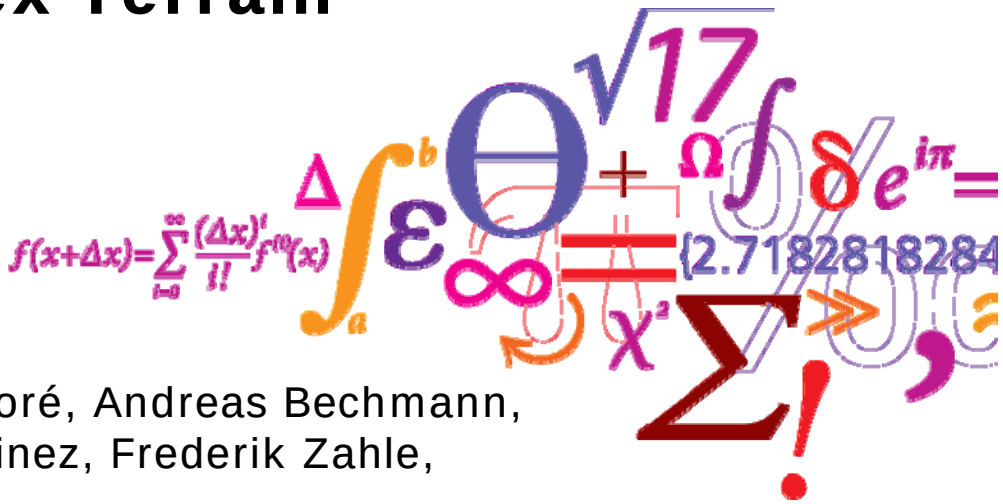
Comparison of OpenFOAM and EllipSys3D for Neutral Atmospheric Flow over Complex Terrain



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

DTU Wind Energy
Department of Wind Energy

Outline

- Motivation
- Theoretical modeling background
- Investigated cases – Askervein and Bolund
- Simulation results – Direct comparisons between results and measurements
- Comparison of simulation times
- Conclusions

Can OpenFOAM be directly used to simulate Atmospheric Boundary Layer (ABL) related problems?

- Open Source CFD code
- No Licensing Costs
- Includes many modules, among them a wind turbine siting module
- Interesting questions regarding its usage:
    - How fast (easy) -
    - How accurate – can the relevant CFD results be accomplished
- In this case compared relative to performance of EllipSys3D

Methods in the Numerical Approach

- **Navier-Stokes (NS) Equations**

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

- **RANS k-eps. Turbulence closure**

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

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

Wall modeling approaches I

- **Atmospheric roughness approach (Richards and Hoxey)**

$$s = \frac{u_*}{\kappa} \ln\left(\frac{z + z_0}{z_0}\right) , \qquad u_0 = k^{\frac{1}{2}} C_\mu{}^{\frac{1}{4}} ,$$

$$k = \frac{u_*^2}{C_\mu^{1/2}} , \qquad u_w = \frac{\kappa s}{\ln\left(\frac{\Delta z + z_0}{z_0}\right)} ,$$

$$\tau_0 = \rho\, u_0\, u_w ,$$

$$\epsilon = \frac{u_0^3}{\kappa\,(\Delta z + z_0)} ,$$

- k-equation boundary condition

$$P_k = \nu_T \left|\overline{S}\right|^2$$

Wall modeling approaches II

- **Nikuradse's equivalent sand roughness length modeling approach**

$$s = \frac{u_*}{\kappa} \ln\left(\frac{Ez}{C_s k_s}\right) \approx \frac{u_*}{\kappa} \ln\left(\frac{z}{z_0}\right) , \qquad k_s = \frac{E}{C_s} z_0 = 19.58\, z_0$$

- $C_s = 0.5$ – a roughness constant
- $E = 9.79$ – a smooth wall constant

$$\nu_T = \frac{u_* \kappa \Delta z}{\ln\left(\frac{E\Delta z}{C_s k_s}\right)} \approx \frac{u_0 \kappa \Delta z}{\ln\left(\frac{\Delta z}{z_0}\right)} .$$

$$\epsilon = \frac{u_0^3}{\kappa \Delta z} ,$$

- Position of the first (near-wall) cell center

$$\Delta z / z_0 \approx 10.0,$$

Wall modeling availability in EllipSys3D and OpenFOAM and Solver Parameters used

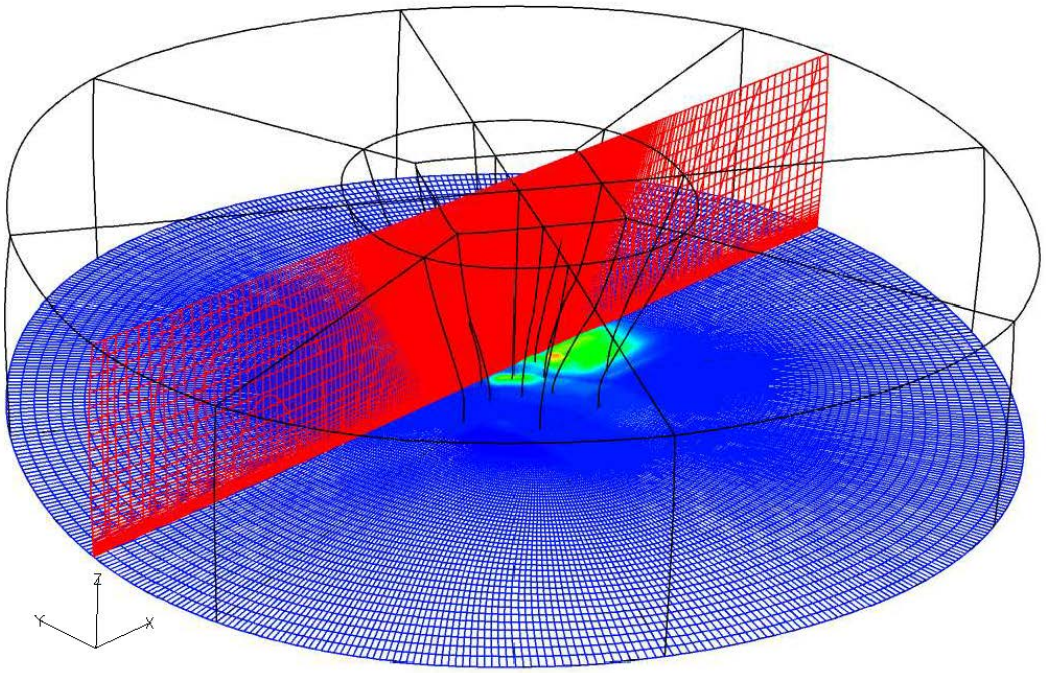
	EllipSys3D	OpenFOAM v.1.7.1	OpenFoam v.2.1.1
Richards and Hoxey Model	**X**		**X**
Nikuradse's Model		**X**	X

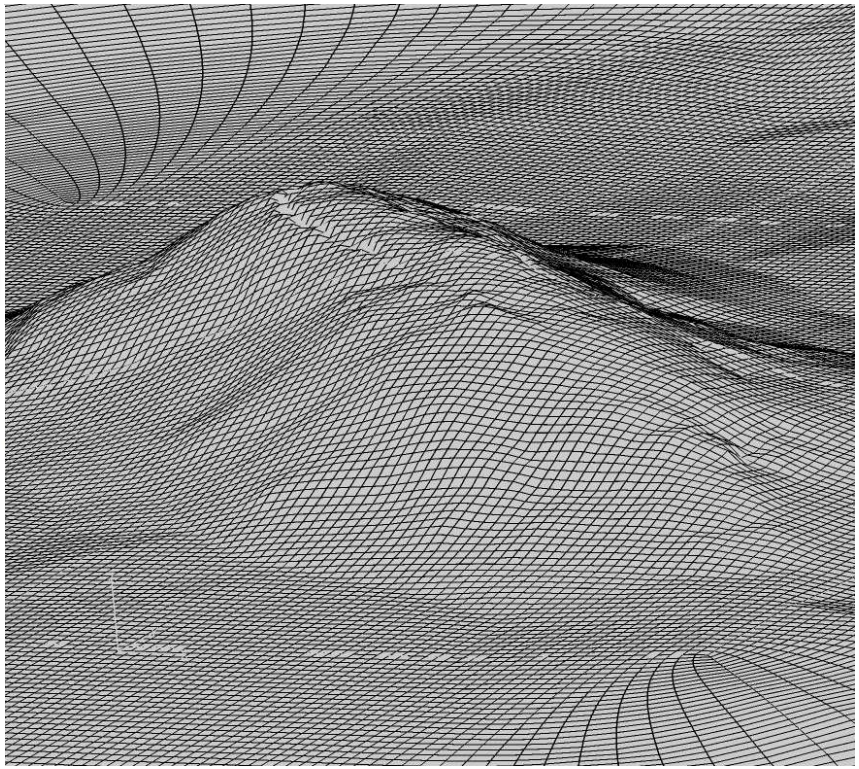
- Same solver parameters used in both OpenFOAM and EllipSys3D
  - QUICK scheme used in RANS equations
  - SIMPLE algorithm utilized for pressure correction
  - Same final convergence level obtained

Grid Generators

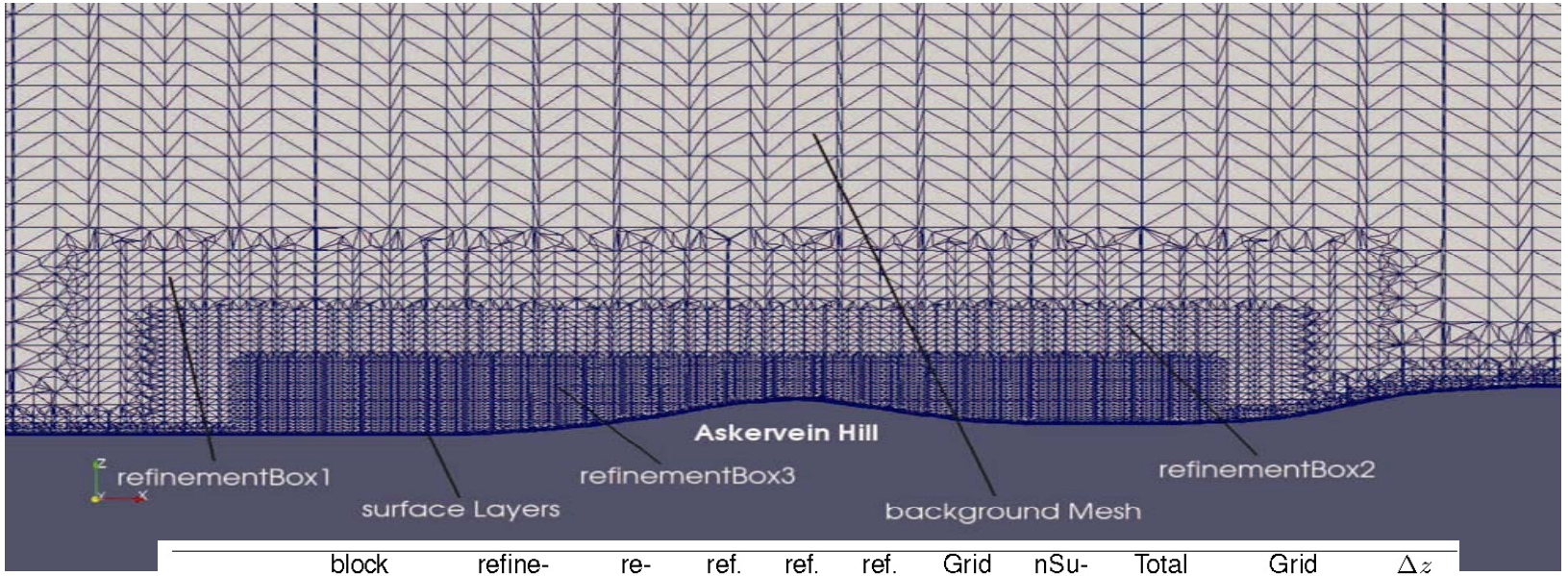
- HypGrid (EllipSys3D, OpenFOAM)
  - It creates a 3D structured hexahedron volume meshes using a hyperbolic marching scheme, based on orthogonality and cell volume from a 3D terrain grid surface definition.
- SnappyHexMesh (OpenFOAM)
  - It creates a 3D unstructured meshes based on 3D terrain surface definition (STL format) and iteratively build up a volume mesh upon it.

Askervein Test Case – HypGrid mesh





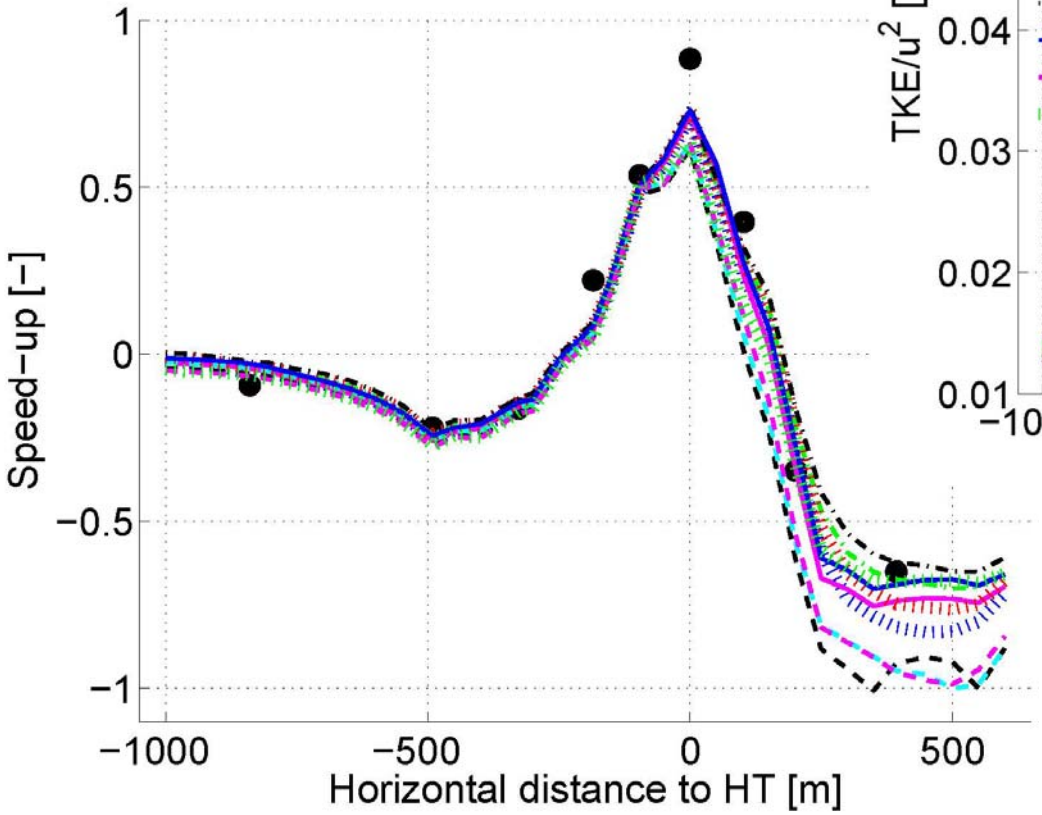
Askervein Test Case – SnapyHexMesh grid

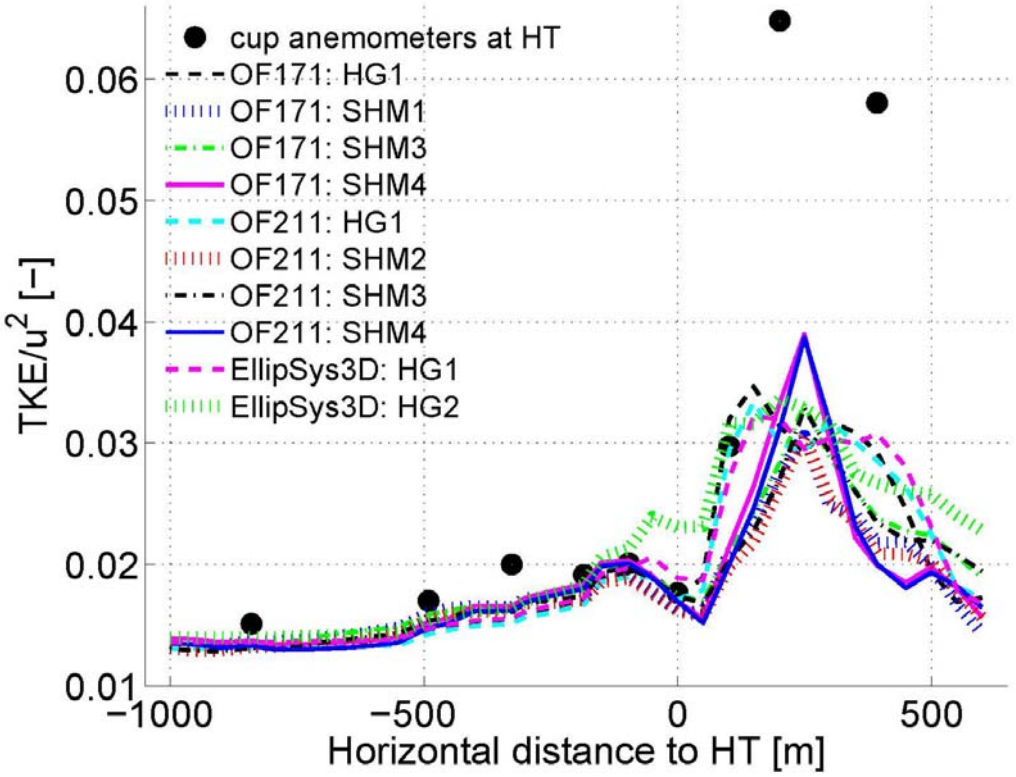


	block Mesh (back-ground grid)	refine-ment Surfa-ces	re-solve Fea-ture Angle (°)	ref. Box 1	ref. Box 2	ref. Box 3	Grid Size Prior Add Surf. Layer (mill.)	nSu-rface La-yers	Total Height of Grid Layer (m)	Grid Size Add Layer (Total) (mill.)	Δz / z_0
SHM0	60,60,40	level (2 2)	3	0	0	0	0.31	6	7.98	0.33 (0.64)	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

Table II. Askervein hill: Overview of different control parameters used to generate the SnappyHexMesh created grids. BlockMesh row shows nr. of grid points in x, y, z directions, respectively. RefinementBox 1-3 rows show the local grid refinement relative to the background grid. RefinementSurfaces row shows the minimum level of surface refinement relative to the background grid (first number) and maximum refinement - if cell intersection angle $>$ resolveFeatureAngle - (second number). 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 is indicated in fig. 3. The SHM are SnappyHexMesh based grids and HG are HypGrid based ones.

Askervein Case Results



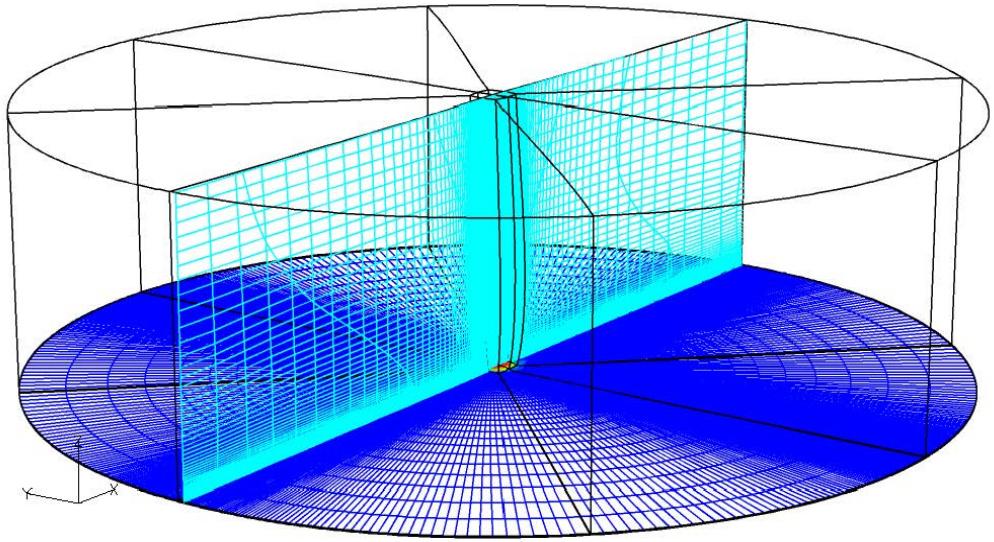


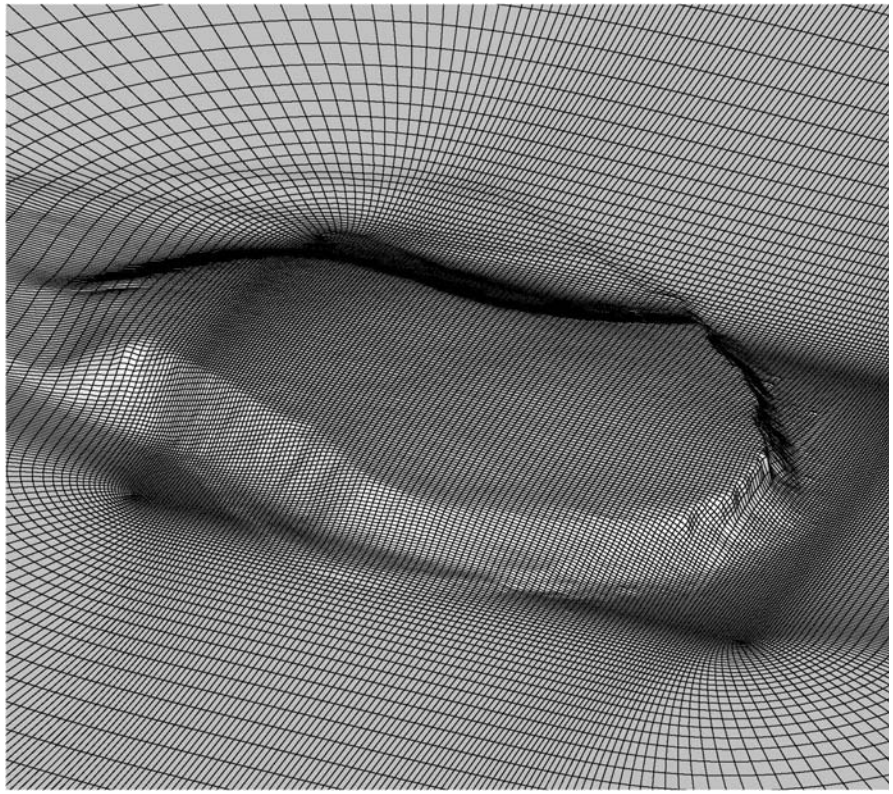
Askervein case – Computational times

			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	10259 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	14421 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

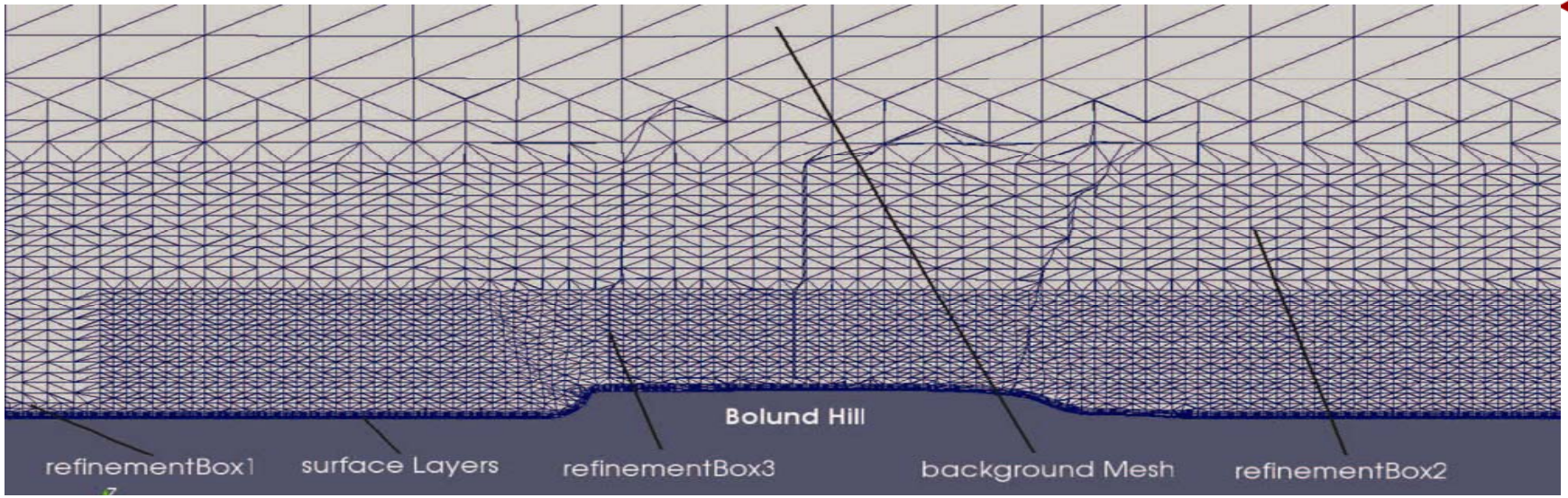
Table III. Askervein hill: Simulation times for EllipSys3D and OpenFOAM codes. For information about different grids utilized in the computations see tab. II and fig. 3.

Bolund Test Case – HypGrid mesh





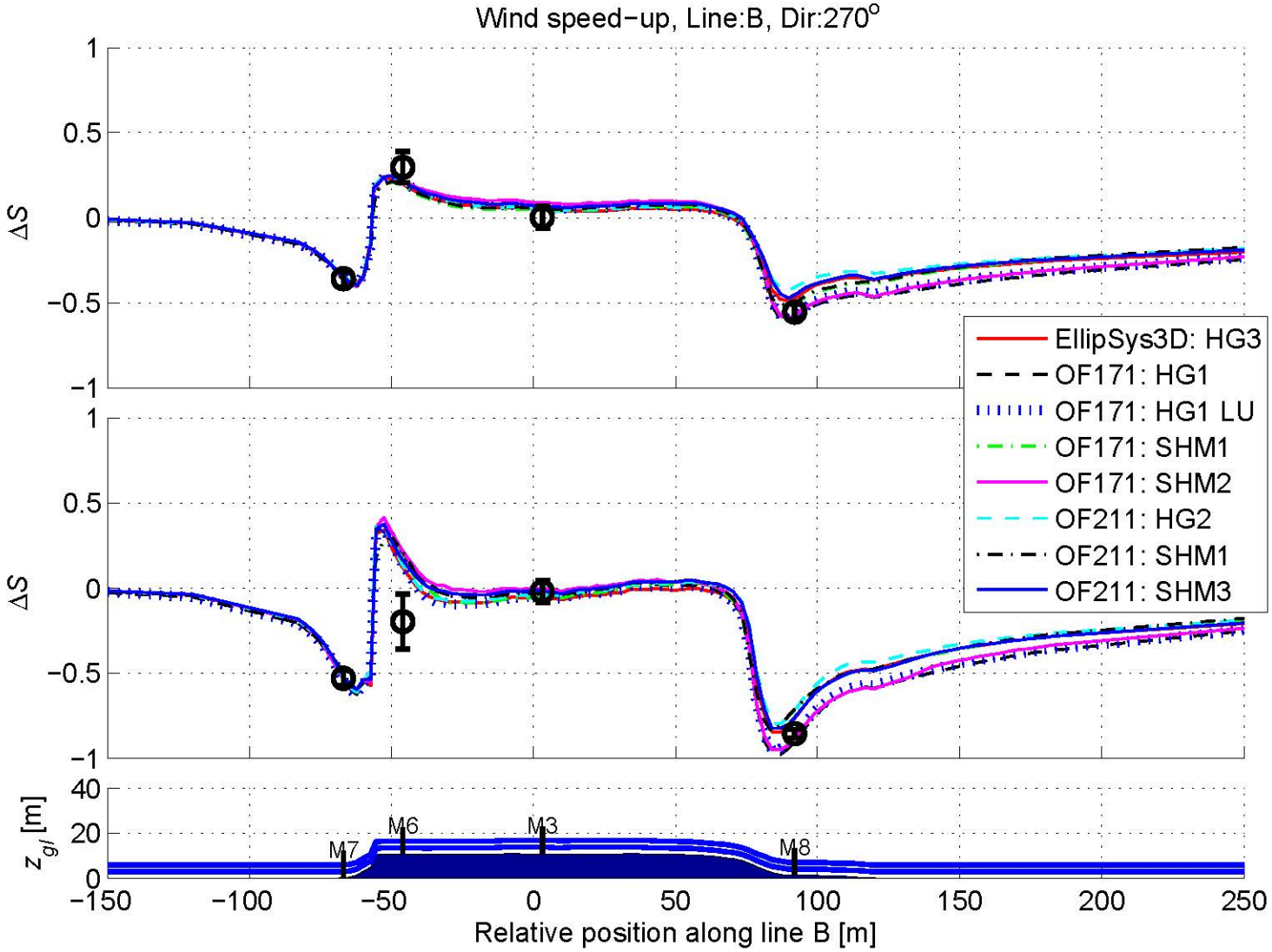
Bolund Test Case – SnappyHexMesh grid



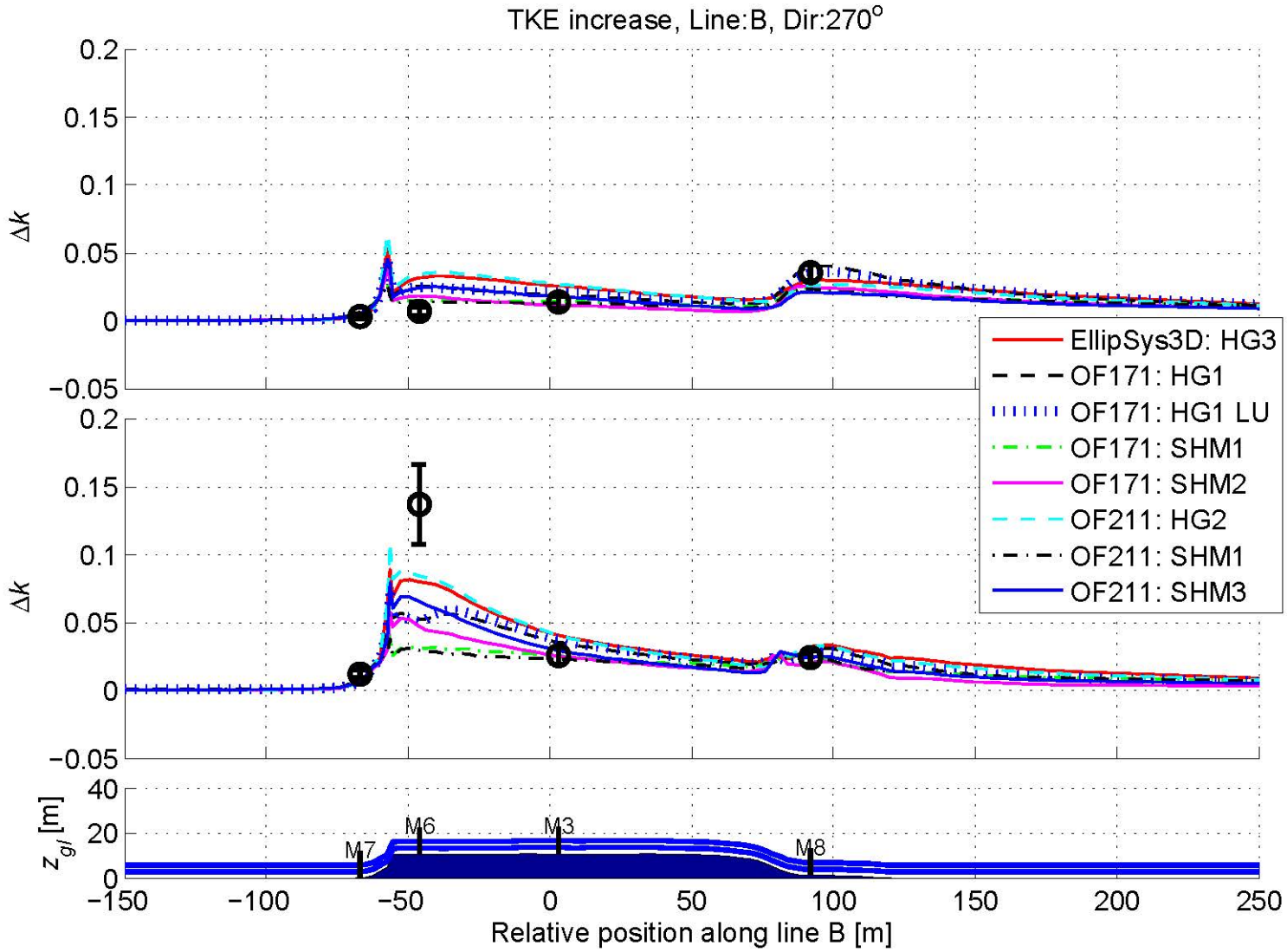
	block Mesh (back-ground grid)	refine-ment Surfa-ces	re-solve Fea-ture Angle (°)	ref. Box 1	ref. Box 2	ref. Box 3	Grid Size Prior Add Surf. Layer (mill.)	nSu-rface La-yers	Total Height of Grid Layer (m)	Grid Size Add Layer (Total) (mill.)	Δz / $z_{0(L)}$
SHM0	45,45,50	level (3 5)	2	0	0	0	0.43	3	1.60	0.41 (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

Table V. Bolund hill: Overview of different control parameters used to generate the SnappyHexMesh created grids. For definition of different parameters in the table - see tab. II. Positions of refinement boxes, surface layers and background grid is indicated in fig. 8. The SHM are SnappyHexMesh based grids and HG are HypGrid based ones. $z_{0(L)} = z_{0-Land} = 0.015\,\mathrm{m}$.

Bolund Test Case – Speed Up Results



Bolund Test Case – TKE results



Bolund Test Case – Computational times

			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	14396 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	$\Delta z = 0.83\, z_{0(L)}$	10153 (5780) s	40630 (33567) s	X		6.29
OF v.2.1.1 :	HG2	$\Delta z = 0.83\, z_{0(L)}$	10259 (5650) s	35857 (27683) 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

Table VI. Bolund hill: Simulation times for EllipSys3D and OpenFOAM codes. For information about different grids utilized in the computations see tab. V and fig. 8. The OpenFOAM v.2.2.1 runs on HG2 grid are conducted using the `PCG` pressure solver (results in parentheses) instead of `GAMG`.

Conclusions

- Mesh Generation
    - **HypGrid (HG)** – Developed for EllipSys3D, works in OpenFOAM with certain adjustments.
    - **SnappyHexMesh (SHM)** – has reasonable capability and flexibility for ABL flows, but very difficult to use. Especially grid layers near the ground difficult (in many cases impossible) to make.
- Accuracy
    - Very good general agreement between OpenFOAM and EllipSys3D.
    - Askervein case – runs on identical grid gave almost identical results, both regarding the speed up and TKE.
- Computational time
    - EllipSys3D is app. **2-6** times faster in obtaining results of similar level of accuracy on grids of similar size, utilizing the EllipSys3D default grid sequencing procedure
    - OpenFOAM **SHM** based computations found to be **3.5-7** times faster (Askervein case) and **1.8-9.8** times faster (Bolund case) then **HG** based calculations