A 1D version of EllipSys

van der Laan, Paul; Sørensen, Niels N.

Publication date:
2017

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
A 1D version of EllipSys
Contents

1 Introduction 3
2 Implementation 3
3 Grid generation 4
4 Methodology 4
  4.1 Governing equations .............................................. 4
  4.2 Grid and boundary conditions .................................... 6
5 Results and Discussion 7
  5.1 ASL ................................................................. 7
  5.2 Leipzig profile .................................................... 8
  5.3 GABLS2 ............................................................ 9
  5.4 Computational costs ................................................ 11
6 Conclusion 11

References 12

A Input files 13
  A.1 1D grid ............................................................. 13
  A.2 3D grid ............................................................. 13
  A.3 EllipSys1D ASL .................................................... 14
  A.4 EllipSys1D Leipzig ................................................. 14
  A.5 EllipSys1D GABLS2 ............................................... 15
  A.6 EllipSys3D ASL .................................................... 16
  A.7 EllipSys3D Leipzig ................................................ 21
  A.8 EllipSys3D GABLS2 ............................................... 22

DTU Wind Energy E-0141
1 Introduction

EllipSys is the in-house general purpose flow solver of DTU Wind Energy, developed by Sørensen [10] and Michelsen [8]. Currently, two and three-dimensional versions exist, referred as EllipSys2D and EllipSys3D, which are suited to perform 2D and 3D Computational Fluid Dynamics (CFD). The main applications of EllipSys are wind turbine rotor aerodynamics and atmospheric flows including terrain, which can be combined to simulate wind turbines operating in realistic flow conditions [12]. It is common to use a precursor simulation to calculate a steady state or transient atmospheric boundary layer (ABL) that can be used as inflow profiles in a CFD simulation. The precursor simulations are currently carried out using EllipSys3D, even though the flow problem is mainly one dimensional. The computational costs of a precursor simulation of a diurnal cycle using EllipSys3D is about 4 hours using 48 cores (198 CPU hours) for simulating 10 days of data. In this report, we introduce a one-dimensional version of EllipSys, referred as EllipSys1D, which can perform precursor simulations much faster while the results compare well with the results from EllipSys3D. For example, a diurnal cycle in EllipSys1D only takes about 10 min using a single core (0.18 CPU hours), which is a reduction of 3 orders of magnitude in CPU hours compared to EllipSys3D. In the atmospheric research community, EllipSys1D could be classified as single-column model. EllipSys1D could also be used for development of new models, since a model implementation in one dimension is often much easier than in two or three dimensions.

The implementation of EllipSys1D is presented in Section 2. A one dimensional grid generation tool is introduced in Section 3. In Section 4, the simulation methodology of three ABL test cases is defined. The test cases are used to compare EllipSys1D with EllipSys3D in Section 5.

2 Implementation

In order for the EllipSys1D code to generate flow that is in discrete balance with the 2D and 3D solvers, and directly allowing implementation of models from 1D to 3D, the code structure is kept nearly identical to the EllipSys2D/3D code. There is only one spatial coordinate in EllipSys1D, which refers to the height $z$ in atmospheric simulations. Therefore, all gradients with respect to $x$ and $y$ can be neglected. The grid is simply a line with distributed cells, which means that all terms related to a curvilinear grid (as used in EllipSys2D and EllipSys3D) can be removed. It is assumed that the advection in $z$ is so small that it can also be neglected, or in other words the velocity in $z$ is zero: $W = 0$. This assumption is valid for neutral steady state atmospheric profiles and an idealized periodic diurnal cycle including effects of temperature. As a result, it is not necessary to solve the $W$-momentum equation and the pressure correction equation that is normally used to ensure mass conservation. Since EllipSys1D is fast, it is chosen to make the code serial and not use the MPI libraries. In addition, the block structure (basis2D and basis3D[8]) and the multigrid are removed. A full list of features that are removed compared to the EllipSys2/3D code is given below:

- No curvilinear coordinates
- No multigrid
- No block structure (basis2D/basis3D)
- No advection
- No pressure correction solver
- No $W$-momentum
- No parallelization
- No restart

The coefficient matrix in a 1D finite volume method is tridiagonal. EllipSys1D makes use of this property by solving the coefficient matrix with a tridiagonal solver that solves the matrix very quickly. EllipSys2D and EllipSys3D have more complicated solvers that are slower than the solver implemented in EllipSys1D. This means that one needs to run more iterations in EllipSys2D and
EllipSys3D compared to EllipSys1D in order to reach the same level of convergence.

The current version of EllipSys1D (last changed at 16-01-2017) can run the following auxiliary models:

- k-\varepsilon model\[5\], k-\varepsilon f_p model\[14\], k-\varepsilon-MO model\[15\], k-\varepsilon-ABL model\[4, 9\].
- Temperature: temperature equation.
- Output: feature that outputs points of data.

3 Grid generation

The grid in EllipSys1D is a line with distributed cells. The boundary conditions are set at the start and the end of line. The line grid can be generated with a tool called linef90, which is a 1D version of boxf90 that is used to generate Cartesian 3D grids for EllipSys3D. A preprocessor step is not necessary because EllipSys1D does not use the block structure from basis2D and basis3D. An example of an input file for linef90 can be found in Appendix A.1.

4 Methodology

Three test cases based on atmospheric flows are used to compare EllipSys1D with EllipSys3D. The first test case is a logarithmic profile, representing a neutral Atmospheric Surface Layer (ASL). The second test case is a steady state ABL, based on the Leipzig measurements \[6\]. The last test case represents a periodic diurnal cycle based on the second GEWEX (Global Energy and Water cycle EXperiment) Atmospheric Boundary Layer Study (GABLS2) test case \[13\].

4.1 Governing equations

An overview of the governing equations that is valid for all test cases is presented here. The Reynolds-averaged Navier-Stokes (RANS) including the Boussinesq approximation \[3\], and Coriolis and Buoyancy forces can be written as:

\[
\frac{DU_i}{Dt} = -\frac{\partial}{\partial x_j} \left[ (\nu + \nu_T) \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] - \frac{1}{\rho} \frac{\partial P}{\partial x_i} + f_c \varepsilon_{ijk} e_k (U_j - G_j) + g_i (1 - \rho_0/\rho)
\]

Momentum
Diffusion
Pressure
Coriolis
Buoyancy
imbalance

(1)

Here we assume that \(z\) is the vertical coordinate, and \(x\) and \(y\) are horizontal coordinates. In addition, \(U_i\) are the velocity components, \(x_j\) are spatial coordinates, \(\nu\) is the molecular viscosity, \(\nu_T\) is the turbulent eddy viscosity, \(\bar{P} = P - \frac{2}{3} \rho k\) is the modified pressure with \(P\) as the pressure, \(\rho\) as the density and \(k\) as the turbulent kinetic energy. The Coriolis term is balanced by prescribed geostrophic wind \(G_i\), where \(f_c\) is the Coriolis parameter that depends on the latitude, \(\varepsilon_{ijk}\) is the Levi-Civita symbol and \(e_k\) is the normal vector in the \(z\) direction. In the buoyancy term, \(g_i = \{0, 0, g\}^T\) is the gravitational acceleration vector with \(g\) as the gravitational acceleration constant and \(\rho_0\) is the reference density.

The turbulence is modeled by a modified version of the \(k-\varepsilon\) model of Launder and Spalding \[5\]. The turbulent eddy viscosity \(\nu_T\) is defined as:

\[
\nu_T = C_{\mu} \frac{k^2}{\varepsilon},
\]

(2)
with $\varepsilon$ as the turbulent dissipation and $C_\mu$ as a constant. The turbulent quantities $k$ and $\varepsilon$ are determined from transport equations:

$$\frac{Dk}{Dt} = \frac{\partial}{\partial x_j} \left[ (\nu + \frac{\nu_T}{\sigma_k}) \frac{\partial k}{\partial x_j} \right] + \mathcal{P} - (\varepsilon - \varepsilon_{\text{amb}}) + B, \quad (3)$$

$$\frac{D\varepsilon}{Dt} = \frac{\partial}{\partial x_j} \left[ (\nu + \frac{\nu_T}{\sigma_\varepsilon}) \frac{\partial \varepsilon}{\partial x_j} \right] + \left( C_{\varepsilon,1} \mathcal{P} - C_{\varepsilon,2} \varepsilon + C_{\varepsilon,3} B \right) \frac{\varepsilon}{k} + C_{\varepsilon,4} \frac{\varepsilon_{\text{amb}}^2}{k_{\text{amb}}}, \quad (4)$$

where $\mathcal{P}$ is the turbulent production due to shear, $B$ is turbulent production or destruction due to buoyancy, $k_{\text{amb}}$ and $\varepsilon_{\text{amb}}$ are ambient value of $k$ and $\varepsilon$, and $C_{\varepsilon,1}, C_{\varepsilon,2}, C_{\varepsilon,3}, \sigma_k, \sigma_\varepsilon$ are constants or relations. $C_{\varepsilon,4}$ is defined as:

$$C_{\varepsilon,4} = C_{\varepsilon,1} + (C_{\varepsilon,2} - C_{\varepsilon,1}) \frac{\ell_t}{\ell_{t,\text{max}}}, \quad (5)$$

and it is used to limit the ABL height as introduced by Apeley and Castro [1]. $\ell_t$ is the turbulent length scale and $\ell_{t,\text{max}}$ is the maximum allowed turbulent length scale. In neutral conditions (Leipzig test case), $\ell_{t,\text{max}}$ is constant and it is determined from the relation of Blackadar $\ell_{t,\text{max}} = 0.00027G/|f_\ell|$, with $G$ as the magnitude of the geostrophic wind vector $G$. When effects of temperature are modeled, as performed in the GABLS2 test case, the buoyancy is modeled as:

$$B = -\frac{\nu_T g}{\sigma_\theta} \frac{\partial \theta}{\partial z}, \quad (6)$$

where $\theta$ is the potential temperature that is modeled by a transport equation:

$$\frac{D\theta}{Dt} = \frac{\partial}{\partial x_j} \left[ (\mu + \mu_T) \frac{\partial \theta}{\partial x_j} \right] + S_\theta, \quad (7)$$

where $S_\theta$ are heat sources and $\sigma_\theta$ is a model constant or a relation. In the GABLS2 test case, the potential temperature is relaxed towards the initial temperature profile over a time span of one day to enforce diurnal periodicity. The initial profile is a constant (289 K) up to 4000 m and then increases with 3.5 K per 1 km.

The following buoyancy related parameterizations for $\sigma_\theta$ and $C_{\varepsilon,3}$ are used:

$$\sigma_\theta = \begin{cases} 0.74 & R_i G > 0 \, \text{with } R_i G = -B j \left( \mathcal{P} + \frac{\alpha_B}{\sigma_\theta} B \right) \quad (8) \\ 0.74 (1 - 15R_i G)^{-1/4} & R_i G < 0\end{cases}$$

$$C_{\varepsilon,3} = (C_{\varepsilon,1} - C_{\varepsilon,2}) \alpha_B + 1, \quad (9)$$

where $\alpha_B$ is defined as:

$$\alpha_B = \begin{cases} 1 - \frac{\ell_t}{\ell_{t,\text{max}}} & \text{if } R_i g > 0 \\ 1 - [1 + C_{\varepsilon,2} - C_{\varepsilon,1}] \frac{\ell_t}{\ell_{t,\text{max}}} & \text{if } R_i g < 0 \, \text{with } R_i g = -B / \mathcal{P} \end{cases} \quad (10)$$

In addition, $\ell_{t,\text{max}}$ is parametrized by an integration of the ABL as introduced by Mellor and Yamada[7]:

$$\ell_{t,\text{max}} = 0.075 \int_0^\infty z \sqrt{k} dz \quad (11)$$

The density is related to the potential temperature using the ideal gas law:

$$\rho = \frac{M P_0}{R \theta}, \quad (12)$$

where $M = 29 \, \text{g/mol}$ is the average molar mass of air, $P_0 = 10^5 \, \text{Pa}$ is the standard atmospheric pressure, and $R = 8.313 \, \text{J/mol/K}$ is the universal gas constant.

The turbulence model constants and input parameters are listed in Tables 1 and 2, respectively. Note that $\ell_t$ represents the friction velocity.
Table 1: Model constants.

<table>
<thead>
<tr>
<th>Case</th>
<th>$C_{\mu}$</th>
<th>$C_{\varepsilon,1}$</th>
<th>$C_{\varepsilon,2}$</th>
<th>$C_{\varepsilon,3}$</th>
<th>$\sigma_h$</th>
<th>$\sigma_v$</th>
<th>$\kappa$</th>
<th>$\sigma_\theta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASL</td>
<td>0.03</td>
<td>1.21</td>
<td>1.92</td>
<td>0</td>
<td>1.00</td>
<td>1.30</td>
<td>0.40</td>
<td>-</td>
</tr>
<tr>
<td>Leipzig</td>
<td>0.03</td>
<td>1.21</td>
<td>1.92</td>
<td>0</td>
<td>1.00</td>
<td>1.30</td>
<td>0.40</td>
<td>-</td>
</tr>
<tr>
<td>GABLS2</td>
<td>0.03</td>
<td>1.52</td>
<td>1.833</td>
<td>eq. 9</td>
<td>2.95</td>
<td>2.95</td>
<td>0.40</td>
<td>eq. 8</td>
</tr>
</tbody>
</table>

Table 2: Input parameters.

<table>
<thead>
<tr>
<th>Case</th>
<th>$u_*$</th>
<th>$z_0$</th>
<th>$G$</th>
<th>$\ell_{\text{max}}$</th>
<th>$f_c$</th>
<th>$g$</th>
<th>$k_{\text{amb}}$</th>
<th>$\varepsilon_{\text{amb}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASL</td>
<td>0.4</td>
<td>0.05</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Leipzig</td>
<td>-</td>
<td>0.3</td>
<td>17.5</td>
<td>41.8</td>
<td>$1.13 \times 10^{-4}$</td>
<td>0</td>
<td>$10^{-4}$</td>
<td>$7.208 \times 10^{-8}$</td>
</tr>
<tr>
<td>GABLS2</td>
<td>-</td>
<td>0.03</td>
<td>9.5</td>
<td>eq. 11</td>
<td>$8.87 \times 10^{-5}$</td>
<td>9.81</td>
<td>$10^{-4}$</td>
<td>$7.208 \times 10^{-8}$</td>
</tr>
</tbody>
</table>

4.2 Grid and boundary conditions

The 1D and 3D Cartesian grids are shown in Figure 1. The grid is 6 km tall and the 3D grid is 40 m wide in horizontal dimensions, $x$ and $y$. 192 cells are used in the normal ($z$) direction, using a first cell height of 0.1 m. 4 cells with uniform spacing of 10 m is applied in the horizontal dimensions of the 3D grid, resulting in 48 blocks of $4^3$ cells.

A wall boundary condition (BC) is applied at $z = z_0$, where $z_0$ is the roughness length, which is consistent with a neutral ASL profile [11]. In the ASL test case, an inlet BC is applied at $z = 6$ km where the analytical solution is set, while a symmetric BC is used both in the Leipzig and GABLS2 test cases. The remaining side boundaries of the 3D grid are set to periodic BCs. In the
GABLES2 test case, a time varying temperature is prescribed at the wall BC that reflects a diurnal variation in temperature.

The input files of the 1D and 3D grids are given in Appendices A.1 and A.2.

5 Results and Discussion

The results of the three ABL test cases are presented in three following sections. The EllipSys1D and EllipSys3D input files are given in Appendices A.3, A.4, A.5, A.6, A.7 and A.8.

5.1 ASL

In both the 1D and 3D simulations, the flow domain is initialized by the analytical solution of the ASL:

\[
U(z) = \frac{u_*}{\kappa} \ln \left( \frac{z}{z_0} \right), \quad k = \frac{u_*^2}{\sqrt{C}} \frac{\varepsilon(z)}{\kappa^{2}}, \quad \nu_T = u_* \kappa z \tag{13}
\]

In the 1D simulation, the profiles of \( U, k \) and \( \varepsilon \) do not change up the 10th digit after 60000 iterations. In the 3D simulation, a convergence in order of \( 10^{-6} \) is achieved after 820000 iterations. This shows that the current numerical setup is quite inefficient for EllipSys3D. The final profiles are plotted in Figure 2 and compared with the analytical solution of eq. 13. The \( k \) profile and velocity profile (to a lesser extent) differ from the analytical solution because of the well known wall problem [2]. The simulated profiles compare very well with each other. The normalized difference between the 1D and 3D solution, defined as:

\[
\Delta \phi = (\phi_{1D} - \phi_{3D})/\phi_{3D}, \tag{14}
\]

is plotted in Figure 3. Note that \( \phi_{1D} \) and \( \phi_{3D} \) represent results of a flow variable of the 1D and 3D versions of EllipSys, respectively. Figure 3 shows that the difference between EllipSys1D and EllipSys3D results is in the order of the convergence level of the EllipSys3D simulation.

![Figure 2: 1D and 3D results of the logarithmic surface layer test case.](Image)
5.2 Leipzig profile

Although the flow problem of the Leipzig test case is steady state, it is solved in transient mode using a fixed time step of 100 s and 8 subiterations to avoid numerical problems in the 1D solver. When the Leipzig test case is run in steady state using the 1D solver, the flow solution diverges. The 3D solver can be used in steady state mode without numerical problems. This indicates that the 1D solver behaves more stiff compared to the 3D solver.

The 1D and 3D results of the Leipzig profiles are shown in Figure 4. A solution after $10^4$ and $10^5$ iterations are plotted for both simulations. The 1D simulation is converged after $10^4$ iteration while the 3D simulations needs $10^5$ iterations to compare well with the 1D simulation.

Figure 5 shows the normalized difference as defined in eq. (14) between the 1D and 3D simulations for a different number of iterations of the 3D simulation: $10^4$, $10^5$ and $10^6$. Note the error in wind direction is normalized by $\max (\phi_{3D}) - \min (\phi_{3D})$ to avoid a division by zero. The largest error occur around the ABL height. It is clear that the difference between the 1D and 3D simulations becomes smaller for longer run times. We can conclude the both EllipSys1D and EllipSys3D calculates the same solution of the Leipzig test case, as long as the number of iterations is large enough.
Figure 5: Normalized difference between 1D and 3D results of the Leipzig test case, as function of number of iterations.

5.3 GABLS2

The GABLS2 test case is simulated for 10 days using a time step of 1 s and 8 subiterations. The results of the last day is used to compare the 1D and 3D simulations. Contours of wind speed, wind direction, $k$ and potential temperature are plotted as function of height and time in Figure 6 for both solvers. In the bottom plots, the wall temperature is shown as a black line. The normalized differences, as defined in eq. (14), between the 1D and 3D simulations are plotted in the right column of Figure 6. Since $k$ can become close to zero, it is normalized by $\max (\phi_{3D}) - \min (\phi_{3D})$. Overall, the differences between the 1D and 3D simulations are small. The largest differences are observed around the ABL height, especially at the time of the day where the wall heats up quickly. This is also visible in Figure 7, where 4 profiles are plotted at different times of the day for both the 1D and 3D simulations. There are only small visible differences mainly occurring near the ABL height for all shown times and also below the ABL height at 12:00 h when the change in wall temperature is relatively large.
Figure 6: 1D (left column) and 3D (middle column) results and normalized difference (right column) of the GABLS2 test case. Wall temperature is shown as a black line in bottom plots.
5.4 Computational costs

The Leipzig and GABLS2 test cases represent typical precursor simulations that can be used as inlet conditions in a larger CFD simulation of flow over terrain (including wind turbines). It is often necessary to perform parametric runs of precursor simulations in order to obtain the desired inflow conditions. Therefore, it is interesting to compare the computational costs of the EllipSys1D and EllipSys3D for the Leipzig and the GABLS3 test cases, as listed in Table 3. The EllipSys1D simulation of the Leipzig test case takes 6 s on a single core (0.0017 CPU hours), while the EllipSys3D simulation takes 20 min using 48 cores (16 CPU hours). Hence, EllipSys1D is about $10^4$ faster than EllipSys3D to obtain a converged solution. The 3D simulation of the GABLS2 test case takes 4 hours and 8 minutes wall clock time using 48 cores (198 CPU hours), while the 1D simulation only takes 10 minutes and 53 seconds to complete on a single core (0.18 CPU hours), which is a reduction of 3 orders of magnitude in terms of CPU hours.

<table>
<thead>
<tr>
<th>Case</th>
<th>EllipSys1D wall clock time</th>
<th>CPU hours</th>
<th>EllipSys3D wall clock time</th>
<th>CPU hours</th>
<th>EllipSys3D/EllipSys1D ratio CPU hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leipzig</td>
<td>6 s</td>
<td>0.0017</td>
<td>20 min</td>
<td>16</td>
<td>9400</td>
</tr>
<tr>
<td>GABLS2</td>
<td>10 min 53 s</td>
<td>0.18</td>
<td>4 h 08 min</td>
<td>198</td>
<td>1100</td>
</tr>
</tbody>
</table>

Table 3: Computational cost of EllipSys1D and EllipSys3D to obtain a converged solution of the Leipzig and GABLS2 test cases.

All simulations are carried out on the Jess computer cluster, consisting of Intel Xeon E5-2680v2 processors that have 10 cores running at 2.8 GHz.

6 Conclusion

A one-dimensional version of EllipSys, labeled as EllipSys1D is presented. Three atmospheric boundary layer test cases are used to show that results of EllipSys1D are exactly the same or very similar as results of EllipSys3D, while EllipSys1D uses 3 to 4 orders of magnitude less CPU hours compared to EllipSys3D.
References


A  Input files

A.1  1D grid

ASL:

ni 193
len 6000
bc 101 201
distribution 2 0 0.00001666666666667 1 1 -1 193

Leipzig/GABLS2:

ni 193
len 6000
bc 101 601
distribution 2 0 0.00001666666666667 1 1 -1 193

A.2  3D grid

ASL:

ni 5
nj 5
nk 193
xlen 40
ylen 40
zlen 6000
xbc 501 501
ybc 502 502
zbc 101 201
x-distribution 2 0 0.25 1 1 0.25 5
y-distribution 2 0 0.25 1 1 0.25 5
z-distribution 2 0 0.00001666666666667 1 1 -1 193

Leipzig/GABLS2:

ni 5
nj 5
nk 193
xlen 40
ylen 40
zlen 6000
xbc 501 501
ybc 502 502
zbc 101 601
x-distribution 2 0 0.25 1 1 0.25 5
y-distribution 2 0 0.25 1 1 0.25 5
z-distribution 2 0 0.00001666666666667 1 1 -1 193
A.3 EllipSys1D ASL

project grid
mstep 1000000
reslim 1.d-6
density 1.225d0
viscosity 1.78406d-5
vinlet 0.d0
vfarfield 0.d0
field v value 0d0
relaxu 0.7
uinlet 0.4
ufarfield 10.0
field u value 10.0
func-const z0 0.05
func-const uStar 0.4
func-const kappa 0.4
func-const cmu 0.03
inlet u 201 300 value uStar/kappa*log((z+z0)/z0)
inlet v 201 300 value 0.0
inlet w 201 300 value 0.0
inlet tke 201 300 value uStar**2/sqrt(cmu)
inlet dtke 201 300 value uStar**3/(kappa*(z+z0))
field u value uStar/kappa*log((z+z0)/z0)
field v value 0.0
field w value 0.0
field tke value uStar**2/sqrt(cmu)
field dtke value uStar**3/(kappa*(z+z0))
turbulence kepsilon
ke_version rough
loglaw true
cel 1.20941505331
ce2 1.92
prtke_ke 1.0
pred_ke 1.3
cmu 0.03
relaxturb 0.7
roughness 0.05
te_inlet 1.00d0
ed_inlet 0.001d0
te_farfield 1.00d0
ed_farfield 0.001d0
output true
nroutput 10000
extract_var tke
extract_var dtke
extract_var vis
extract-line-zcc

A.4 EllipSys1D Leipzig

project grid
mstep 100000
reslim 1.d-8
relaxu .6d0
subiterations 8
transient true 100.d0
density 1.225d0
viscosity 1.78406d-5
uinlet 0.d0
vinlet 0.d0
func-const teamb 1.d-4
func-const edamd 7.208434d-8
field w value 0.0
field tke value teamb
field dtke value edamd
turbulence kepsilon
ke_version rough-abl
loglaw true
cmu 0.03
cel 1.20941505330508
ce2 1.92
pred_ke 1.3
prtke_ke 1.0
relaxturb 0.6
te_inlet 1.0d-1
ed_inlet 0.003d0
ufarfield 12.374368671
vfarfield 12.374368671
field u value 12.374368671
field v value 12.374368671
roughness 0.3d0
turb_cori true 0.000113
lmax 41.8d0
ambient_ke 1d-4 7.208434d-8
output true
nroutput 1000
extract_var tke
extract_var dtke
extract_var vis
extract-line-zcc

A.5 EllipSys1D GABSL2

project grid
mstep 864000
subiterations 8
reslim 1.d-4
transient true 1.d0
relaxu .7d0
density 1.225d0
viscosity 1.78406d-5
ufarfield 6.717514421
vfarfield 6.717514421
func-const teini 0.1
func-const edini 0.003d0
field u value 6.717514421
field v value 6.717514421
field \( w \) value 0.0
field \( tke \) value \( \text{teini} \)
field \( dtke \) value \( \text{edini} \)
gravitation -9.81
temperature air
prandtl 0.74 0.74
prandtl_var
inlet temp 100 151 file 1 Twalllow.dat
field temp value \( 289+(z+0.1)*3.5d-3 \)
field temp min \( 289+(4000+0.1)*3.5d-3 \)
temp_inlet 289
temp_wall 289
temp_relaxation
temp_farfield 289
relaxtemp 0.6
turbulence kepsilon
ke_version rough-abl
loglaw true
roughness 0.03
lmaxy 0.075
ambient_ke 1.0d-4 7.208d-8
kappa 0.4
cmu 0.03
cel 1.52
ce2 1.833
pred_ke 2.95131
prtke_ke 2.95131
relaxturb 6.0d-1
te_inlet 1.0d-1
ed_inlet 0.003d0
tefarfield 1.0d-1
ed_farfield 0.003d0
turb_cori true 8.87d-5
output true
noutput 600
extract_var temperature
extract_var tke
extract_var dtke
extract_var lmax
extract_var uf
extract_var heatflux
extract_var prandtltempvar
extract_var den
extract-line-zcc

A.6 EllipSys3D ASL

project grid
grid_level 1
mstep 1000000
reslim 1.d-6 1.d-5
reslimp 2.d-1
diff_scheme quick
pres_corr simple
interpolationorder 2
nrgraphout 100
nrestart 100000
relaxp 0.2d0
density 1.225d0
viscosity 1.78406d-5
vinlet 0.d0
vfarfield 0.d0
field v value 0d0
winlet 0.d0
wfarfield 0.d0
field w value 0d0
relaxu 0.7
uinlet 0.4
ufarfield 10.0
field u value 10.0
func-const z0 0.05
func-const uStar 0.4
func-const kappa 0.4
func-const cmu 0.03
inlet u 201 300 value uStar/kappa*log((z+z0)/z0)
inlet v 201 300 value 0.0
inlet w 201 300 value 0.0
inlet tke 201 300 value uStar**2/sqrt(cmu)
inlet dtke 201 300 value uStar**3/(kappa*(z+z0))
field u value uStar/kappa*log((z+z0)/z0)
field v value 0.0
field w value 0.0
field tke value uStar**2/sqrt(cmu)
field dtke value uStar**3/(kappa*(z+z0))
turbulence k epsilon
ke_version rough
loglaw true
ce1 1.20941505331
ce2 1.92
prtke_ke 1.0
pred_ke 1.3
cmu 0.03
relaxturb 0.7
roughness 0.05
te_inlet 1.00d0
ed_inlet 0.001d0
te_farfield 1.00d0
ed_farfield 0.001d0
output true
nroutput 10000
extract_var tke
extract_var dtke
extract-point 20.00000000 20.00000000 0.05113467 0.05113467
extract-point 20.00000000 20.00000000 0.15574225 0.15574225
extract-point 20.00000000 20.00000000 0.26513318 0.26513318
extract-point 20.00000000 20.00000000 0.37952610 0.37952610
extract-point 20.00000000 20.00000000 0.49914962 0.49914962
extract-point 20.00000000 20.00000000 0.62424281 0.62424281
extract-point 20.00000000 20.00000000 0.75505567 0.75505567
extract-point 20.00000000 20.00000000 0.89184953 0.89184953
A.7 EllipSys3D Leipzig

project grid
grid_level 1
mstep 1000000
mstepp 5
subiterations 8
reslim 1.d-6
reslimp 2.d-1
diff_scheme quick
pres_corr simples 1.0 1.0
pres_levels 5
interpolationorder 2
nrrestart 1000000
transient true 100.d0
relaxu .7d0
relaxp .2d0
density 1.225d0
viscosity 1.78406d-5
winlet 0.d0
uinlet 0.4d0
wfarfield 0.d0
turbulence kepsilon
ke_version rough-abl
turbcrossterms false
loglaw true
ambient_ke 1.0d-4 7.208d-8
cmu 0.03
cel 1.2094150530508
ce2 1.92
pred_ke 1.3
prtke_ke 1.0
relaxturb 6.0d-1
te_inlet 1.0d-1
ed_inlet 0.003d0
te_farfield 1.0d-1
ed_farfield 0.003d0
forceallocation
ufarfield 12.374368671
field u value 12.374368671
vfarfield 12.374368671
field v value 12.374368671
roughness 0.3d0
turb_cori true 0.000113
lmax 41.8
ambient_ke 1d-4 7.208434d-8
output true
nroutput 1000
extract_var tke
extract_var dtke
extract_var lmax
+ the extraxt-point lines as defined in the input file of Section A.6.

A.8 EllipSys3D GABL2

project grid
grid_level 1
mstep 864000
mstepp 5
subiterations 8
reslim 1.d-4
reslimp 2.d-1
diff_scheme quick
pres_corr simples 1.0 1.0
pres_levels 5
interpolationorder 2
nrrestart 1000000
transient true 1.d0
relaxu .7d0
relaxp .2d0
density 1.225d0
viscosity 1.78406d-5
winlet 0.d0
uinlet 0.4d0
wfarfield 0.d0
ufarfield 6.717614421
vfarfield 6.717614421
gravitation 0 0 -9.81
temperature air
prandtl 0.74 0.74
prandtl_var
inlet temp 100 151 file 1 Twalllow.dat
field temp value 289+(z+0.1)*3.5d-3
field temp min 289+(4000+0.1)*3.5d-3
temp_inlet 289
temp_wall 289
temp_relaxation
temp_farfield 289
relaxtemp 0.6
turbulence kepsilon
ke_version rough-abl
turbcross terms false
loglaw true
roughness 0.03
lmaxmy 0.075
ambient_ke 1.0d-4 7.208d-8
cmu 0.03
cel 1.52
ce2 1.833
pred_ke 2.95131
prtke_ke 2.95131
relaxturb6.0d-1
te_inlet 1.0d-1
ed_inlet 0.003d0
tefarfield 1.0d-1
ed_farfield 0.003d0
forceallocation
turb_cori true 8.87d-5
output true
noutput 600
extract_var temperature
extract_var tke
extract_var dtke
extract_var lmax
extract_var uf
extract_var heatflux
extract_var prandtlttempvar
+
the extraxt-point lines as defined in the input file of Section A.6.