Wind Energ. Sci., 1, 55–70, 2016
www.wind-energ-sci.net/1/55/2016/
doi:10.5194/wes-1-55-2016

eawe european academy of wind energy
Open Access
WIND ENERGY SCIENCE

Comparison of OpenFOAM and EllipSys3D for neutral atmospheric flow over complex terrain

Dalibor Cavar[1], Pierre-Elouan Réthoré[1], Andreas Bechmann[1], Niels N. Sørensen[1], Benjamin Martinez[2], Frederik Zahle[1], Jacob Berg[1], and Mark C. Kelly[1]

[1]Technical University of Denmark, Wind Energy Department, Risø Campus, 4000 Roskilde, Denmark
[2]Vattenfall Nordic R & D, 7000 Fredericia, Denmark

Correspondence to: Dalibor Cavar (daca@dtu.dk)

Received: 24 February 2016 – Published in Wind Energ. Sci. Discuss.: 10 March 2016
Accepted: 28 April 2016 – Published: 20 May 2016

Abstract. The flow solvers OpenFOAM and EllipSys3D are compared in the case of neutral atmospheric flow over terrain using the test cases of Askervein and Bolund hills. Both solvers are run using the steady-state Reynolds-averaged Navier–Stokes k–ϵ turbulence model.

One of the main modeling differences between the two solvers is the wall-function approach. The OpenFOAM v.1.7.1 uses a Nikuradse's sand roughness model, while EllipSys3D uses a model based on the atmospheric roughness length. It is found that Nikuradse's model introduces an error dependent on the near-wall cell height. To mitigate this error the near-wall cells should be at least 10 times larger than the surface roughness. It is nonetheless possible to obtain very similar results between EllipSys3D and OpenFOAM v.1.7.1. The more recent OpenFOAM v.2.2.1, which includes the atmospheric roughness length wall-function approach, has also been tested and compared to the results of OpenFOAM v.1.7.1 and EllipSys3D.

The numerical results obtained using the same wall-modeling approach in both EllipSys3D and OpenFOAM v.2.1.1 proved to be almost identical.

Two meshing strategies are investigated using HypGrid and SnappyHexMesh. The performance of OpenFOAM on SnappyHexMesh-based low-aspect-ratio unstructured meshes is found to be almost an order of magnitude faster than on HypGrid-based structured and high-aspect-ratio meshes. However, proper control of boundary layer resolution is found to be very difficult when the SnappyHexMesh tool is utilized for grid generation purposes.

The OpenFOAM is generally found to be 2–6 times slower than EllipSys3D in achieving numerical results of the same order of accuracy on similar or identical computational meshes, when utilization of EllipSys3D default grid sequencing procedures is included.

1 Introduction

Wind resource assessment is of major importance to assess the economic viability of a wind farm project. The traditional tools used by the wind industry rely on linearized flow models and do not perform accurately in complex terrain (Bechmann et al., 2011). Thus, there is a growing interest from the wind industry to use full nonlinear computational fluid dynamics (CFD) solutions.

Several CFD solvers are currently available to the industry. Among them, OpenFOAM (OpenFOAM, 2016) has recently received a lot of interest from both the research community and the wind industry. Its popularity seems to be firstly caused by its gratuity and its open-source license, which is in strong contrast to most industry-oriented CFD solvers, and secondly by its good performance[1], e.g., at the Bolund blind comparison (Berg et al., 2011; Bechmann et al., 2011). OpenFOAM therefore directly offers accurate results at no licensing costs.

[1]Using a non-default, Richards and Hoxey (1993)-based, user-implemented wall-function formulation.

Two questions are, however, unanswered: how easily and how fast OpenFOAM can produce accurate results. These two questions are of course relative to other flow solvers, but they are relevant and important for the wind industry, as they can represent substantial costs in terms of both necessary hardware and manpower required, due to potentially extensive meshing and computational times. In order to address these questions, OpenFOAM is compared with EllipSys3D (Sørensen, 1995; Michelsen, 1992). EllipSys3D is an in-house CFD flow solver designed from the ground up for wind energy applications (e.g., atmospheric boundary layer flows, flow over complex terrain and wind turbine rotor computations) at DTU Wind Energy (formerly Risø National Laboratory; Sørensen, 1995) and DTU-MEK (Michelsen, 1992).

The comparison presented here is focused on the mesh requirements of OpenFOAM relative to EllipSys3D and how this can affect the accuracy of the results. Furthermore, the two CFD codes are benchmarked in speed on the computer cluster at DTU Wind Energy, utilizing two Intel Xeon® nodes (2×12 CPU cores) connected with InfiniBand interconnect.

In order to keep the comparison as close as possible, the two flow solvers are run, to the best possible extent, using the same models and parameters. The main model difference between the two flow solvers is the wall function applied for modeling the effect of the ground roughness on the flow. EllipSys3D (Sørensen et al., 2007a) uses a wall function based on the atmospheric roughness length (Richards and Hoxey, 1993), while OpenFOAM v.1.7.1 uses the Nikuradse sand roughness model (Nikuradse, 1950). In OpenFOAM v.2.1.1 a wall function based on the atmospheric roughness length (Richards and Hoxey, 1993) is also implemented. It should be noted that difference in wall-function modeling has a significant impact on the mesh requirement concerning the height of the first cell above the ground level.

2 Methods

2.1 Basic equations

Both EllipSys3D and OpenFOAM are based on the finite-volume solution of the Reynolds-averaged Navier–Stokes (RANS) equations. If molecular viscosity is neglected due to the high Reynolds number and the eddy-viscosity hypothesis of Boussinesq (1877) is followed, then the RANS equations can be written as

$$\frac{\partial \overline{\boldsymbol{u}}_i}{\partial t} + \frac{\partial \left(\overline{\boldsymbol{u}}_i \overline{u}_j\right)}{\partial x_j} = -\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(2\nu_T \overline{S}_{ij}\right) + \overline{f}_i, \tag{1}$$

where the Einstein summation notation is used. $\overline{\boldsymbol{u}}_i = (\overline{u}; \overline{v}, \overline{w})$ denotes the mean velocity vector; $x_i = (x, y, z)$ are axes of the coordinate system, with z being the vertical direction; $\overline{p}$ is the dynamic pressure; $\overline{f}_i$ represents body forces; $\overline{S}_{ij}$ is the strain rate tensor; and ν_T is the eddy viscosity which needs to be modeled. The transient term of Eq. (1) has been retained even though the equations in this work are solved in the steady-state mode.

The classical two-equation high Reynolds number k–ϵ model (Launder and Spalding, 1974) is utilized in both flow solvers to calculate the eddy viscosity. Transport equations for the turbulent kinetic energy, k, and its dissipation rate ϵ, used by k–ϵ turbulence model are solved and have the following form:

$$\frac{\partial k}{\partial t} + \frac{\partial}{\partial x_j}\left(\overline{u}_j k\right) - \frac{\partial}{\partial x_j}\left(\frac{\nu_T}{\sigma_k}\frac{\partial k}{\partial x_j}\right) = \nu_T |\overline{S}|^2 - \epsilon, \tag{2}$$

$$\frac{\partial \epsilon}{\partial t} + \frac{\partial}{\partial x_j}\left(\overline{u}_j \epsilon\right) - \frac{\partial}{\partial x_j}\left(\frac{\nu_T}{\sigma_\epsilon}\frac{\partial \epsilon}{\partial x_j}\right) = C_{\epsilon 1}\frac{\epsilon}{k}\nu_T |\overline{S}|^2 - C_{\epsilon 2}\frac{\epsilon^2}{k}. \tag{3}$$

The originally proposed model constants by Launder and Spalding (1974) were established for industrial flows, while slightly different values are appropriate for atmospheric flows (Bechmann and Sørensen, 2010). Two test cases are simulated in the present work: Askervein Hill and Bolund Hill. Calibrated model constants are used for Askervein Hill, while standard atmospheric values are used for Bolund Hill (see Tables 1 and 4). Identical constants have been used in both EllipSys3D- and OpenFOAM-based calculations.

2.2 Boundary conditions

In order to model the effect of surface roughness and avoid resolving the laminar sublayer, it is common practice to use the wall function for atmospheric flows. One of the major differences between EllipSys3D and OpenFOAM v.1.7.1 is related to how the wall function is implemented. In the more recent OpenFOAM v.2.1.1 the same wall function as the one in EllipSys3D is included in the official release.

2.2.1 EllipSys3D

In EllipSys3D the traditional high Reynolds number equilibrium assumptions are used to derive the wall function (see Sørensen, 1995; Sørensen et al., 2007b; and Hackman, 1982, for details). These inherently neglect the laminar sublayer, which usually is less than a millimeter thick for atmospheric flows. The logarithmic equilibrium profiles used for the mean wind speed, u, and turbulent kinetic energy are

$$u = \frac{u_*}{\kappa}\ln\left(\frac{z + z_0}{z_0}\right), \tag{4}$$

$$k = \frac{u_*^2}{C_\mu^{1/2}}, \tag{5}$$

where z_0 is the roughness length, $\kappa = 0.40$ is the von Kármán constant and u_* is the friction velocity.

As seen from Eq. (4), the wall is placed on top of the roughness elements ($u = 0$ for $z = 0$) and is consequently displaced by the roughness length. This has been done to avoid a minimum height restriction of the first computational cell

(see Sect. 2.2.2), and EllipSys3D can thereby resolve large near-wall velocity gradients using shallow (high aspect ratio) computational cells.

In order not to abandon the momentum equation in the near-wall cell by simply fixing the velocity according to log law (Eq. 4), EllipSys3D instead implements the wall function through the wall shear stress, τ_0. Based on Eqs. (4) and (5) the shear stress is calculated from $\tau_0 = \rho\, u_{w_1}\, u_{w_2}$ using

$$u_{w_1} = k^{\frac{1}{2}} C_\mu^{\frac{1}{4}}, \tag{6}$$

$$u_{w_2} = \frac{\kappa u}{\ln\left(\frac{\Delta z + z_0}{z_0}\right)}, \tag{7}$$

where Δz is the distance from the bottom cell face to the cell center. The ϵ equation is abandoned in the near-wall cells; instead, ϵ is specified according to

$$\epsilon = \frac{u_{w_1}^3}{\kappa\,(\Delta z + z_0)}, \tag{8}$$

while the k equation is reduced to a balance between production ($P_k = \nu_T |\overline{S}|^2$) and dissipation rate ϵ and a zero-gradient boundary condition is set for k. The boundary condition for the velocity is the standard no-slip condition ensuring the velocity to be zero at the wall.

2.2.2 OpenFOAM

OpenFOAM v.1.7.1 – Nikuradse's equivalent sand roughness length model

The wall function available in OpenFOAM v.1.7.1 is based on Nikuradse's equivalent sand roughness length, k_s (Nikuradse, 1950), and can be used to model flows where the laminar sublayer is important. The OpenFOAM implementation of an atmospheric equilibrium boundary-layer condition, according to the expression for surface roughness of Nikuradse, reduces to the following equation for the mean wind speed[2] (Martinez, 2011; Blocken et al., 2007),

$$u = \frac{u_*}{\kappa} \ln\left(\frac{E z}{C_s k_s}\right) \approx \frac{u_*}{\kappa} \ln\left(\frac{z}{z_0}\right), \tag{9}$$

where $C_s = 0.5$ is a roughness constant, $E = 9.79$ is a smooth wall constant and $k_s = \frac{E}{C_s} z_0 = 19.58\, z_0$ has been used.

Comparing with Eq. (4) it is seen that the roughness for OpenFOAM v.1.7.1 is placed on top of the wall ($u = 0$ for $z = z_0$). While this is physically the "correct" approach, it does set some constraints on the heights of the near-wall cells. Firstly, as argued in Blocken et al. (2007), it is not physically meaningful to have grid cells within the roughness height; therefore, the height of the near-wall cell center should be at least equal to or larger than the Nikuradse's roughness length ($\Delta z \geq k_s$). Blocken et al. (2007) do, however, state that it is mathematically/numerically possible to have $\Delta z \leq k_s$, and comparison between measurements and a sand-grain-based velocity function fit (Sumer, 2007) shows that their agreement is very good when $\Delta z \geq 0.2\, k_s$. Secondly, OpenFOAM has a restriction on the cell aspect ratio, i.e., the ratio between the longest and the shortest cell dimension. Meshes with cell aspect ratio larger than 1000 have a tendency to introduce numerical errors and to converge slower (Martinez, 2011) and it is therefore difficult to make shallow cells that resolve the flow close to the ground. As a compromise between the cell constraints and the need to resolve the near-wall flow, $\Delta z / z_0 \approx 10.0$ was found to generally give the least numerical error for flat-terrain simulations (Martinez, 2011) and is used in the following.

Using the built-in routine (`nutRoughWallFunction`), OpenFOAM v.1.7.1 implements the wall function by specifying the eddy viscosity, which, for fully rough flows, can be written as

$$\nu_T = \frac{u_* \kappa \Delta z}{\ln\left(\frac{E \Delta z}{C_s k_s}\right)} \approx \frac{u_{w_1} \kappa \Delta z}{\ln\left(\frac{\Delta z}{z_0}\right)}. \tag{10}$$

Similar to EllipSys3D, the ϵ equation is abandoned in the near-wall cells. Using the built-in function (`epsilonWallFunction`), ϵ is set to

$$\epsilon = \frac{u_{w_1}^3}{\kappa \Delta z}, \tag{11}$$

while a zero-gradient condition is set on k (using the `kqRWallFunction`). A no-slip condition is set on the velocity vector.

OpenFOAM v.2.1.1 – Richards and Hoxey (1993)-based model

The model is identical to the model implemented in EllipSys3D. Regarding the OpenFOAM implementation, the only difference compared to Nikuradse's roughness length model is in the way turbulent viscosity is determined.

Using the built-in routine (`nutkAtmRoughWallFunction`), OpenFOAM v.2.1.1 implements the wall function by specifying the eddy viscosity as

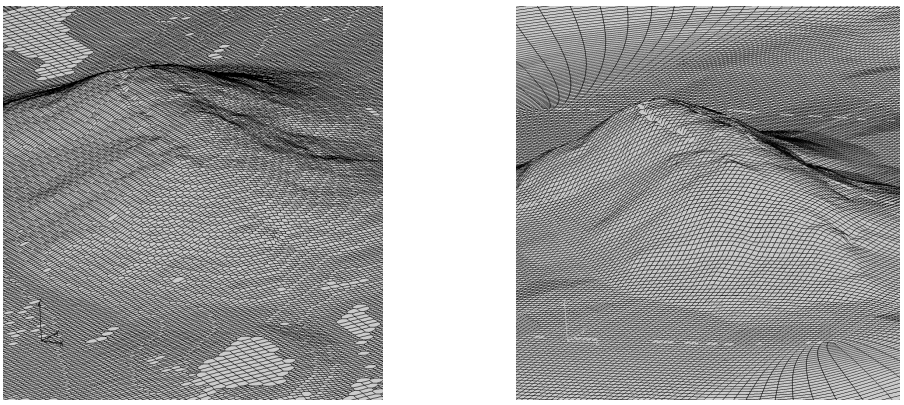
$$\nu_T = \frac{u_{w_1} \kappa \Delta z}{\ln\left(\frac{\Delta z + z_0}{z_0}\right)}. \tag{12}$$

All other boundary conditions used in connection with this wall model are identical to the ones described in the previous paragraph.

2.3 Solution techniques

In the simulations carried out in this work, in both flow solvers, the RANS equations are discretized using

[2] Assuming $C_s k_s^+ \gg 1$, where $k_s^+ = \frac{u_* k_s}{\nu}$ and ν is a molecular viscosity.



(a) (b)

Figure 1. Askervein Hill: surface grid generated with **(a)** SnappyHexMesh and **(b)** surfgrid.

the QUICK scheme (Leonard, 1979) (`QUICKV`, the vectorial version for OpenFOAM). The k and ϵ equations are discretized using the first-order Upwind Discretization Scheme (UDS). The pressure is solved using the SIMPLE algorithm (Patankar and Spalding, 1972) and is accelerated using a multigrid approach (`GAMG` for OpenFOAM).

2.4 Mesh generators

2.4.1 HypGrid

EllipSys3D uses the mesh generator HypGrid (Sørensen, 1998), a hyperbolic mesh generator developed at DTU Wind Energy. HypGrid creates a 3-D structured hexahedron volume mesh using a hyperbolic marching scheme, based on orthogonality and cell volume from a 3-D terrain grid surface definition. It can produce meshes with cells of low non-orthogonality and low skewness. There is no constraint on cell aspect ratio.

2.4.2 SnappyHexMesh

SnappyHexMesh is a hexahedral unstructured mesh generation tool included in the distribution of OpenFOAM. SnappyHexMesh can use a 3-D terrain surface and iteratively build a mesh upon it. Some options allow construction of several cell layers of a controllable height above the terrain surface. This feature makes it possible to have a refined mesh in the region of high velocity gradients, close to the ground. Several regions can be selected to be refined to a desired level, and this method is creating a mesh of a cell aspect ratio close to 1. This is done in order to ensure that OpenFOAM can solve the numerical problems with the highest efficiency and accuracy using meshes generated with SnappyHexMesh. The meshing tool SnappyHexMesh can be run in parallel on a computer cluster or a PC.

Table 1. Set of atmospheric parameters used for Askervein, as described in Taylor and Teunissen (1987).

u_* [m s^{-1}]	z_0 [m]	κ	C_μ	σ_k	σ_ε	$C_{\varepsilon 1}$	$C_{\varepsilon 2}$
0.6110	0.03	0.4	0.119	1.000	1.301	1.564	1.920

3 Results

3.1 Askervein

3.1.1 Simulation inputs

The inputs used to simulate Askervein Hill are fitted based on the upstream mast measurements.

The complete set of all simulation inputs, in accordance with Taylor and Teunissen (1987), is found in Table 1.

3.1.2 Description of the mesh

HypGrid

The Askervein map is used as input to the in-house tool "surfgrid". The surfgrid tool generated a circular (32.5 km in diameter) surface mesh, with a refined region of 2 km × 2 km (4 blocks) in the center, at the position where Askervein Hill is located (see Fig. 1b). From this surface mesh the HypGrid tool is used to hyperbolically march the surface into a 6.5 km high 3-D mesh. The final 3-D volume mesh is composed of 20 blocks of $64 \times 64 \times 64$: 5.2 million cells (see Fig. 2b).

Two meshes are created with this method, one with a first cell center height of $0.4\,\mathrm{m} = 13.3\,z_0$ (ideal for OpenFOAM v.1.7.1 – the HG1 grid), and one with a first cell center height equal to a half of the roughness length $z_0 = 0.03$ m (ideal for EllipSys3D – the HG2 grid). Two additional meshes (with the first cell center height of $\Delta z = 0.83\,z_0$ and $\Delta z = 1.5\,z_0$) were created for OpenFOAM v.2.1.1, but they both failed several grid compliance tests performed by the `checkMesh` OpenFOAM tool. A converged solu-

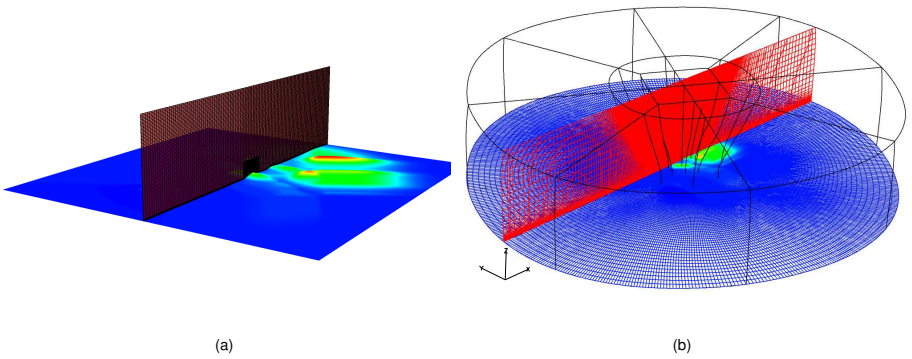


Figure 2. Askervein Hill: volume grid generated with **(a)** SnappyHexMesh and **(b)** HypGrid.

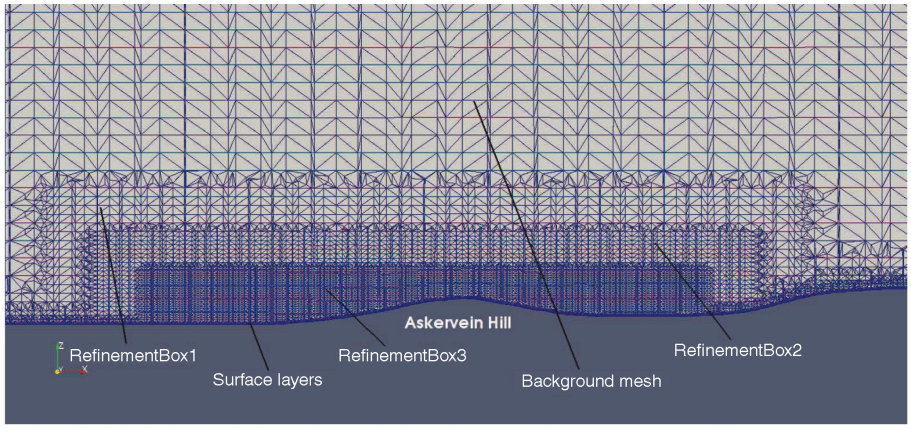


Figure 3. Askervein Hill: cross section of the grid created by SnappyHexMesh. Three refinement boxes together with variations in number of surface layers indicated in the figure are used as a basis for generation of the grids described in Table 2.

tion could, however, be obtained on those grids only if the `mapFields` tool is used to, for example, interpolate the final solution obtained on the grid with the first cell center height at $\Delta z = 13.3\, z_0$. As the proper speed convergence tests could not be conducted on the mentioned grids, the numerical results based on them are not included here. The coarse HG0 grid (Table 2) is created exclusively to accommodate a proper grid sequencing procedure in the OpenFOAM HypGrid-based computations.

SnappyHexMesh

The surfgrid-generated surface grid used for HypGrid meshing is converted to STL file format and utilized to generate a ground boundary surface suitable for the SnappyHexMesh OpenFOAM utility. The SnappyHexMesh-generated surface mesh close-up view is shown in Fig. 1a. A rectangular domain of $11.03 \times 11.03 \times 3.10$ km is discretized using the blockMesh utility (see Table 2), creating a background mesh with a resolution of $95.9 \times 95.9 \times 38.75$ m in the x, y, and z directions, respectively. Only the SHM4 mesh (Table 2) has a background mesh resolution of 77.5 m in the z direction. The cross section of the SnappyHexMesh-created (SHM4) volume grid can be seen in Fig. 2a. Indicators, locating positions of refinement boxes and surface layers used to generate meshes SHM(0–4), are presented in Fig. 3. Changing refinement levels in refinement boxes 1–3 together with the number of inserted surface layers are the controlling parameters used in the SnappyHexMesh grid creation process (Table 2). The `refinementSurface` and `resolveFeatureAngle` parameters are used to control the surface refinement level relative to the background grid. The coarse SHM0 (Table 2) grid is only intended for use in connection with grid sequencing procedures.

As can be seen in Table 2, grids with optimal position of the first near-ground cell heights, suitable for both OpenFOAM v.1.7.1 and v.2.1.1, could be created in this way.

3.1.3 Simulation results

The main results of the Askervein test case are presented in Figs. 4 and 5. Both figures show the results of the simulations compared with the cup anemometer measurements along the line A – see Taylor and Teunissen (1987). The basic simu-

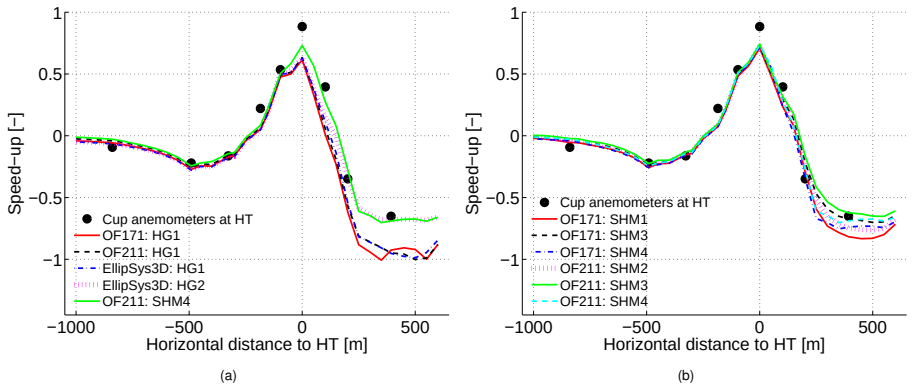


Figure 4. Askervein Hill: speedup along the line A.

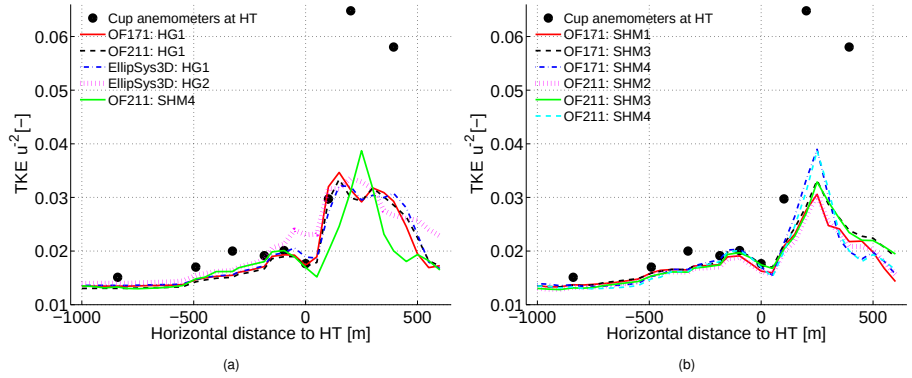


Figure 5. Askervein Hill: turbulent kinetic energy along the line A.

lations are carried out on HypGrid-based grids HG(1–2) and SnappyHexMesh-based grids SHM(1–4). In Figs. 4a and 5a EllipSys3D calculation on structured mesh HG1 and OpenFOAM (both v.1.7.1 and v.2.1.1) computations on the same mesh are presented. The lines in red, black and blue therefore represent the results of OpenFOAM and EllipSys3D calculations using the identical mesh.

Overall, the results of all simulations are very similar to the measurements in the region upstream of the hilltop, both in terms of speedup and turbulent kinetic energy (TKE).

After the hill-top however, results of the simulations start to differ from each other significantly, in terms of speedup and to a lesser extend in terms of TKE. In comparison with the measurements, the speedup results are relatively close to the EllipSys3D simulation, using the mesh with a first cell height equal to the surface roughness (HG2 grid – dotted magenta line – Fig. 4a) and OpenFOAM v.2.1.1 SnappyHexMesh SHM(2–4)-based simulations – Fig. 4b. Both OpenFOAM (v.1.7.1 and v.2.1.1) and EllipSys3D computations based on the HG1 grid have the largest deviations from the measurements in this area. In terms of TKE, all simulations are about half the value obtained by the measurements.

One should also note the very good agreement between OpenFOAM v.2.1.1 and EllipSys3D (the dashed black and blue curves), based on an identical wall-function model (Richards and Hoxey, 1993) and calculated on identical computational meshes (HG1), especially in terms of speedup (Fig. 4a) but also in terms of TKE (Fig. 5a).

3.1.4 Simulation time

Default values (i.e., from `simpleFoam` tutorials), regarding basic OpenFOAM solver inputs, are used in all OpenFOAM-based calculations. Several attempts have been made to change/tweak some of (many) multigrid pressure solver – `GAMG` parameters, but they all resulted in prolonged computational times and in some cases led to a periodic rather than monotonic decay residual behavior. For extensive details about all input parameters considered, the interested reader is referred to Martinez (2011).

The computational process in EllipSys3D has been done both with the standard five-level grid sequencing[3] and without it. In the OpenFOAM case the direct grid sequencing pro-

[3]A built-in EllipSys3D function. It can easily be used to directly perform, for example, a grid dependency study.

Table 2. Askervein Hill: overview of different control parameters used to generate the SnappyHexMesh-created grids. The "blockMesh" column shows the number of grid points in the x, y, and z directions in the background mesh. The "Refinement surfaces" column shows the minimum level of surface refinement relative to the blockMesh-created background grid (first number in the parentheses) and the maximum surface refinement level (second number in the parentheses), which is used if a cell intersection angle (angle between two adjacent cells) is greater than `resolveFeatureAngle` (number shown in column three). The "Ref. box" 1–3 columns show the local grid refinement level relative to the background grid. The following three columns show grid size before the addLayersControls option is used, the number of added surface layers, and a total height of grid zone corresponding to added surface layers. The last two columns show the number of grid points in added surface layers (total number of grid points in the entire mesh shown in the parentheses) and the ratio between the first cell center height and the roughness length. The definition of all grids, including the HypGrid-based ones, together with the corresponding grid sizes is also included. Positions of refinement boxes, surface layers and background grid are indicated in Fig. 3. SHM0–4 are the SnappyHexMesh-based grids and HG0–2 are the HypGrid-based ones.

	blockMesh (background grid)	Refinement surfaces	Resolve feature angle (°)	Ref. box 1	Ref. box 2	Ref. box 3	Grid size before add. surf. layer (mill.)	No. surface layers	Total height of grid layer (m)	Grid size of add surf. layer (total) (mill.)	$\Delta z/z_0$
SHM0	60, 60, 40	level (2 2)	3	0	0	0	0.31	6	7.98	0.33 (0.64)	13.3
SHM1	115, 115, 80	level (2 3)	3	1	0	2	1.69	6	7.98	1.75 (3.44)	13.3
SHM2	115, 115, 80	level (2 3)	3	1	0	2	1.69	8	0.99	2.32 (4.01)	1.0
SHM3	115, 115, 80	level (2 3)	3	1	0	3	2.10	10	9.03	2.90 (5.00)	8.3
SHM4	115, 115, 40	level (3 3)	3	1	2	3	2.53	6	12.0	5.02 (7.55)	20.0
HG0	–	–	–	–	–	–	–	–	–	(1.31)	13.3
HG1	–	–	–	–	–	–	–	–	–	(5.24)	13.3
HG2	–	–	–	–	–	–	–	–	–	(5.24)	0.5

cedure does not exist. Here, two grids – the first consisting of $32 \times 32 \times 64$: 1.3 million cells – HG0 grid (Table 2) has been used in connection with OpenFOAM calculations on HG1 grid and the second one – SHM0 (Table 2) is used in connection with SnappyHexMesh-based OpenFOAM calculations. Upon reaching a suitable convergence level on HG0 and SH0 grids, OpenFOAM's `mapFields` tool has then been used to interpolate the results to the fine HG(1–2) and SHM(1–4) grids. The computational process on the fine grids is then continued until the convergence criterion is met. Also, the OpenFOAM computations are carried out without using the aforementioned procedure, i.e., solving on the fine grid using the standard initial values for all variables.

The obtained computational times are presented in Table 3.

In the comparison of EllipSys3D to OpenFOAM runs (Table 3), especially the fastest ones obtained on grids of similar size and location of the first near-ground computational cell (the EllipSys3D HG1 grid and OpenFOAM SHM3 grid), it can be observed that EllipSys3D is approximately a factor of 1.9–2.5 times faster in obtaining the numerical solution of the same level of accuracy.

3.2 Bolund

3.2.1 Simulation inputs

The inputs used to simulate Bolund Hill test case are based on quantities proposed by Bechmann et al. (2011). The values used are presented in Table 4.

Table 3. Askervein Hill: simulation times for EllipSys3D and OpenFOAM codes. For information about different grids utilized in the computations, see Table 2 and Fig. 3.

	Grid sequencing on	Grid sequencing off	Grid size (mill.)
EllipSys3D : HG2 $\Delta z = 0.50\, z_0$	509 s	826 s	5.24
EllipSys3D : HG1 $\Delta z = 13.3\, z_0$	454 s	754 s	5.24
OF v.1.7.1 : HG0 $\Delta z = 26.6\, z_0$	–	1323 s	1.31
OF v.1.7.1 : HG1 $\Delta z = 13.3\, z_0$	6655 s	10 259 s	5.24
OF v.1.7.1 : SHM0 $\Delta z = 13.3\, z_0$	–	68 s	0.64
OF v.1.7.1 : SHM1 $\Delta z = 13.3\, z_0$	693 s	1167 s	3.44
OF v.1.7.1 : SHM3 $\Delta z = 8.33\, z_0$	989 s	1909 s	5.00
OF v.1.7.1 : SHM4 $\Delta z = 20.0\, z_0$	1867 s	4311 s	7.55
OF v.2.1.1 : HG1 $\Delta z = 13.3\, z_0$	3042 s	14 421 s	5.24
OF v.2.1.1 : SHM2 $\Delta z = 1.00\, z_0$	527 s	1037 s	4.01
OF v.2.1.1 : SHM3 $\Delta z = 8.33\, z_0$	862 s	2013 s	5.00
OF v.2.1.1 : SHM4 $\Delta z = 20.0\, z_0$	1591 s	3447 s	7.55

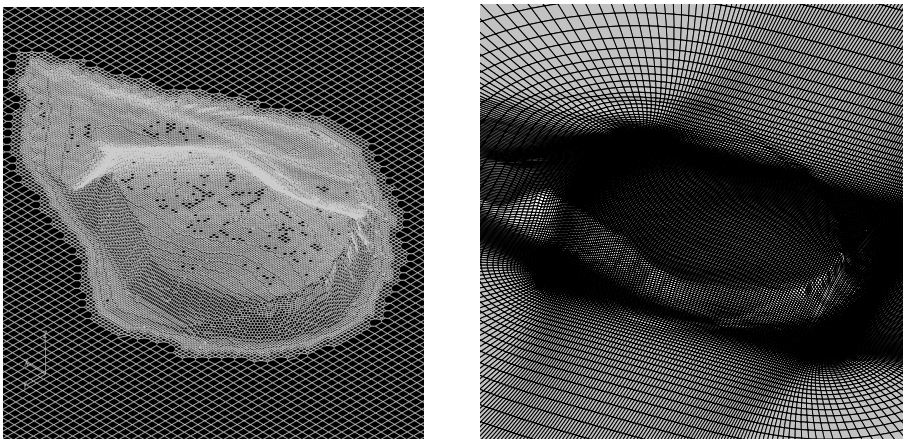
3.2.2 Description of the mesh

HypGrid

A Bolund map used in connection with the blind comparison test presented in Bechmann et al. (2011) is used as a basis for the generation of the surface mesh. The surfgrid tool could be readily used to generate an appropriate surface mesh for the EllipSys3D flow solver, but it turned out to be difficult to produce a surface mesh which could be used in both the EllipSys3D and the OpenFOAM solvers. A high-complexity and very abrupt change in the surface structure on the

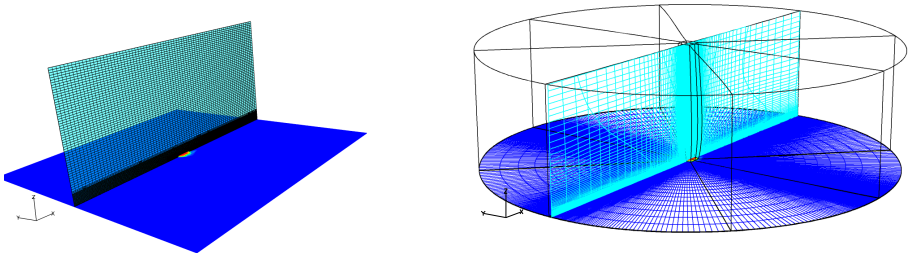
Table 4. Set of atmospheric parameters used for Bolund Hill test case.

u_* [m s^{-1}]	z_0 [m]	κ	C_μ	σ_k	σ_ε	$C_{\varepsilon 1}$	$C_{\varepsilon 2}$
0.4	3×10^{-4} for $z < 0.8$ m 1.5×10^{-2} for $z \geq 0.8$ m	0.4	0.03	1.0	1.30	1.21	1.92



(a) (b)

Figure 6. Bolund Hill: surface grid generated by **(a)** SnappyHexMesh and **(b)** HypGrid and smoothed by Gridgen.



(a) (b)

Figure 7. Bolund Hill: volume grid generated with **(a)** SnappyHexMesh and **(b)** HypGrid.

Bolund Hill front side caused severe difficulties with regard to obtaining a suitable OpenFOAM 3D meshes, without problems in cell face orientation. The grid validity check conducted using the `checkMesh` OpenFOAM tool showed that bad cells could always be located in the mentioned area. As the surfgrid tool does not have an option to visualize the grid during the creation process or provide a possibility to smooth the created grids, Pointwise's Gridgen mesh generation software has thus been used for the purpose of smoothing the surfgrid-generated mesh[4]. A close-up view of the generated surface grid is presented in Fig. 6b. The 4 km in diameter surface mesh is centered on and refined around the Bolund Island position. The HypGrid tool is used to hyperbolically march the grid in the third dimension (up to a height of 1 km). The final volume grid (Fig. 7b) is comprised of 24 blocks of $64 \times 64 \times 64$ cells, approximately 6.3 million grid points. Basically, three meshes are created in this way, one with the first cell center height of $\Delta z = 0.1875$ m, i.e., $k_s(z_{0(0.0003)}) \approx 0.006\,\text{m} < \Delta z < k_s(z_{0(0.015)}) = 0.294$ (OpenFOAM v.1.7.1 – the HG1 grid), a second one with the first cell center height of $\Delta z = 0.0125$ m, i.e., $z_{0(0.0003)} < \Delta z < z_{0(0.015)}$ (OpenFOAM v.2.1.1 – the HG2 grid), and a third one with the first cell center height of $\Delta z = 0.0005$ m, i.e., $z_{0(0.0003)} < \Delta z < z_{0(0.015)}$ (EllipSys3D – the HG3 grid).

As can be seen, some compromises on the position of the first grid point in the surface normal direction (especially in the OpenFOAM cases) had to be made due to the change in the surface roughness length throughout the computational

[4]It turned out that a very limited number of cells very close to the Bolund Hill front edge needed to be rearranged. This was done by still keeping the fixed surface projection, ensuring that the created grid had the same qualitative Bolund Hill surface representation.

Table 5. Bolund Hill: overview of different control parameters used to generate the SnappyHexMesh-created grids. For the definition of different parameters in the table, see Table 2. Positions of refinement boxes, surface layers and background grid are indicated in Fig. 8. SHM0–3 are the SnappyHexMesh-based grids and HG0–3 are the HypGrid-based ones. $z_{0(L)} = z_{0-\mathrm{Land}} = 0.015$ m.

	blockMesh (background grid)	Refinement surfaces	Resolve feature angle (°)	Ref. box 1	Ref. box 2	Ref. box 3	Grid size before add. surf. layer (mill.)	No. surface layers	Total height of grid layer (m)	Grid size add surf. layer (total) (mill.)	Δz z_0
SHM0	45, 45, 50	level (3 5)	2	0	0	0	0.43	3	1.60	0.41 (0.84)	13.3
SHM1	75, 75, 60	level (2 3)	2	3	2	3	3.47	6	2.61	2.13 (5.60)	10.0
SHM2	75, 75, 60	level (3 5)	2	0	2	3	3.53	3	1.60	1.15 (4.68)	13.3
SHM3	75, 75, 60	level (3 5)	2	0	2	3	3.53	11	1.74	4.10 (7.63)	1.0
HG0	–	–	–	–	–	–	–	–	–	(0.78)	25.0
HG1	–	–	–	–	–	–	–	–	–	(6.29)	12.5
HG2	–	–	–	–	–	–	–	–	–	(6.29)	0.83
HG3	–	–	–	–	–	–	–	–	–	(6.29)	0.03

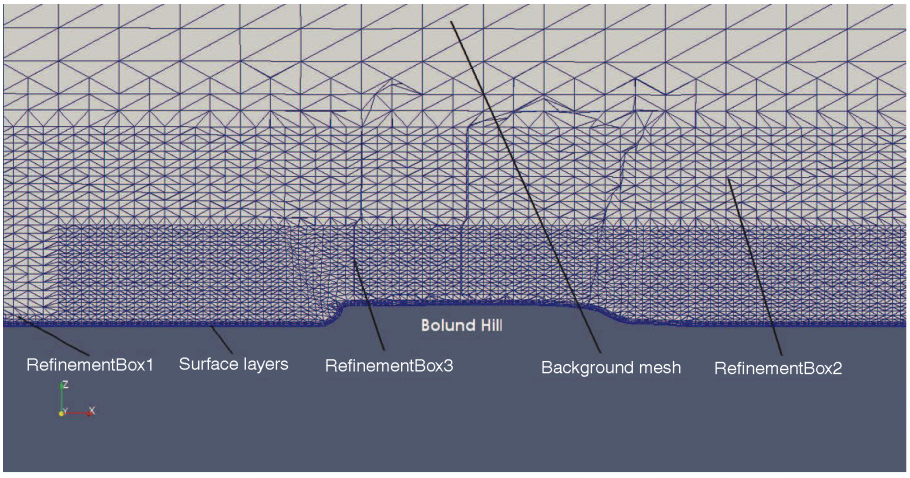


Figure 8. Bolund Hill: cross section of the grid created by SnappyHexMesh. Three refinement boxes together with variations in the number of surface layers indicated in the figure are used as a basis for generation of the grids described in Table 5.

domain. Even though no formal model-based restriction exists regarding the position of the first cell center height in the Richards and Hoxey (1993) implementation of the wall function in OpenFOAM v.2.1.1, the previously discussed aspect ratio issue dictated that $\Delta z = 0.0125$ m. With this Δz value the max aspect ratio is approximately 7000, and the only error/warning reported by the `checkMesh` tool was regarding the cell aspect ratio. Diminishing the Δz value further introduces several other mesh related errors reported by the `checkMesh` tool and any attempt to obtain the numerical solution resulted in almost immediate divergence.

SnappyHexMesh

A similar procedure to that in the Askervein Hill test case, regarding the re-use of the surfgrid-created ground surface mesh utilized in the HypGrid meshing tool, is conducted in the Bolund Hill case. Upon the surface grid conversion to STL file format, a rectangular domain of $2.3 \times 2.3 \times 1.0$ km is discretized using the blockMesh utility (see Table 5) creating a background mesh with resolution of $30.7 \times 30.7 \times 16.7$ m in x, y, z directions, respectively. The final surface grid created using the SnappyHexMesh tool is presented in Fig. 6a. Analogously to the Askervein Hill case, the cross section of the SnappyHexMesh-created (SHM3) grid is shown in Fig. 7a and indicators locating positions of refinement boxes and surface layers used to generate meshes SHM(0–3) are presented in Fig. 8. Similarly, both adjustment of the refinement levels in refinement boxes 1–3 and the number of inserted surface layers are the controlling parameters used for the SnappyHexMesh grid creation (Table 5). The `refinementSurface` and `resolveFeatureAngle` parameters are used to control the surface refinement level relative to the background grid. They seem to play a much more important role in the Bolund Hill case than previously, apparently due to the sudden and abrupt change in the surface topology characterizing the Bolund Hill case. It should be noted that increasing the surface refinement level to more

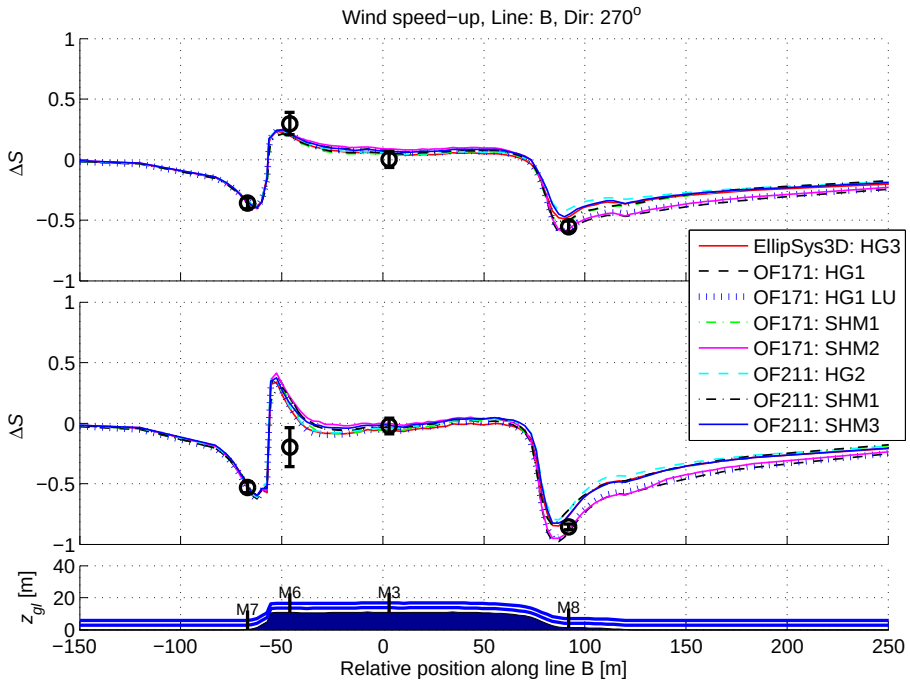


Figure 9. Bolund Hill: speedup along line B. Uppermost panel: 5 m a.g.l. (above ground level); middle panel: 2 m a.g.l. Measurements are denoted as circles with corresponding uncertainties denoted by an error bar.

than level 5 (Table 5) led directly to an inability of SnappyHexMesh to create valid surface layers. The coarse SHM0 and HG0 (Table 5) grids are again only intended for use in connection with grid sequencing procedures.

3.2.3 Simulation results

The results of the Bolund test case are presented in Figs. 9 and 10. The computations are conducted for the incoming wind flow direction of 270°, and results are compared with measurements along the line B (for details see Bechmann et al., 2011).

The obtained results are compared to the results submitted by other participants of the Bolund blind comparison (not shown here) and found to be in close agreement with the submitted numerical results attained utilizing two-equation turbulence models.

As a roughness change characterizes the Bolund Hill test case, some considerations regarding the position of the first grid point in the surface normal direction do exist. The wall-function closure used in EllipSys3D flow solver does not restrict the position of the first grid point in the surface normal direction, so the position chosen here corresponds to $3\times$ roughness length of the water ($z_0 = 0.0003$ m) and $1/15\times$ roughness length of the land ($z_0 = 0.015$ m). In the OpenFOAM v.1.7.1 case, based on investigations of Martinez (2011), the largest roughness length for land had to be used as a basis for the grid creation process in order to avoid problems with the limitations of the Nikuradse's sand roughness length closure ($\Delta z/z_{0-\mathrm{Land}} = 12.5$). This, however, places the first grid point at the inlet and whole region upstream of the Bolund Island position at a very large relative distance from the terrain ($\Delta z/z_{0-\mathrm{Water}} = 625$). This issue could potentially negatively influence the results upstream of the Bolund Island position.

For this reason two different 2-D flat terrain computations involving the grids where the position of the first grid point in the surface normal direction was varied from $\Delta z/z_{0-\mathrm{Water}} = 12.5$ to $\Delta z/z_{0-\mathrm{Water}} = 625$ were conducted. The obtained results (not shown here), however, appeared to be in a close agreement with each other. Also, comparison of OpenFOAM (both v.1.7.1 and v.2.1.1) and EllipSys3D results at the position of mast 7 (M7) – positioned just upstream of Bolund Island – shown in Fig. 11 indicates that the mentioned issue does not seem to have any negative influence on most of the presented OpenFOAM (both v.1.7.1 and v.2.1.1) results. As seen from Fig. 11, only OpenFOAM (both v.1.7.1 and v.2.1.1) results based on the SHM1 grid deviate in terms of TKE from the rest of the computations. This issue will be discussed further in the next section.

From Figs. 9 and 10 a general good agreement can be observed between results of OpenFOAM v.2.1.1, v.1.7.1 and EllipSys3D together with their relatively good correspondence with the measurements. Largest differences between results can be observed in TKE plots at 2 m a.g.l. (above ground level). This subject will also be discussed further in Sect. 4.2. Furthermore, in particular, results using Open-

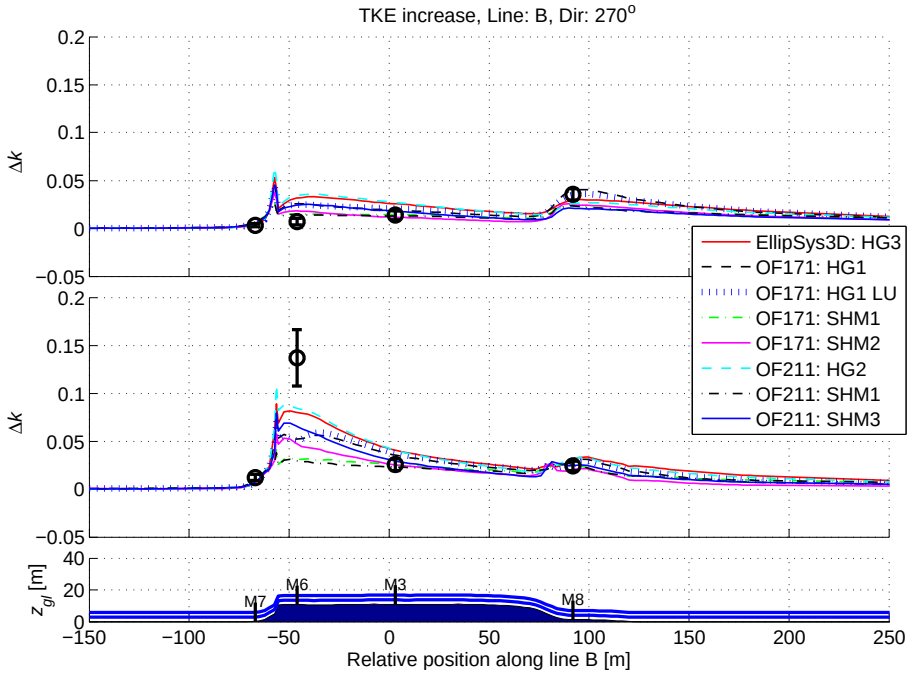


Figure 10. Bolund Hill: turbulent kinetic energy along line B. Uppermost panel: 5 m a.g.l.; middle panel: 2 m a.g.l.

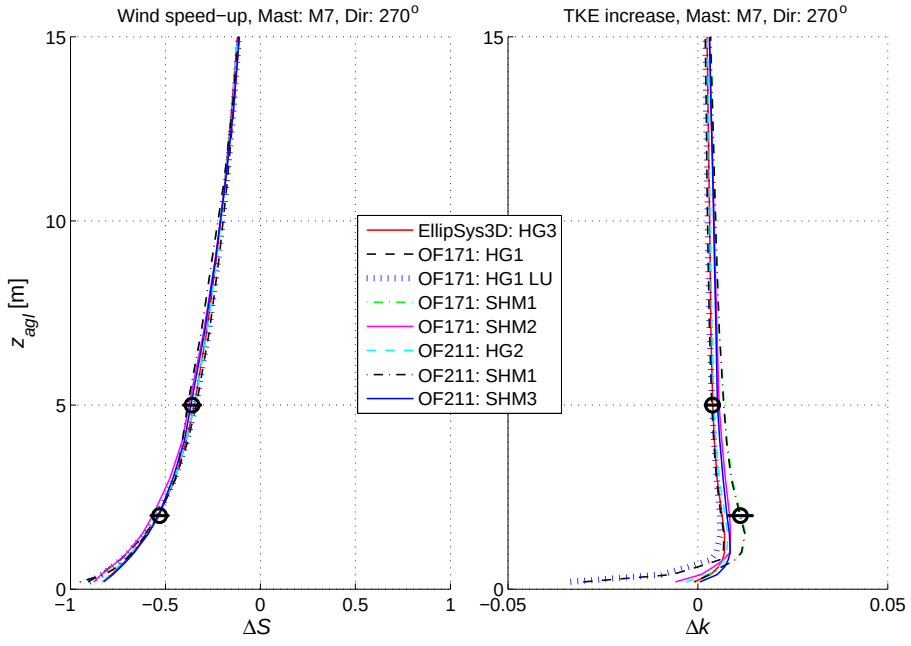


Figure 11. Bolund Hill: mast position 7, speedup and turbulent kinetic energy.

FOAM v.2.1.1 (HG2 grid) and EllipSys3D, based on an identical wall-function model, have very good agreement. It is also seen that all calculations have similar difficulties in agreements with the measurements in the area immediately after the sharp Bolund Hill front edge, as previously reported in Bechmann et al. (2011).

3.2.4 Simulation time

The solver inputs, apart from the Bolund Hill-specific ones presented in Table 4, regarding both OpenFOAM and EllipSys3D, are kept identical to the inputs previously used in the Askervein test case. It should be noted that stable convergence could not be obtained on SnappyHexMesh-based

Table 6. Bolund Hill: simulation times for EllipSys3D and OpenFOAM codes. For information about different grids utilized in the computations, see Table 5 and Fig. 8. The OpenFOAM v.2.2.1 runs on the HG2 grid are conducted using the `PCG` pressure solver (results in parentheses) instead of `GAMG`.

	Grid sequencing on	Grid sequencing off	QUICK(V)	Linear upwindV	Grid size (mill.)
EllipSys3D : HG3 $\Delta z = 0.03\, z_{0(L)}$	286 s	3656 s	X		6.29
OF v.1.7.1 : HG0 $\Delta z = 25.0\, z_{0(L)}$	–	734 s	X		0.78
OF v.1.7.1 : HG1 $\Delta z = 12.5\, z_{0(L)}$	4073 s	14 396 s	X		6.29
OF v.1.7.1 : HG1 $\Delta z = 12.5\, z_{0(L)}$	3307 s	9838 s		X	6.29
OF v.1.7.1 : SHM0 $\Delta z = 13.3\, z_{0(L)}$	–	119 s		X	0.84
OF v.1.7.1 : SHM1 $\Delta z = 10.0\, z_{0(L)}$	1844 s	2108 s		X	5.60
OF v.1.7.1 : SHM2 $\Delta z = 13.3\, z_{0(L)}$	1507 s	1700 s		X	4.68
OF v.2.1.1 : HG2 $\Delta z = 0.83\, z_{0(L)}$	10 153 (5780) s	40 630 (33 567) s	X		6.29
OF v.2.1.1 : HG2 $\Delta z = 0.83\, z_{0(L)}$	10 259 (5650) s	35 857 (27 683) s		X	6.29
OF v.2.1.1 : SHM1 $\Delta z = 10.0\, z_{0(L)}$	1171 s	1789 s		X	5.60
OF v.2.1.1 : SHM3 $\Delta z = 1.00\, z_{0(L)}$	1656 s	2820 s		*X*	7.63

grids using the `QUICKV` discretization scheme. For that reason, a formally second-order `linearUpwindV`[5] scheme was used in those cases. Computations on the HypGrid-based HG1 grid in OpenFOAM were redone using the `linearUpwindV` scheme in order to be able to access the speed of convergence and quality of the obtained results in a proper manner. The OpenFOAM v.2.2.1 runs on the HG2 grid are conducted using the `PCG` pressure solver also (results in parentheses in Table 6) instead of `GAMG`. The grid sequencing procedure is analogous to the one presented in Sect. 3.1.4, only here a grid corresponding to grid level 2 in EllipSys3D (every second point in all three directions is removed – the HG0 grid) has been separately created and utilized for the mesh sequencing procedure in the OpenFOAM HG(1–2) runs. The coarse SHM0 grid is created for the same purpose according to the specifications in Table 5 with regard to SnappyHexMesh-based OpenFOAM simulations.

The obtained computational times are shown in Table 6.

Comparing EllipSys3D to fastest OpenFOAM runs on grids of similar size (Table 6), it can be observed that EllipSys3D, in obtaining the numerical solution of the same level of accuracy, is approximately a factor of 4–6 times faster when the grid sequencing procedure is turned on and a factor of 1.3–1.7 times slower when it is turned off.

4 Discussion

Firstly, it should be pointed out that several simulations using the Nikuradse's equivalent sand roughness length wall model have been rerun in the more recent OpenFOAM v.2.1.1. Also, OpenFOAM v.2.1.1 has been compiled using the optimized Intel icc compiler for the Intel Xeon® (enclosed in the DTU Wind Energy cluster facility) CPU platform, and selected simulations were rerun in this environment as well. In both cases no noticeable differences in the computational times, compared to the ones presented here, were observed.

4.1 Mesh generation

4.1.1 SnappyHexMesh

OpenFOAM's own SnappyHexMesh meshing tool seems to have reasonable grid generation applicability and flexibility for the investigated neutral atmospheric boundary layer (ABL) flow over complex terrain, although a number of problems were encountered during the meshing process.

In the Askervein Hill case it was possible to create several grids with good general and boundary layer resolution capabilities using it. Dedicated grids for both Nikuradse's equivalent sand roughness length wall model and the atmospheric roughness length wall-function approach could be made directly. Generating the surface layer, crucial for appropriate boundary layer description in the ABL simulations, was a quite difficult task. Only usable results were obtained by splitting the *add surface layer* grid generation process form the rest of the SnappyHexMesh mesh generation procedure and disabling all mesh quality checks during this phase. Otherwise, very strange results with several regions of missing surface layer parts were obtained. The grids created during the present work basically reflect more of a limit in what SnappyHexMesh is capable of handling regarding the addition of surface layers in the grid, rather than a carefully considered user-based specification of sizes and extent of different surface layer parameters.

In the Bolund Hill case, some of the mentioned surface layer generation problems were even more emphasized. The abrupt change in surface structure at the Bolund Hill front side created severe problems in the gener-

[5] Specifically, div(phi,U) Gauss linearUpwindV cellMDLimited Gauss linear 1.

ation of the surface layers, so extending, for example, the `refinementSurfaces` level parameter to more than 5, in order to better approximate the Bolund ground surface, was not possible in the current case. Basically, there is very little freedom in specification of many surface-layer-related parameters. Generally, it seems very difficult as a user to be in control of the surface layer creation process using the SnappyHexMesh tool, making it quite difficult for it to be used consistently in relation to grid generation processes relevant for ABL flows.

4.1.2 HypGrid

In contrast, HypGrid, a meshing tool developed deliberately to create low-skewness hyperbolic 3-D meshes based on complex surface topologies, appears to be able to cope with both investigated geometries without significant problems. EllipSys3D could easily handle all grids created by HypGrid, but some adjustments, described in Sect. 3.2.2, were necessary in order to make suitable grids for OpenFOAM runs.

4.2 Accuracy

4.2.1 Askervein Hill case

The results obtained with OpenFOAM for the Askervein Hill case show very good general agreement with EllipSys3D and cup anemometer measurements in terms of speedup curves presented in Fig. 4b. In particular, results using OpenFOAM v.2.1.1 SHM(2–4) grids, OpenFOAM v.1.7.1 SHM(3–4) grids and the EllipSys3D HG2 grid (Fig. 4a) seem to have a very good correspondence, both prior to and after the hilltop. The OpenFOAM (v.1.7.1 and v.2.1.1) calculation based on the HypGrid-created mesh (HG1) together with EllipSys3D calculation on the same grid appear to deviate significantly from the abovementioned cases and measurements on the lee side of the hill. Results based on the OpenFOAM v.1.7.1 SHM1 grid seem to be placed in between the two above-described sets of results. An important thing to note here is a very good correspondence between results of OpenFOAM v.2.1.1 and EllipSys3D, which are run on identical computational grids (HG1) using the same wall-function modeling approach[6].

Regarding the TKE plots presented in Fig. 5, a good correspondence between all computations (and partially measurements) can be observed on the front side of the hill. Only the EllipSys3D HG2 grid-based calculation deviates from the rest of the computations in the immediate vicinity of the hill top. A similar behavior regarding the TKE in this particular region has been observed in the OpenFOAM v.2.1.1 calculations on previously mentioned grids with cell center heights of the order of the roughness length – $\Delta z = 0.83\, z_0$ and $\Delta z = 1.5\, z_0$ (not shown here).

[6]Practically the only difference can be seen for horizontal distance from HT > 500 m.

The general deviations between numerical results on the lee side of the hill are much larger, but the differences between all computations and measurements here appear much more significant (numerical findings underpredict measurements by more than 50 %) and dominant than the differences between the computations. In the obtained steady state solutions occurrence of the flow separation has not been detected. As intermittent flow separation seemed to occur during the observational period, and as separation increases the general turbulence levels, this can be a possible reason why currently used RANS turbulence model cannot predict levels of turbulent kinetic energy on the lee side of the hill accurately (Undheim et al., 2006). The RANS models in general are reported to have a substantial problem in predicting the measured turbulent kinetic energy levels correctly in the mentioned zone (Sørensen, 1995; Kim and Patel, 2000; Eidsvik, 2005). However, Castro et al. (2003) – using high-order accurate schemes and unsteady RANS (URANS) formulation – seem to better capture the measured turbulence properties whereas recent large eddy simulation (LES) studies (Chow and Street, 2009) and hybrid RANS–LES studies (Bechmann and Sorensen, 2011) do show some general, significant and promising improvements in this regard.

Considering OpenFOAM results from both Figs. 4 and 5, the OpenFOAM computations, based on SnappyHexMesh-created grids SHM(2–4), seem to have the best agreement with measurements and EllipSys3D.

4.2.2 Bolund Hill case

The first observation regarding the Bolund Hill results presented in Figs. 9 and 10 is very close agreement between results of OpenFOAM v.2.1.1 (HG2 grid, dashed cyan line) and EllipSys3D (HG3 grid, red line), which both use the Richards and Hoxey (1993)-based wall-function approach. This close agreement indicates that both flow solvers are indeed generally quite capable of producing reliable CFD results.

The differences in wall-modeling approach appear to play a dominant role in flow predictions in the recirculation zone on the lee side of the hill. Here, the OpenFOAM v.1.7.1-based calculations (HG1 (HG1 LU) and SHM2) seem to collapse to a single curve, while OpenFOAM v.2.1.1-based calculations (HG2, SHM(1,3) grids and EllipSys3D HG3 grid) seem to collapse to another curve at both 2 and 5 m a.g.l. Only the OpenFOAM v.1.7.1 SHM1 grid-based calculation follows the EllipSys3D and OpenFOAM v.2.1.1 results rather than the rest of the OpenFOAM v.1.7.1 computations.

Regarding the TKE plots in Fig. 10, it is seen that all simulations at the position of mast 6 (M6) seem to have difficulty in producing the correct level of the turbulent kinetic energy, especially for the measured height of 2 m a.g.l. Considerable variation between the results can be seen in this particular region, where almost all OpenFOAM v.2.1.1 and EllipSys3D results predict higher peak values close to the M6 position than the OpenFOAM v.1.7.1-based calculations. The SHM1

grid-based calculation (now in OpenFOAM v.2.1.1) deviates again from general tendencies and is seen to produce almost identical results to the OpenFOAM v.1.7.1 calculation on the same grid.

The behavior of computed results on SHM1 grid therefore appears more to reflect the ability of the grid to "snap" the correct Bolund ground surface (`refinementSurface` level 3 is used in SHM1 grid vs. `refinementSurface` level 5 in SHM(2,3) grids) rather than the wall-modeling approach used in the computations. This inability of SHM1 grid to properly represent the Bolund ground surface is also visible in results at mast position M7 (Fig. 11), especially on the TKE plot subfigure. It should be noted that increasing `refinementSurface` level to a value higher than 5 resulted in an inability of SnappyHexMesh to create any valid surface layers, due to grid distortion of closely spaced almost perpendicular cell layers, underlining a general difficulty/inability of full user control over surface grid layer creation in SnappyHexMesh OpenFOAM utility. Therefore, the OpenFOAM v.2.1.1 SHM3-based results represent the highest peak value close to M6 position which can be produced with the current SnappyHexMesh-based setup. The OpenFOAM v.2.1.1-based HG2 computation is seen to easily extend the SHM3 mesh predicted peek value in this particular region.

Comparison of OpenFOAM v.1.7.1-based (HG1 and HG1 LU) results which use identical settings and grid, with the only difference being in the discretization scheme used (`QUICKV` vs. `linearUpwindV`), shows practically no difference in the obtained results, indicating that utilization of the `linearUpwindV` scheme for SnappyHexMesh-based computations does not seem to impair the quality of the computed results.

4.3 Speed

4.3.1 Askervein Hill case

Computational times from Table 3 show that a converged solution on a SnappyHexMesh-based grid SHM3 is between 3.5 and 6.5 (grid sequencing on) and 5 and 7 (grid sequencing off) times faster to obtained than results on a HypGrid-based grid (HG1) with similar number of grid points using both the OpenFOAM v.1.7.1 and v.2.1.1 solvers. A closer look at the residual curves showed that a single iteration takes roughly the same amount of time in both SnappyHexMesh- and HypGrid-based cases, but the number of iterations needed to reach the convergence level of 2×10^{-4} is, as indicated in Table 3, much higher in the HypGrid-based mesh case. This indicates that a structured meshing tool like HypGrid might not be the most optimal choice for grid generation purposes in OpenFOAM.

Focusing now on SnappyHexMesh-based results, it can be seen that a speed increase close to a factor of 2 is gained by using the grid sequencing procedure. The same is almost true in the HypGrid-based cases: there is slightly lower speed gain in the OpenFOAM v.1.7.1 case and slightly higher gain in the OpenFOAM v.2.1.1 case. In the SnappyHexMesh SHM3 case the speed differences between OpenFOAM v.2.1.1 and OpenFOAM v.1.7.1 computations are almost negligible, indicating that differences in the wall-modeling approach used in the two calculations do not seem to affect the convergence speed process significantly. Also, considering the increase in the number of grid points – following cases SHM(1,3,4) OpenFOAM v.1.7.1 and SHM(2,3,4) OpenFOAM v.2.1.1 – a proportional reflection in the computational times from Table 3 can be observed, although the proportionality factor seems to vary in a nonlinear manner.

Compared to the Bolund Hill test case, the grid sequencing procedure in the EllipSys3D computations does not appear to influence the computational time considerably. The reasons for this probably lie in the general flow complexity (or rather lack of the same) of the Askervein Hill case, as it apparently prevents the solver from fully utilizing the advantage of a solution on the coarser grid level, compared to using the standard start guess in the solution procedure on the fine grid. Placing the first near-wall cell very close to the ground (the HG2 grid) is seen to have a slight influence on the computational times also (relative to the HG1 grid).

In comparison of the EllipSys3D and OpenFOAM runs, best done on grids of similar size and position of the first cell center above ground level (i.e., the HG1 and SHM3 grids), it is seen that EllipSys3D is approximately 1.9 (grid sequencing on) and 2.5 (grid sequencing off) times faster in obtaining the converged solution.

4.3.2 Bolund Hill case

In the Bolund Hill case[7], a significant difference in computational times between SnappyHexMesh- and HypGrid-based OpenFOAM cases is again observed (Table 6). When comparing results on grids of similar sizes and near-ground surface resolution capabilities, conducted using the same `linearUpwindV` discretization scheme, a speed difference of a factor of 1.8 (grid sequencing on), 4.7 (grid sequencing off) for OpenFOAM 1.7.1 (cases HG1 and SHM1) and a factor of 3.4 (grid sequencing on) and 9.8 (grid sequencing off) for OpenFOAM 2.1.1 (cases HG2 – fastest, `PCG`-based solution and SHM3) can be observed. This underlines once again a general OpenFOAM issue with grids comprised of high-aspect-ratio cells. It is also seen from Table 6 that the multigrid `GAMG` pressure solver is not functioning optimally and is significantly outperformed by the more conventional `PCG` solver in HypGrid-based OpenFOAM v.2.1.1 cases.

Furthermore, Table 6 indicates that converged solution using `QUICKV` scheme is slower to obtain than the corresponding solution using `linearUpwindV` discretization scheme,

[7] A residual convergence level of 10^{-4} is reached in all computations.

especially when grid sequencing procedure is turned off (of the order of 20–30 %).

Grid sequencing procedures seem to have a smaller positive influence on the SnappyHexMesh-based calculations in the Bolund Hill case (speedup factor of < 2) compared to the Askervein Hill case, while the opposite is true for the HypGrid-based calculations (speedup factor of approximately 3–5).

The EllipSys3D code seems to benefit considerably from its automatic grid sequencing procedure, as it speeds up the computations by more than a factor of 10.

From comparison of the EllipSys3D run with best directly comparable OpenFOAM run (v.2.1.1 SHM3 smallest Δz, identical wall-modeling approach) it can be seen that EllipSys3D is approximately a factor of 6 times faster in obtaining the converged solution (grid sequencing on), but despite the fact that OpenFOAM SHM3 computation includes almost 20 % more grid points, it is faster (approximately 30 %) to obtain the solution on it than using the structured EllipSys3D solver with grid sequencing turned off.

5 Conclusion

In this work, the unstructured OpenFOAM flow solver is compared to the structured EllipSys3D flow solver on two test cases calculating neutral atmospheric boundary layer flow over complex terrain.

Two meshing tools are considered: the structured hyperbolic 3-D mesh generator HypGrid and OpenFOAM's own hexahedral mesh generator SnappyHexMesh. OpenFOAM was found to be able to successfully perform calculations on the HypGrid-created meshes in both considered test cases. SnappyHexMesh was also able to produce reasonable grids in both the Askervein Hill and Bolund Hill test cases. A very important parameter for computational grids in ABL flows – height of the first near-ground cell – proved to be very difficult to directly control using the SnappyHexMesh tool reflected in its (in)ability to create surface layers. This issue makes it quite difficult to use SnappyHexMesh consistently in relation to grid generation processes relevant for ABL flows, so a tool with more direct user control over this crucial part of the grid generation process can be recommended.

In terms of accuracy, both flow solvers are found to perform equally well on the two test cases, regarding both the mean flow velocity and turbulence quantities. In particular, OpenFOAM v.2.1.1 and EllipSys3D calculations, performed on identical (Askervein Hill case) and very similar (Bolund Hill case) computational grids, using the same approach to wall-function modeling of ABL flows (Richards and Hoxey, 1993), were found to have a great mutual correspondence, underlining the fact that both flow solvers are quite capable of producing reliable numerical results.

However, a large discrepancy in the speed performance is found. A very large difference in calculation times is obtained between HypGrid- and SnappyHexMesh-based OpenFOAM calculations, indicating that a structured meshing tool, typically creating grids with high-aspect-ratio cells in ABL flows, causes the OpenFOAM solver to perform inefficiently. Results of the present work show that this performance issue can be partially addressed by introducing a grid sequencing procedure in the OpenFOAM runs. Generally, the grid sequencing procedure had a very positive effect on almost all OpenFOAM computations and can be highly recommended.

In the comparison of EllipSys3D, which utilizes the grid sequencing procedure by default, and OpenFOAM SnappyHexMesh-based calculation times, the structured EllipSys3D solver is found to perform approximately 2–6 times faster on grids with similar properties.

Using a combination of hexahedral and polyhedral cells the SnappyHexMesh tool can produce suitable grids in many relevant ABL flow cases and bring the computational times close to the level of the structured EllipSys3D code, but the inability to fully control ground surface approximation and associated surface layer creation, as in the Bolund Hill case, can potentially impair the quality of the obtained SnappyHexMesh-based results. On the other hand, use of the structured HypGrid solver, where terrain description and surface layer creation can be fully controlled, proved to be accompanied with a high speed performance penalty in the OpenFOAM runs. Thus, an unstructured mesh generation tool where both surface approximation and accompanied surface layer creation is better controlled might be an optimal meshing tool for ABL flow calculations using the OpenFOAM solver.

Acknowledgements. This work is supported by the Center for Computational Wind Turbine Aerodynamics and Atmospheric Turbulence, funded by the Danish Council for Strategic Research, grant number 09-067216. Partial support from the ERANET Plus project called the New European Wind Atlas is appreciated.

Edited by: S. Aubrun

References

Bechmann, A. and Sørensen, N.: Hybrid RANS/LES Method for Wind Flow over Complex Terrain, Wind Energy, 13, 36–50, doi:10.1002/we.246, 2010.

Bechmann, A. and Sorensen, N. N.: Hybrid RANS/LES applied to complex terrain, Wind Energy, 14, 225–237, doi:10.1002/we.414, 2011.

Bechmann, A., Sørensen, N. N., Berg, J., Mann, J., and Réthoré, P.-E.: The Bolund Experiment, Part II: Blind Comparison of Microscale Flow Models, Bound.-Lay. Meteorol., 141, 245–271, doi:10.1007/s10546-011-9637-x, 2011.

Berg, J., Mann, J., Bechmann, A., Courtney, M. S., and Jørgensen, H. E.: The Bolund Experiment, Part I: Flow Over a Steep,

Three-Dimensional Hill, Bound.-Lay. Meteorol., 141, 219–243, doi:10.1007/s10546-011-9636-y, 2011.

Blocken, B., Stathopoulos, T., and Carmeliet, J.: CFD simulation of the atmospheric boundary layer: wall function problems, Atmos. Environ., 41, 238–252, doi:10.1016/j.atmosenv.2006.08.019, 2007.

Boussinesq, J.: Théorie de l'écoulement tourbillant, Mém. prés. par div. savants á l'Acad. Sci. Paris, 23, 46–50, 1877.

Castro, F. A., Palma, J. M. L. M., and Silva Lopes, A.: Simulation of the Askervein Flow. Part 1: Reynolds Averaged Navier–Stokes Equations (k–ϵ Turbulence Model), Bound.-Lay. Meteorol., 107, 501–530, doi:10.1023/A:1022818327584, 2003.

Chow, F. K. and Street, R. L.: Evaluation of Turbulence Closure Models for Large-Eddy Simulation over Complex Terrain: Flow over Askervein Hill, J. Appl. Meteorol. Clim., 48, 1050–1065, doi:10.1175/2008JAMC1862.1, 2009.

Eidsvik, K. J.: A system for wind power estimation in mountainous terrain. Prediction of Askervein hill data, Wind Energy, 8, 237–249, doi:10.1002/we.145, 2005.

Hackman, L.: A Numerical Study of the Turbulent Flow Over a Backward Facing Step Using a Two Equation Turbulence Model, PhD thesis, Univ Waterloo, Ontario, 1982.

Kim, H. and Patel, V.: Test of turbulence models for wind flow over terrain with separation and recirculation, Bound.-Lay. Meteorol., 94, 5–21, doi:10.1023/A:1002450414410, 2000.

Launder, B. and Spalding, D.: The Numerical computation of turbulent flows, Comput. Meth. Appl. Mech. Eng., 3, 269–289, 1974.

Leonard, B. P.: A stable and accurate convective modelling procedure based on quadratic upstream interpolation, Comput. Meth. Appl. Mech. Eng.,, 19, 59–98, 1979.

Martinez, B.: Wind resource in complex terrain with OpenFOAM, MSc Thesis, June, Risø DTU, Roskilde, Denmark, 2011.

Michelsen, J. A.: Basis3D-a platform for development of multiblock PDE solvers, Report AFM, http://www.citeulike.org/user/pire_1024/article/5890200 (last access: May 2016), 1992.

Nikuradse, J.: Laws of Flow in Rough Pipes, Technical Memorandum November, NACA, http://ntrs.nasa.gov/search.jsp?R=19930093938 (last access: May 2016), 1950.

OpenFOAM: http://openfoam.org/, last access: May 2016.

Patankar, S. V. and Spalding, D. B.: A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-Dimensional Parabolic Flows, Int. J. Heat Mass Trans., 15, 1787–1972, 1972.

Richards, P. and Hoxey, R.: Appropriate boundary conditions for computational wind engineering models using the k–E turbulence model, J. Wind Eng. Indust. Aerodyn., 47, 145–153, 1993.

Sørensen, N. N.: General purpose flow solver applied to flow over hills, Tech. rep., Risø, http://www.citeulike.org/user/pire_1024/article/5890204 (last access: May 2016), 1995.

Sørensen, N. N.: HypGrid2D a 2-D mesh generator, Tech. rep., Risø DTU, Roskilde, Denmark, 130.226.56.153/rispubl/VEA/veapdf/ris-r-1035.pdf (last access: May 2016), 1998.

Sørensen, N. N., Bechmann, A., Johansen, J., Myllerup, L., Botha, P., Vinther, S., and Nielsen, B. S.: Identification of severe wind conditions using a Reynolds Averaged Navier-Stokes solver, J. Phys., 75, 012053, doi:10.1088/1742-6596/75/1/012053, 2007a.

Sørensen, N. N., Bechmann, A., Myllerup, L., and Botha, P.: Identification of severe wind conditions using a Reynolds Averaged Navier-Stokes solver, Proceedings of the EWEA/EAWE special topic conference: The Science of making Tourque from Wind, The Technical University of Denmark, http://orbit.dtu.dk/en/publications/identification (last access: May 2016), 2007b.

Sumer, B. M.: Lecture Notes on Turbulence, DTU (Technical University od Denmark), http://orbit.dtu.dk/en/publications/lecture-notes-on-turbulence (last access: May 2016), 2007.

Taylor, P. and Teunissen, H.: The Askervein Hill project: overview and background data, Bound.-Lay. Meteorol., 39, 15–39, 1987.

Undheim, O., Andersson, H. I., and Berge, E.: Non-linear, microscale modelling of the flow over Askervein hill, Bound.-Lay. Meteorol., 120, 477–495, doi:10.1007/s10546-006-9065-5, 2006.