}}$
`/`	
`&DYNAMICS`	
`cu = 1.`	Galilean transformation velocity in the $x-$direction (trick to reduce time step)
`cv = 0.`	Galilean transformation velocity in the $y-$direction
`iadv_mom = 2`	2:2nd order (Central Difference) advection scheme for momentum
`iadv_tke = 2`	
`iadv_thl = 2`	
`iadv_qt = 2`	
`/`	
`&NAMSUBGRID`	
`ldelta = .true.`	length scale in TKE scheme $\ell = \Delta = (\Delta x \Delta y \Delta z)^{1/3}$
`/`	
`&NAMCHECKSIM`	
`tcheck = 5`	computes and writes the max value of the divergence of the velocity field every 5 s
`/`	
`&NAMTIMESTAT`	
`ltimestat = .true.`	compute statistics of scalars (`tmser` and `tmsurf`)
`dtav = 60`	write this output to file every 60 s
`/`	
`&NAMGENSTAT`	
`lstat = .true.`	computes horizontal slab mean statistics
`dtav = 60`	compute slab average every 60 s
`timeav = 600`	write average during a simulation period of 600 s
`/`	
`&NAMNETCDFSTATS`	
`lnetcdf = .true.`	provide output in NCDF format
`/`	

Table 4: Example of a minimal `namoptions` file. Many more switches can be added, or the value of switches can be changed, which will be discussed in the remainder of the course.

4 The Convective Boundary Layer

The Convective Boundary Layer (CBL) is one of the first boundary-layer regimes that was investigated with LES (Nieuwstadt et al., 1993)[9]. To study how its time evolution depends on the surface forcing and atmospheric stability we will now perform some sensitivity experiments. For each experiment use a different experiment number.

Let us now perform a second experiment that will simulate a period of 10 hours. To this end download a new input file in the `Cases` directory,

```
cd ~/Cases/CBL
wget www.srderoode.nl/Teaching/LES_course/CBL/prof.inp.cbl_fixed_grad_v2
```

Create a new directory `H002` and download a new `namoptions` file in this directory,

```
cd ~/Experiments/CBL
mkdir H002
cd H002
wget www.srderoode.nl/Teaching/LES_course/CBL/namoptions
```

The main differences with your first `namoptions` settings are as follows:

`iexpnr`	=	`002`	new experiment number
`runtime`	=	`36000`	run time of 10 hours
`ladaptive`	=	`.true.`	reduces time step to ensure $\|u\|\Delta t/\Delta x < 1$
`itot`	=	`64`	nr of grid points in x-direction
`jtot`	=	`64`	nr of grid points in x-direction
`xsize`	=	`3200`	domain size in x-direction (m)
`ysize`	=	`3200`	domain size in y-direction (m)

Copy the other necessary input files and executable to `H002`,

```
cp ~/Cases/CBL/prof.inp.cbl_fixed_grad_v2 prof.inp.002
cp ~/Cases/CBL/lscale.inp.cbl_fixed_grad lscale.inp.002
cp ~/dales.build/src/dales4 .
```

and execute dales4

```
mpirun -np 2 dales4 |tee output.002
```

At this point it will be convenient to open a text document in which you briefly describe some key details about the different runs, for example

```
atom ~/Experiments/CBL/cbl_run_info.txt
```

with a content like

```
H001 - test run of 60 secs
H002 - run of 10 hours
```

At this point this sounds trivial, but if you have done more than 10 runs, after some

[9] As this paper is difficult to find we have posted it at `www.srderoode.nl/Teaching/LES_course/Literature/Nieuwstadt_etal_LES_CBL.pdf`.

time you will likely not be able to remember the details and motivation of the various experiments.

Exercise 2: Evolution of the mean state.

Use the results of the second experiment to make a plot of the temperature (*i*) T and (*ii*) potential temperature θ at different times, for example the first and the last time block. These variables are written to the ASCII file `field.iexpnr`[a]. Discuss their vertical gradients. (*iii*) Can you see the development of a thermal inversion layer? Use the file `tmser1` to make a plot of the quantity `zi` as a function of time. (*iiii*) Plot and discuss the subgrid, resolved and total fluxes of $\overline{w'\theta'}$ as a function of height in a single plot, for example for the last time block. We will call this simulation the Reference case.

Exercise 3: Sensitivity experiments.

Investigate to which extent the results change if you modify, for example, the horizontal grid size Δx. This can be achieved by changing the horizontal domain sizes `xsize` and `ysize`. As a suggestion, try $\Delta x = \Delta y$ =10, 50, 250, 500, 1000 m and compare your results with the Reference case.

[a] It is also possible to read the same data set saved in NCDF format with aid of the matlab routine `netcdfplotexample.m` available from `www.srderoode.nl/Teaching/LES_course/Software`

5 Formulation of the LES model

Here we will briefly summarize the governing LES equations that apply to an atmosphere free of clouds. For a detailed description of an LES model like DALES we refer the reader to Heus et al. (2010) and Böing (2014) who explain the updated anelastic version for simulations of deep convection.

5.1 Prognostic equations

LES models solve the budget equations for filtered variables including momentum and thermodynamic state variables, such as heat, entropy, or the total water specific humidity. After application of the LES filter the prognostic equation for an arbitrary scalar φ can be written as

$$\frac{\partial \tilde{\varphi}}{\partial t} + \tilde{u}_j \frac{\partial \tilde{\varphi}}{\partial x_j} = -\frac{\partial \widetilde{u_j'' \varphi''}}{\partial x_j} + \tilde{S}_\varphi, \tag{8}$$

with t the time, and the velocity vector components $(u_1, u_2, u_3) = (u, v, w)$ in the (x, y, z) direction, respectively. In the absence of clouds $\varphi \in \{\theta, q\}$, with θ representing the potential temperature and q the water vapor specific humidity. A tilde indicates the filtered mean value and the SFS scalar flux is denoted by $\widetilde{u_j'' \varphi''} \equiv \widetilde{u_j \varphi} - \tilde{u}_j \tilde{\varphi}$. The Boussinesq form of the filtered momentum equation reads,

$$\frac{\partial \tilde{u}_i}{\partial t} + \tilde{u_j} \frac{\partial \tilde{u}_i}{\partial x_j} = \frac{g}{\theta_0} \delta_{i3} (\tilde{\theta_\mathrm{v}} - \overline{\theta_\mathrm{v}}) - \frac{\partial \pi}{\partial x_i} - \frac{\partial \tau_{ij}}{\partial x_j} + \tilde{S}_{u_i}, \tag{9}$$

where g is the gravitational acceleration, θ_0 the reference state virtual potential temperature, δ_{ij} the Kronecker-delta function, the virtual potential temperature is defined as,

$$\theta_\mathrm{v} \equiv \theta(1 + \epsilon_I q), \tag{10}$$

with $\epsilon_I \approx 0.608$ a thermodynamic constant, π is the modified pressure (Deardorff, 1973),

$$\pi = \frac{p}{\rho_0} + \frac{2}{3}\tilde{e}, \tag{11}$$

and an overbar is used to indicate a horizontal slab-mean value. For compact notation we have included the mean horizontal pressure gradient and the Coriolis force in the source function S_{u_i}. The deviatoric part of the SFS momentum flux τ_{ij} is computed from (Deardorff, 1980),

$$\tau_{ij} \equiv \widetilde{u_i u_j} - \tilde{u}_i \tilde{u}_j - \frac{2}{3}\delta_{ij}\tilde{e} = -K_\mathrm{m} \left(\frac{\partial \tilde{u}_j}{\partial x_i} + \frac{\partial \tilde{u}_i}{\partial x_j} \right), \tag{12}$$

and

$$\widetilde{u_j'' \varphi''} = -K_\mathrm{h} \frac{\partial \tilde{\varphi}}{\partial x_j}. \tag{13}$$

The factor $\frac{2}{3}\delta_{ij}\tilde{e}$ that is subtracted in (12) does not arise from the filtering procedure. To compensate it has been added to the filtered pressure term to give the modified pressure. Here K_m and K_h represent the eddy viscosity for momentum and the eddy diffusivity for the thermodynamic scalars, respectively. In a TKE closure approach both are taken proportionally to the square root of the SFS TKE (e),

$$K_\mathrm{m} = c_\mathrm{m} \lambda \tilde{e}^{1/2}, \tag{14a}$$

$$K_\mathrm{h} = c_\mathrm{h} \lambda \tilde{e}^{1/2}, \tag{14b}$$

with λ the characteristic length scale of the SFS turbulent eddies and c_m and c_h proportionality constants. By analogy with the molecular Prandtl number, which is defined as the ratio of the viscosity to the thermal diffusivity, the ratio $K_\mathrm{m}/K_\mathrm{h}$ can be interpreted as a turbulent SFS Prandtl number,

$$\mathrm{Pr_T} = \frac{K_\mathrm{m}}{K_\mathrm{h}} = \frac{c_\mathrm{m}}{c_\mathrm{h}}. \tag{15}$$

The budget equation for $\tilde{e}$ reads,

$$\frac{\partial \tilde{e}}{\partial t} + \tilde{u}_j \frac{\partial \tilde{e}}{\partial x_j} = \frac{g}{\theta_0}\widetilde{w''\theta''_\mathrm{v}} - \widetilde{u''_i u''_j}\frac{\partial \tilde{u}_i}{\partial x_j} - \frac{\partial \widetilde{u''_j e}}{\partial x_j} - \frac{1}{\rho_0}\frac{\partial \widetilde{u''_j p''}}{\partial x_j} - \epsilon, \tag{16}$$

with ρ_0 a reference density and p the pressure. The SFS flux $\widetilde{u''_i u''_j}$ is computed as τ_{ij} following (12), and (13) is used to calculate the SFS fluxes of the θ and q which in turn are used to calculate the SFS buoyancy flux. The total turbulent transport term is computed following a downgradient diffusion approach,

$$\widetilde{u''_j e} + \frac{\widetilde{u''_j p''}}{\rho_0} = -2K_\mathrm{m}\frac{\partial \tilde{e}}{\partial x_j}, \tag{17}$$

and the viscous dissipation of e by molecular viscosity (ϵ) is calculated as

$$\epsilon = c_\epsilon \frac{\tilde{e}^{3/2}}{\lambda}, \tag{18}$$

with c_ϵ a proportionality constant.
In the remainder of the text we will omit the tildes. With this notation the parameterized equation for the SFS TKE can be written as

$$\underbrace{\frac{\partial e}{\partial t}}_{\text{tendency}} + \underbrace{u_j \frac{\partial e}{\partial x_j}}_{\text{mean advection}} = -\underbrace{K_\mathrm{h} N^2}_{\text{buoyancy}} + \underbrace{K_\mathrm{m} S^2}_{\text{shear production}} + \underbrace{2\frac{\partial}{\partial x_j}\left(K_\mathrm{m}\frac{\partial e}{\partial x_j}\right)}_{\text{turbulent transport}} - \underbrace{\frac{c_\epsilon e^{3/2}}{\lambda}}_{\text{dissipation}}, \tag{19}$$

with

$$S^2 \equiv \frac{1}{2}\left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j}\right)^2 = \left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j}\right)\frac{\partial u_i}{\partial x_j}. \tag{20}$$

The classical Smagorinsky model assumes a balance between shear production and dissipation of TKE (Smagorinsky, 1963). Stratification effects can be included by maintaining the buoyancy flux (Mason, 1989),

$$-K_\mathrm{h} N^2 + K_\mathrm{m} S^2 - \frac{c_\epsilon e^{3/2}}{\lambda} = 0. \tag{21}$$

This simplified form of the SFS TKE equation thus neglects the tendency, mean advection and turbulent transport.

5.2 Formulations of the length scale

5.2.1 Constant length scale l_Δ

Deardorff (1973) proposed to use the geometric mean l_Δ of the filter mesh sizes $\Delta x, \Delta y$ and Δz as a representative length scale for SFS eddies,

$$l_\Delta \equiv (\Delta x \Delta y \Delta z)^{1/3}. \tag{22}$$

5.2.2 Stability dependent length scale l_Δ

Deardorff (1980) argued that for a stable stratification the length scale of the eddies may become smaller than the grid size. The vertical stability can be expressed in terms of the Brunt-Väisälä frequency N,

$$N \equiv \left(\frac{g}{\theta_0} \frac{\partial \theta_\mathrm{v}}{\partial z} \right)^{1/2}. \tag{23}$$

The gradient Richardson number $\mathrm{Ri_g}$ is defined by

$$\mathrm{Ri_g} \equiv \frac{N^2}{S^2}. \tag{24}$$

Deardorff proposed the following stability dependent length scale,

$$l_\mathrm{D} = c_\mathrm{n} \frac{e^{1/2}}{N}, \tag{25}$$

to be used only if its magnitude is smaller than l_Δ,

$$\lambda = \min\left(l_\Delta, l_\mathrm{D}\right). \tag{26}$$

For $\lambda = l_\mathrm{D}$ the quantity c_h becomes dependent on the stability,

$$c_\mathrm{h} = \left(c_{\mathrm{h},1} + c_{\mathrm{h},2} \frac{\lambda}{l_\Delta} \right) c_\mathrm{m}, \tag{27}$$

with $c_{\mathrm{h},1} = 1$ and $c_{\mathrm{h},2} = 2$. This approach effectively lets the turbulent Prandtl number depend on the stability, with $\mathrm{Pr_T}$ approaching unity for a very strong stable stratification. The factor c_ϵ is also adapted according to

$$c_\epsilon = c_{\epsilon,1} + c_{\epsilon,2} \frac{\lambda}{l_\Delta}. \tag{28}$$

5.2.3 Mason and Thomson length scale l_M

Last we mention the length scale l_M that was constructed by Mason and Thomson (1992) to let the resulting eddy viscosity better match observed MO similarity relations. Specifically, they proposed

$$\frac{1}{l_\mathrm{M}^n} = \frac{1}{[\kappa(z+z_0)]^n} + \frac{1}{(c_\mathrm{s} l_\Delta)^n}, \tag{29}$$

with z_0 the roughness length. Brown et al. (1994) suggested to use $n = 2$.

5.3 Model constants

In the remainder we will use subscripts 'Δ', 'D' and 'M' to indicate quantities that are derived with the constant length scale l_Δ defined by (22), the stability dependent length scale l_D according to (25) and (26), and the length scale l_M following (29), respectively.
DALES has evolved from the LES code used by Nieuwstadt et al. (1993), and the original setting $c_\mathrm{h} = 3c_\mathrm{m}$ is still used, i.e. $\mathrm{Pr}_\Delta = 1/3$. The factor c_s represents the Smagorinsky constant,

$$c_\mathrm{s} \equiv \left(\frac{c_\mathrm{m}^3}{c_\epsilon} \right)^{1/4} = \frac{c_\mathrm{f}}{2\pi} (\frac{3}{2}\alpha_\mathrm{m})^{-3/4}, \tag{30}$$

with c_f the filter constant and α_m the Kolmogorov constant (see Table 5 for their values used in DALES). We note that there is no general consensus on the optimum values of these quantities, causing differences in the value for the Smagorinsky constant. For example, Lesieur et al. (2005) uses $\alpha_\mathrm{m} = 1.4$, whereas Schumann (1975) and Meneveau and Katz (2000) use a value of 1.5 and 1.6, respectively. As compared to DALES Mason (1989) uses a smaller filter constant of $c_\mathrm{f} = 2$. Kleissl et al. (2003) used an array of sonic anemometers to measure SFS diffusion constants in the atmospheric surface layer. They actually found that c_s is not constant, but is reduced near the ground surface and also tends to become smaller with increasing stability.

c_f	α_m	$c_{\mathrm{h},1}$	$c_{\mathrm{h},1}$	$c_{\epsilon,1}$	$c_{\epsilon,2}$	c_n
2.5	1.5	1	2	0.19	0.51	0.76

Table 5: Summary of constants used in the SFS TKE model. The values are corresponding to the ones used in DALES.

	c_m	c_s	c_h	c_ϵ
l_Δ	0.12	0.22	0.35	0.7

Table 6: Summary of dependent quantities for the SFS TKE model.

6 Shallow Cumulus

In this experiment we will investigate the formation of shallow cumulus clouds. To this end we will again use the set up of our previous CBL simulation including moisture, with an initially stable constant vertical gradient $\partial\theta/\partial z = 6$ K/km, but without large-scale subsidence. However, to allow the formation of clouds we will increase the relative humidity to 80%.

6.1 Conditional sampling

To apply a conditional sampling of updraft fields you will need to add the following data block to `namoptions`,

```
 &NAMSAMPLING
dtav = 60
timeav = 600.
lsampup = .true.
/
```

This asks the model to provide time-averaged statistics at each 600 s on the basis of analyses of the 3D fields at time steps of 60 s. Run the CBL case, and take a look at the updraft statistics.
It is also possible to apply the conditional sampling criterion on clouds or on the cloud core, where the last represents only positively buoyant clouds (see Table 7). Include these options for all your cumulus cloud simulations.

Case 1a. *Shallow cumulus, no precipitation,* $\overline{w'\theta'} = 0.1$ mK/s
Run the case for a period of 10 hours using 64 points in each horizontal direction, and 128 in the vertical direction with $\Delta z = 25$ m. Prescribe constant flux values $\overline{w'\theta'} = 0.1$ mK/s and $\overline{w'q_v'} = 6.67 \cdot 10^{-5}$ m/s. These settings are fairly representative for a case over land.

Case 1b. *Shallow cumulus, no precipitation,* $\overline{w'\theta'} = 0.01$ mK/s
Repeat case 1a, but now with a reduced surface sensible heat flux, $\overline{w'\theta'} = 0.01$ mK/s, which is representative for a situation over the oceans.

6.2 Precipitation

Case 2, Shallow cumulus with precipitation

The model includes two different parameterizations to calculate the formation of rain, the scheme by Khairoutdinov and Kogan (1999) and Seifert and Beheng (2001). The latter scheme is the default one used by Dales. If one wants to apply the Khairoutdinov and Kogan (1999) scheme one should set `l_sb=.false.` in `namoptions`.

Because the rain droplet number concentration and its specific humidity are prognostic variables, the addition of the parameters `nsv=2` and `iadv_sv` to `namoptions` (see Table 7 for an explanation of their meaning and their position in the file) tells the model that these two scalars should be included. In addition, a new input file `scalar.inp.iexpnr` has to be added to the set of requested input files. `scalar.inp.iexpnr` usually contains only zero values for the initial rain water droplet concentration and its specific humidity (see Table 8 for an example of its structure).

Last, because the rain water can reach rather large vertical velocities it is recommended to reduce the maximum time step `dtmax=5` as otherwise the model simulation may crash.

Examples of a `namoptions` and `scalar.inp` files are to be found on brightspace.
Run the case with precipitation by prescribing a cloud droplet concentration of 100×10^6 and 10×10^6 m^{-3}, respectively.

&RUN		
dtmax	= 5	reduce time step to avoid model crashes due to high precipitation fluxes
...		
nsv	= 2	include two extra scalars (rain water and droplet concentration)
/		
&DYNAMICS		
...		
iadv_sv	= 2 2	apply a 2nd order advection scheme to the scalars
/		
&NAMMICROPHYSICS		
imicro	= 2	apply a bulk microphysics scheme
l_sb	= .true.	choose the microphysics parameterization
Nc_0	= 50e6	prescribe a fixed cloud droplet concentration (m^{-3})
/		
&NAMBULKMICROSTAT		
lmicrostat	= .true.	calculate statistics of microphysics
dtav	= 60	time interval to compute slab-averaged statistics
timeav	= 600	averaging time
/		
&NAMSAMPLING		conditional sampling
dtav	= 60	
timeav	= 600.	
lsampup	= .true.	updraft properties
lsampcl	= .true.	cloud properties
lsampco	= .true.	cloud core properties
/		

Table 7: Additional `namoptions` switches for the shallow cumulus case.

```
File usually including zero initial values
height(m) rain water droplet concentration and rain water specific humidity
12.500 0.000 0.000
37.500 0.000 0.000
62.500 0.000 0.000
....
3162.500 0.000 0.000
3187.500 0.000 0.000
```

Table 8: Example of a scalar.inp file. This file usually contains zero values for the initial conditions of the rain water droplet concentration and rain water specific humidity.

7 Instruction notes on the Stable Boundary Layer

7.1 Introduction

So far we have performed experiments with a constant surface flux forcing. Dales also allows to apply time-dependent forcings, which needs to be prescribed in the file `ls_flux.inp`. We will look at some new values of switches in `namoptions`.

7.2 The GABLS1 stable boundary layer model intercomparison case.

The GABLS1 (**G**lobal Energy and Water cycle Exchanges **A**tmospheric **B**oundary **L**ayer **S**tudy) was set up to test the skill of a wide variety of atmospheric models to represent the turbulence structure of the nocturnal boundary layer (Beare et al., 2006). Create a new directory ∼`/Cases/Gabls1` and download the input files to this directory,

```
wget www.srderoode.nl/Teaching/LES_course/Gabls1.tar
```

A `.tar` files collects multiple files in a single file structure. This is handy if you want to copy files to another platform. Useful comments are

`tar -tf Gabls1.tar`	shows the list of the included files
`tar -xvf Gabls1.tar`	unpacks the files
`tar -cf all_files.tar file1 file2`	collects `file1` and `file2` in `allfiles.tar`
`gzip all_files.tar`	compress `all_files.tar`
`gunzip all_files.tar`	uncompress `all_files.tar`

Note that in general information about Linux commands can be readily obtained from the `man` command, e.g. `man tar`

Because we will perform a new series of runs of Gabls1, it is recommended to create a new directory ∼`/Experiments/Gabls1`. For your first test run create a new subdirectory ∼`/Experiments/Gabls1/H000` and prepare this experiment by

```
cp ∼/Cases/Gabls1/prof.inp.gabls1 ∼/Experiments/Gabls1/H000/prof.inp.000
cp ∼/Cases/Gabls1/lscale.inp.gabls1 ∼/Experiments/Gabls1/H000/lscale.inp.000
cp ∼/Cases/Gabls1/ls_flux.inp.gabls1 ∼/Experiments/Gabls1/H000/ls_flux.inp.000
cp ∼/Cases/Gabls1/namoptions.000 ∼/Experiments/Gabls1/H000/namoptions
```

A summary of some key switches of `namoptions` is presented in Table 9.

7.2.1 Adaptive time step

An important difference with the CBL case is that the horizontal grid resolution is much higher for Gabls1. This choice is motivated by the fact that the turbulent eddies for a turbulent stratification are much smaller Run this case to see whether it runs smoothly. Do a second run in which you change enlarge the domain size by change `xsize` and `ysize` to 800 m. Because of the `namoptions` switch `ladaptive=.true` a minimum value of the numerical time step is evaluated at every time step. Check this by comparing the wall clock times for the two experiments. The reason why the coarse horizontal resolution run is much faster can be explained from the Courant-

& RUN		
iexpnr	= 000	
runtime	= 300	short test run
dtmax	= 10	
ladaptive	= .true.	
/		
&DOMAIN		
itot	= 32	
jtot	= 32	
xsize	= 100.	this value gives a horizontal grid resolution $\Delta x = 3.125$ m
ysize	= 100.	
xlat	= 73.	high latitude
/		
&PHYSICS		
ltimedep	= .true.	to indicate time-dependent large-scale forcing
/		
&NAMSURFACE		
isurf	= 2	to indicate that we prescribe the surface temperature as a lower boundary condition
/		
&NAMBUDGET		
lbudget	= .true.	to compute the budget of the Turbulent Kinetic Energy
dtav	= 60.	computes the budget at intervals of 60 s
timeav	= 600.	computes and saves time averaged values over a period of 600 s
/		

Table 9: Key `namoptions` switches for the Gabls1 case.

Friedrichs-Lewy (CFL) stability criterion which for 1D reads,

$$\mathrm{CFL} = u\frac{\Delta t}{\Delta x} < \mathrm{CFL}_{\mathrm{crit}}. \tag{31}$$

For $\mathrm{CFL}_{\mathrm{crit}} = 1$ This criterion can be interpreted as one that states that properties should not travel to an adjacent grid size within one time step Δt. A second criterion checks the turbulent viscosity K_{m} to let the time step satisfy the following cell Peclet number (Pe) condition,

$$\mathrm{Pe} = K\frac{\Delta t}{\Delta x}^2 < \mathrm{Pe}_{\mathrm{crit}}. \tag{32}$$

Dales computes CFL and Pe as follows, respectively,

$$\mathrm{CFL} = \left(\frac{u^2}{\Delta x^2} + \frac{v^2}{\Delta y^2} + \frac{w^2}{\Delta z^2}\right)\Delta t^2 < 1, \tag{33}$$

$$\mathrm{Pe} = \frac{K_{\mathrm{m}}\Delta t}{\min(\Delta x, \Delta y, \Delta z)^2} < 0.15, \tag{34}$$

Note that in Dales $\mathrm{CFL}_{\mathrm{crit}}$ may be smaller for higher-order advection schemes[10]. In conclusion Dales takes the minimum time step following from the Peclet and CFL criteria, and in general larger time steps are permitted for a coarser grid resolution. It is important to stress that these criteria are necessary but do not guarantee stability.

7.2.2 Input file for time-dependent large-scale forcings ls_flux.inp

If the switch `ltimedep` is set to `.true.` dales expects the presence of the file `ls_flux.inp` which contains the time-dependent large-scale forcing conditions. In the previous example of the CBL we set `isurf=4` to prescribe the surface heat fluxes

[10] see the source file `tstep.f90`

and the surface roughness length z_0. Any time dependent surface fluxes can be prescribed in the first data block of `ls_flux.inp`. However, in Gabls1 the surface temperature was prescribed, and its use must be enforced by setting `isurf=2`. Except for `isurf=1` the default setting of Dales is that a saturated surface is assumed. The number of times in `ls_flux` can be chosen arbitrarily and Dales will linearly interpolate the values between the given times.
The next part of `ls_flux` contains vertical profiles of the large-scale processes similar to `lscale.inp`. Please notice that the latter file is always required, although for `isurf=2` its values are overwritten by `ls_flux`. This needs to be improved in a future version of Dales.

```
Gabls1.Hres
time       wtsurf        wqsurf        thls        qts         psurf
[s]        [K m/s]       [kg m/s]      [K]         [kg/kg]     [Pa]
0.0        0.0000e+00    0.0000e+00    265.000     0.000       100000.0
3600.0     0.0000e+00    0.0000e+00    264.750     0.000       100000.0
7200.0     0.0000e+00    0.0000e+00    264.500     0.000       100000.0
10800.0    0.0000e+00    0.0000e+00    264.250     0.000       100000.0
14400.0    0.0000e+00    0.0000e+00    264.000     0.000       100000.0
18000.0    0.0000e+00    0.0000e+00    263.750     0.000       100000.0
21600.0    0.0000e+00    0.0000e+00    263.500     0.000       100000.0
25200.0    0.0000e+00    0.0000e+00    263.250     0.000       100000.0
28800.0    0.0000e+00    0.0000e+00    263.000     0.000       100000.0
32400.0    0.0000e+00    0.0000e+00    262.750     0.000       100000.0


large scale forcing terms
height     ug            vg            wfls        not_used    not_used    dqtdt       dthlrad
#          0.00000
1.56250    8.000         0.000         0.00e+00    0.00e+00    0.00e+00    0.00e+00    0.00e+00
4.68750    8.000         0.000         0.00e+00    0.00e+00    0.00e+00    0.00e+00    0.00e+00
7.81250    8.000         0.000         0.00e+00    0.00e+00    0.00e+00    0.00e+00    0.00e+00
...
...

large scale forcing terms
height     ug            vg            wfls        not_used    not_used    dqtdt       dthlrad
#          32400.0
1.56250    8.000         0.000         0.00e+00    0.00e+00    0.00e+00    0.00e+00    0.00e+00
```

Table 10: Snapshot of the Gabls1 `ls_flux.inp` file.

8 Radiative transfer

year	1980	1990	2000	2010
grid points	64^3	128^3	256^3	512^3
simulation time	4 hrs	$4-10$ hrs	$4-10$ hrs	$\sim$ days
hor. domain size (km^2)	3×3	6×6	25×25	200×200
computing	single CPU		multiple CPU	(ultrafast) GPU
code development		advection schemes		anelastic approach
new physics	turbulence	moist thermodynamics	'warm' microphysics	ice microphysics
		simple radiation		detailed radiation
case	CBL	Cumulus	Precipitating Clouds	Cloud Transitions
		Stratocumulus	Stable BL	Continuous runs
				Deep Convection
key questions	turbulence	entrainment	precipitation	mesoscale organization
				weather prediction
				climate research
				wind and solar energy

Table 11: Summary

Table 11 shows a schematic overview of the development of DALES. So far we have been concerned with LES of turbulence in clear air (CBL, SBL), and in (precipitating) shallow cumulus. These runs have in common that they could all be simulated without interactive radiation. The first simulations of stratocumulus requested the use of a radiation scheme to capture the strong longwave radiative cooling at its top. Duynkerke et al. (1999) proposed a parameterization that only contained a few lines of model code. A few years later Stevens et al. (2005) added some new code to capture the effect of longwave warming near the cloud base, in addition to a longwave radiative cooling of the free troposphere. An analytical solution to the solar radiative transfer problem in clouds was used for simulations of the full diurnal cycle of stratocumulus (Duynkerke et al., 2004). These parameterizations are all present in Dales.

Although these parameterizations are very simple they are rather accurate (see for example Larson et al. (2007)). However, for operational use they have a couple of major disadvantages. First of all, they require a specification of the downwelling flux at the top of the LES domain. This may be obtained by running a full radiative transfer model. Secondly, the schemes are insensitive to the presence of other quantities than liquid water. This is a problem since, for example, the downwelling longwave radiation strongly depends on the temperature and water vapor specific humidity. We will therefore not discuss these parameterizations, but the interested reader can find their descriptions in the literature mentioned above.

As part of the CGILS experiment the Rapid Radiative Transfer Model for General Circulation Models (RRTMG) was implemented in Dales (Blossey et al., 2013). The longwave and shortwave bands are divided in smaller subintervals, and since the scheme is applied to the full atmopheric column it is a computationally expensive code. The radiation code may be called at arbitrary time intervals. Often a radiation time step of a minute is used, and the computed radiative tendencies will be be

assumed to be constant in time until the next radiative flux computation. To reduce the computational time up to about ∼20 years ago global weather forecast models called the radiation scheme only at rather large time intervals of a few hours. The model also contains the so-called Fu-Liou radiative transfer scheme which can be run very cheaply (Pincus and Stevens, 2009).

To make a step towards more realism we will learn how we can include detailed shortwave and longwave radiative transfer calculations with Dales and study its results.

8.1 The atmospheric structure above the LES domain: backrad.inp

The magnitude of the solar radiation flux is changing *throughout* the atmospheric column, from the top of the atmosphere down to the ground surface. A striking effect on solar radiation is its absorption, mainly in the UV waveband, by stratospheric ozone. This implies that in order to make accurate radiation calculations on the one hand the vertical domain should reach the top of the atmosphere. On the other hand, turbulence might only be present in the lower few kilometers, which asks for a high density of grid points in this layer only. In LES models these opposing demands have been nicely compromised by adding a horizontally homogeneous layer on top of the LES domain. The structure of the upper layer is prescribed in a `backrad` file and contains vertical profiles of pressure, temperature, humidity and ozone. In DALES this atmospheric layer is assumed to be stationary, and is used only for radiative transfer calculations. Although the RRTMG and Fu-Liou schemes both contain standard ozone profiles, the user may optionally prescribe a different ozone concentration. In the course we will apply simulations with the Fu-Liou scheme. This is a pragmatic choice as its `backrad` file is in ASCII format whereas RRTMG requests input in a NetCDF format.

An example of a so-called background profile is shown in Table 12. There are a few notable issues:

- Only pressure levels are needed and not height.
- The file requests the temperature.
- Ozone is used only if `usero3 = .true.` in `namoptions`.
- The number of vertical levels can be chosen arbitrarily. More is always better but also more expensive.
- The radiation scheme will always use the full 3D LES computational domain and data from `backrad.nc` for levels above.
- Each LES subcolumn has an identical thermodynamic profile above its top.
- In the current Dales setting the default aerosol concentration is zero, but it can be set in the variable `ztaua` in the subroutine `rrtmg_sw_rad.f90`.

In conclusion, if one is interested in the effect of clouds on radiation, one must choose the LES domain high enough to capture them all.

```
netcdf backrad.ctl_s11_Nz_128 {
dimensions:
 lev = 64 ;
variables:
 float lev(lev) ;
 lev:units = "Pa" ;
 lev:long_name = "isobaric level" ;
 float T(lev) ;
 T:units = "K" ;
 T:long_name = "Temperature" ;
 float q(lev) ;
 q:units = "kg/kg" ;
 q:long_name = "water vapor content" ;
 float o3(lev) ;
 o3:units = "kg/kg" ;
 o3:long_name = "Ozone mass mixing ratio" ;

// global attributes:
 :Title = "CGILS radiation input" ;
data:

 lev = 101967.2, 101674.7, 101269.1, 100736, 100076.6, 99302.08, 98397.48,
 97336.52, 96116.12, 94736.95, 93468.34, 93468.15, 93198.96, 91502.81,
 89652.63, 87655.99, 85520.44, 83252.66, 80861.59, 78359.75, 75759.24,
 73070.53, 70305.65, 67479.39, 64606.13, 61697.88, 58767.62, 55830.45,
 52908.93, 50031.8, 47227.08, 44520.95, 41937.78, 39487.86, 37167.38,
 34971.04, 32892.95, 30927.47, 29069.2, 27312.95, 25653.76, 24086.83,
 22607.61, 21211.71, 19894.93, 18653.28, 17482.86, 16380.03, 15341.24,
 14342.19, 13351.92, 12362.02, 11372.46, 10383.19, 9394.112, 8405.158,
 7416.284, 6427.467, 5438.624, 4449.765, 3460.937, 2472.099, 1483.253,
 494.4198 ;

 T = 291.1213, 290.8827, 290.551, 290.1137, 289.5704, 288.929, 288.1753,
...
 q = 0.01021351, 0.01021351, 0.01021351, 0.01021351, 0.01021351, 0.01021351,
...
 o3 = 4.673502e-08, 4.682244e-08, 4.695263e-08, 4.712739e-08, 4.734801e-08,
...
}
```

Table 12: Example of an `ncdump` of `backrad.nc` used by RRTMG which uses an input file in NCDF format. Fu-Liou contains the same variables, but in ASCII format.

9 Warm start, visualization

9.1 Introduction

In practice one sometimes wishes to take a detailed look at the 3D LES fields, or to compute some additional statistics, for example during during a particular period of time of a previous simulation. The `initd*` files that are written by Dales allow to start a new simulation on the basis of all relevant variables that have been saved during a previous one. Here we will explain how the 'warm' start can be executed, how one can produce 3D fields, and how one can perform additional statistics for an arbitrary period of time on the basis of fields that were generated in a previous run.

9.2 How to perform a warm start

Here we will assume that you have performed a simulation with experiment number `iexpnr` and which has generated files `initdhhhmmmxxprocnryyprocnr.iexpnr`, for example, `initd08h00mx000y000.001`. Here `hh` is the hour at which the data are written, `mm` is the minute, `xprocnr` and `yprocnr` are the processor numbers that contain a part of the horizontal domain area, and `iexpnr` is the number of the experiment. In the standard settings of the model `init*` files are produced after each hour, but smaller time steps are also possible. The `init*` are binary (non-ascii) files, and the data are saved in exactly the same way as they were present in the computer memory. This allows to redoing a part of the simulation identically to the original one.
Let us perform a new simulation. In the following example we will assume a new CBL experiment with number `iexpnr=002` and which uses input files from hour `hh=08` of a previous simulation with number `iexpnr_old=001`. To this end
• Make the directory `~/Experiments/CBL/H002` and copy the dales executable into this directory
`mkdir ~/Experiments/CBL/H002`
`cp ~/dales.build/src/dales4 ~/Experiments/CBL/H002/.`
• Go to the directory `~/Experiments/CBL`
`cd ~/Experiments/CBL` • Because we will perform a warm start, the settings of the new run need to be the same, except for some quantities like the run time. Copy the namoptions file,
`cp H001/namoptions.001 H002/namoptions.002`
• copy the `lscale`, `prof` and `baseprof` input files from H001 to H002 with the appropriate extensions, i.e. `cp H001/prof.inp.001 H002/prof.inp.002`. The model is coded such that it needs to read the vertical levels from `prof.inp`, but for a warm start it will obviously not use the initial vertical proflies of the (thermo-) dynamic variables.
• for each processor field, copy the file of the eight hour,
`cp H001/initd08* H002/.`
The '`*`' is a wildcard and acts to copy all files which names start with `initd08`. This is convenient as it will copy all the input files of hour 8 that were produced by each processor, and will be needed for the new run.

We will need to adapt `namoptions.iexpnr` to let the model know to use the `init*` files. This is done by adding `lwarmstart` and `startfile` in the data block that has the header `&RUN`. The end of the data block iis indicated by '`/`' symbol:

```
&RUN
iexpnr = 002
```

```
.....
lwarmstart = .true.
startfile = 'initd08h00mx000y000.001'
/
```

The model is coded such that each processor will read its own `init*` files of hour `08`. Before adding new information to `namoptions` run this experiment to see if it works.

9.3 Creating 3D fields (Optional exercise)

Now we wish to obtain a snapshot of the atmospheric structure for H001. To this end we will keep working in H002. Change `runtime = 10` and add the following block of data to namoptions:

```
&NAMFIELDDUMP
lfielddump = .true.
klow = 10
khigh = 10
dtav = 10
/
```

This setting will enforce the model to write the fields at the tenth vertical level. In principle $1 \leq$ `klow` $\leq$ `khigh` $\leq$ `kmax`.
If you run the model with these settings you will find that as many `fielddump*` files as processor numbers have been generated. You can find information about the content of file from[11]

• `ncdump fielddump.000.000.002.nc -h`, gives a summary of the variables that are present,
• `ncdump fielddump.000.000.002.nc -v thl`, gives the values of a variable, in this example θ_l

Another handy tool is `ncview`[12] . If you type `ncview fielddump.000.000.002.nc` you can quickly plot a 2D field. However, one can find in the Dales source file `~/dales/src/modfielddump.f90` that the wind velocities have to be divided by a factor 1000 to get them in units m/s. Also, θ_l first has to be multiplied by a factor 100, and consequently a constant factor of 300 has to be added. This data manipulation allows to write data as compact integers, which keeps the `fielddump` files as small as possible.

[11] first install the `ncdump` package from the command `sudo apt install netcdf-bin`
[12] first install the `ncview` package from the command `sudo apt install ncview`

10 Compiling Dales

10.1 Compiling new code

- Store the original file `filename.f90` in a separate directory, i.e. ∼`/Les_versions/Original/filename.f90`. Storing previous versions is convenient in case you have messed up a modified version.
- Put the modified file `filename.f90` in the directory ∼`/dales/src`.
- Go to the directory ∼`/dales.build`.
- Type `make clean`. This will remove all previously compiled code.
- Type `make`. A new executable version of Dales will be build.

10.2 Compilers

To build your dales executable, you can choose how efficient (or fast) it will be. Why shouldn't you always want to get the fastest version possible? The version you are working with uses a high level optimization '`O3`'. This is a fast option, but in case the model crashes, you will not be able to see at which point the model quitted. To keep some relevant printed information in your output file you may want to use '`O0`'. However, the bookkeeping the model performs on the fly comes at the expense of computational speed.
To build a script that compiles your code you could take the following steps. First, you have to make a directory in which you want to store executable. It is convenient not to do this in the directory where you have stored your source code, as this is linked to git. So let's call it `dales.build`. Go to this directory and type `ccmake` ∼`/dales`. Then you will enter an environment with at the bottom the following text block:

```
EMPTY CACHE:
Press [enter] to edit option CMake Version 3.3.2
Press [c] to configure
Press [h] for help Press [q] to quit without generating
Press [t] to toggle advanced mode (Currently Off)
```

Next type `c` and `t` to configure and to switch to toggle advanced mode, respectively. You can change the code optimization at `CMAKE_Fortran_FLAGS_RELEASE` by using your keyboard arrows to go to the desired option, after which you enter the Return key. You can now change the settings. Enter the Return key to confirm, then `c` and `g` to confirm the changes and to generate a `Makefile`, respectively.
The `cmake` script chooses the Intel compiler by default. This is controled by the variable `SYST='localpc_ifort'`. If you want to use `gcc` you have rest the value of `SYST` by `unset SYST`.

11 Model crashes

*** STOP ERROR:Namoptions does not exist**

you likely have a file namoptions.extension but it should be just namoptions (without `iexpnr`.

* At line 380 of file /home/dales/dales/src/modglobal.f90 (unit = 1, file = 'prof.inp.000')
Fortran runtime error: End of file

This means that possibly
- the file `prof.inp.000` does not exist
- the number of levels in `prof.inp` does not match with `kmax`
- the experiment number in `namoptions` differs from the `*inp` extension of the input files

Level height requirements The staggered grid of the model assumes a structure like $z_\mathrm{f}(1) = \Delta z/2, z_\mathrm{f}(2) = 1.5\Delta z$ etc. A commonly made mistake is to set $z_\mathrm{f}(1) = 0, z_\mathrm{f}(2) = \Delta z$, or $z_\mathrm{f}(1) = \Delta z, z_\mathrm{f}(2) = 2\Delta z$. In the latter two cases the model will crash.

A Linux tips

It is possible to open multiple terminals, for example one for the directory ∼/Cases and one for ∼/Experiments. If you want to quickly change from one directory to another, you can define a shortcut, e.g. `alias exp='cd ∼/Experiments'`. Put this line in the file ∼/.bashrc and type `source ∼/.bashrc` to activate the alias. Note that .bashrc will be executed automatically when you open a new terminal. The .bashrc file can be filled with as many aliases as you want.
Ubuntu includes an open source version of Matlab which is called Octave.

B Installing Dales on a Mac

It is possible to run dales directly from your Mac, without the need of VirtualBox. To this end you will need to install some packages. Note that this takes some time (time scale ∼1 hour), but it is worth the effort.
Install Xcode which is available for free from Mac Appstore. Once installed it will ask if you want to create a project, this is not needed. Note that installation of some packages with brew requires the installation of xcode as well. xcode in turn may require that OSX is up to date, which may require a large download. Do this step before class if possible.
Next, open a terminal and type the following commands,

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"[13]
brew install git wget openmpi netcdf cmake[14]
cd
git clone https://github.com/dalesteam/dales.git[15]
cd dales
git checkout v4.1_educational
cd ../
mkdir dales.build
cd dales.build
cmake ../dales[16]
make
```

The first command (the one with 'ruby' sometimes gives rise to the popping up of a larger than symbol '>'). The instructions above are also posted on the Dales website dales-portal.eu, and if you copy the 'ruby' command to your terminal this problem will disappear.

C Installing Dales on Ubuntu 16.04

First install the needed packages:

```
sudo apt install git
sudo apt install cmake
sudo apt install netcdf-dev
sudo apt install netcdf
sudo apt install ccmake
sudo apt install cmake-curses-gui
sudo apt install libnetcdf-dev
sudo apt install libopenmpi-dev
sudo apt install gfortran
sudo apt install libnetcdff-dev
sudo apt install doxygen
```

[13] see https://brew.sh
[14] installs the packages required for dales
[15] downloads dales from the git repository
[16] if the model does not compile, you may try `brew upgrade` to ensure you work with the latest version

Note that for more recent versions of Ubuntu you can type `sudo apt install libnetcdf-dev libnetcdff-dev libopenmpi-dev openmpi-bin gfortran make cmake git libhypre-dev fftw-dev`. Next, clone the Dales git repository

```
cd ∼
git clone git@github.com:dalesteam/dales.git, works only if you have a github account, else use
git clone https://github.com/dalesteam/dales.git
cd dales
git checkout v4.1
```

Now build Dales from source code

```
cd
mkdir dales.build
cd dales.build
cmake ../dales
make
```

The dales4 executable is now in `src/dales`.

D Installing Dales on the VRLab system

If you have an account on the computer cluster 'VRlab' you can perform runs using multiple cores. It contains 4 Intel(R) Xeon(R) CPUs E5520@2.27GHz, each with 16 cores, and 8 Intel(R) Xeon(R) CPU E5-2695 v4@2.10GHz, each with 18 cores.
Remote logon is possible from every internet connected machine via ssh. The external address of our bastion host is ?bastion-grs.vrlab.tudelft.nl" . You can login into the system with

`ssh -Y login@bastion-grs.vrlab.tudelft.nl`,

with `login` your username. On this machine, all disks are mounted, to allow access to all your files. However, there is no software installed on the bastion machine. In order to run applications, you need to logon from the bastion, using ssh, to your VRLab desktop machine or one of the public machines available, for example shell-1 or shell-2:

```
ssh login@shell-1
```

To be able to run Dales you will have to create a file called `.bashrc` with the following contents

```
export ARCH=linux
export SYST='localpc_ifort'

module load intel
module load hdf5/intel
module load netcdf/intel
module load netcdf-fortran/intel
module load openmpi/intel
```

This file will be executed after you have logged in. However, if you have just made this file, you can execute it with

```
source .bashrc
```

To download and compile Dales you can apply the steps as in Appendix C (for Ubuntu). If you are going to do research with the model, you might skip the step `git checkout v4.1` which version has been proposed in particular for the present course.
For running the model it is recommended to do this on Labdata, which is a big storage server, and which can be expanded to Petabyte sizes if the need arises. This is the place to store your data from any source. This folder can be made accessible to your supervisor if you are a student. As you will likely work on Labdata, it will be convenient to add to your `.bashrc` file the alias `alias lab='cd /net/labdata/login'`.
On labdata, you can make a directory called like `Experiments/Case`, and for your first simulation

a 'working' directory like `Experiments/Case/H001`, with the next ones `H002` etc. In your working directory you should put the `dales4` executable, all the necessary `*inp` files, and `namoptions`. For large runs one is requested to submit jobs to the SLURM queuing system[17], with

```
sbatch job.001
```

and if you want to use a number of `Ncores` (an integer value of the number of requesed cores) the contents of your job file should read like

```
#!/bin/bash
#SBATCH --ntasks=Ncores
#SBATCH --cpus-per-task=1
#SBATCH --ntasks-per-node=Ncores
#SBATCH --partition inf
#SBATCH --time=5000000
#SBATCH --mem=10G
#SBATCH -J wjob.001
#SBATCH --output
#SBATCH --error

ulimit -s unlimited

mpirun dales4
```

Once submitted, you can check the status of your job by typing `squeue`.

E Error messages

If your model gives an `end of file` error while reading an input file this may be due to an inconsistent value for the number of vertical levels in `namoptions` (`kmax`) with respect to number of levels in your input file, or the input file may not exist at all.

[17] see `https://slurm.schedmd.com/quickstart.html` for further instructions, for example about how to delete your running job

References

Beare, R. J., and Coauthors, 2006: An intercomparison of large-eddy simulations of the stable boundary layer. *Boundary-Layer Meteorol.*, **118**, 247–272.

Blossey, P. N., and Coauthors, 2013: Marine low cloud sensitivity to an idealized climate change: The CGILS LES intercomparison. *J. Adv. Model. Earth Syst.*, **5**, 1–25.

Boers, R., T. Brandsma, and A. P. Siebesma, 2017: Impact of aerosols and clouds on decadal trends in all-sky solar radiation over the netherlands (1966?2015). *Atmos. Chem. Phys.*, **17**, 8081–8100.

Böing, S. J., 2014: The interaction between deep convective clouds and their environment. Ph.D. thesis, Technical University Delft, Delft, 133 pp (Available from Technical University Delft, Delft, The Netherlands, http://repository.tudelft.nl).

Brown, A. R., S. Derbyshire, and P. J. Mason, 1994: Large-eddy simulation of stable atmospheric boundary layers with a revised stochastic subgrid model. *Quarterly Journal of the Royal Meteorological Society*, **120 (520)**, 1485–1512.

Deardorff, J. W., 1973: Three-dimensional numerical modeling of the planetary boundary layer. *Workshop on Micrometeorology*, D. A. Haugen, Ed., Amer. Met. Soc. Boston, 271–311.

Deardorff, J. W., 1980: Cloud-top entrainment instability. *J. Atmos. Sci.*, **37**, 131–147.

Duynkerke, P. G., and Coauthors, 1999: Intercomparison of three- and one-dimensional model simulations and aircraft observations of stratocumulus. *Boundary-Layer Meteorol.*, **92**, 453–487.

Duynkerke, P. G., and Coauthors, 2004: Observations and numerical simulations of the diurnal cycle of the EUROCS stratocumulus case. *Quart. J. Roy. Meteorol. Soc.*, **130**, 3269–3296.

Heus, T., and Coauthors, 2010: Formulation of the dutch atmospheric large-eddy simulation (DALES) and overview of its applications. *Geosci. Model Development*, **3**, 415–444, doi: 10.5194/gmd-3-415-2010.

Khairoutdinov, M. F., and Y. L. Kogan, 1999: A large eddy simulation model with explicit microphysics: Validation against aircraft observations of a stratocumulus-topped boundary layer. *Journal of the atmospheric sciences*, **56 (13)**, 2115–2131.

Kleissl, J., C. Meneveau, and M. B. Parlange, 2003: On the magnitude and variability of subgrid-scale eddy-diffusion coefficients in the atmospheric surface layer. *J. Atmos. Sci.*, **60**, 2372–2388.

Larson, V. E., K. E. Kotenberg, and N. B. Wood, 2007: An analytic longwave radiation formula for liquid layer clouds. *Mon. Weather Rev.*, **135**, 689–699.

Lesieur, M., O. Métais, and P. Comte, 2005: *Large-eddy simulations of turbulence*. Cambridge University Press.

Mason, P. J., 1989: Large-eddy simulation of the convective atmospheric boundary layer. *J. Atmos. Sci.*, **46**, 1492–1516.

Mason, P. J., and D. J. Thomson, 1992: Stochastic backscatter in large-eddy simulations of boundary layers. *J. Fluid Mech.*, **242**, 51–78.

Meneveau, C., and J. Katz, 2000: Scale-invariance and turbulence models for large-eddy simulation. *Annu. Rev. Fluid Mech.*, **32**, 1–32.

Nieuwstadt, F. T., P. J. Mason, C.-H. Moeng, and U. Schumann, 1993: Large-eddy simulation of the convective boundary layer: A comparison of four computer codes. *Turbulent Shear Flows 8*, Springer, 343–367.

Pincus, R., and B. Stevens, 2009: Monte Carlo spectral integration: a consistent approximation for radiative transfer in large eddy simulations. *J. Adv. Model. Earth Syst.*, **1**, doi:10.3894/JAMES. 2009.1.1.

Schumann, U., 1975: Subgrid scale model for finite difference simulations of turbulent flows in plane channels and annuli. *J. Comp. Phys.*, **18**, 376–404.

Seifert, A., and K. D. Beheng, 2001: A double-moment parameterization for simulating autoconversion, accretion and selfcollection. *Atm. Res.*, **118**, 265–281–290, doi:10.1016/S0169-8095(01) 00126-0.

Smagorinsky, J., 1963: General circulation experiments with the primitive equations. Part I: The basic experiment. *Mon. Weather Rev.*, **91**, 99–164.

Stevens, B., and Coauthors, 2005: Evaluation of large-eddy simulations via observations of nocturnal marine stratocumulus. *Mon. Weather Rev.*, **133**, 1443–1462.