Turbulence statistics in a spectral-element code: a toolbox for high-fidelity simulations

Ricardo Vinuesa[1], **Adam Peplinski**[1], **Marco Atzori**[1], **Lambert Fick**[2], **Oana Marin**[2], **Elia Merzari**[2], **Prabal Negi**[1], **Álvaro Tanarro**[1] **and Philipp Schlatter**[1]

[1] Linné FLOW Centre, KTH Mechanics, SE-100 44 Stockholm, Sweden
and Swedish e-Science Research Centre (SeRC), Stockholm, Sweden

[2]MCS Division, Argonne National Laboratory, 9700 Cass Ave., Lemont, Illinois 60439, USA.

Technical Report (2019)

1. Introduction

In the present document we describe a toolbox for the spectral-element code Nek5000 Fischer *et al.* (2008), aimed at computing turbulence statistics. The toolbox is presented for a small test case, namely a periodic hill with $L_x = 9h$, $L_y = 3.035h$ and $L_z = 4.5h$, where x, y and z are the horizontal, vertical and spanwise directions, respectively, and h is the hill height. The number of elements in the $xy-$plane is 442, and the number of elements in z is 19, leading to a total of 8,398 spectral elements. A polynomial order of $N = 5$ is chosen, and the mesh is generated using ICEM-CFD. In this case, transition is triggered by tripping the boundary layers on both surfaces by means of a random-volume force implemented in the subroutine `userf` in `per_hill.usr`. The toolbox presented here allows to compute mean-velocity components, the Reynolds-stress tensor as well as turbulent kinetic energy (TKE) and Reynolds-stress budgets. Note that the present toolbox allows to compute turbulence statistics in turbulent flows with one homogeneous direction (where the statistics are based on time-averaging and averaging in the homogeneous direction), and also in fully three-dimensional flows (with no periodic directions, where only time-averaging is considered). The structure of the `StatsToolbox` folder is as follows:

(i) The folder `compile` includes the files necessary to compile the main code, both when running with 2D statistics (*i.e.*, for flows with one homogeneous direction) and with 3D statistics.
(ii) The folder `run` contains all the files necessary to run the case with the statistics.
(iii) The folders `cpost3d` and `post3d` contain the files necessary to compile and run the code used to postprocess the files obtained during the run when using the 3D version of the statistics.

(iv) The folders `cinterp3d` and `interp3d` contain the files necessary to compile and run the code to interpolate spectrally the postprocessed statistics into a given set of points.

(v) The folder `post2d` contains all the necessary files to compile and run the postprocessing code corresponding to the 2D version of the statistics.

The usage of the statistics toolbox is described below, and the current version for Nek5000-v17 is available at: `https://github.com/atanarro94/new-statistics.git`

2. Description of the statistics toolbox

The statistics code is located in the `compile` folder. In particular, the `userchk` subroutine from the main, `per_hill.usr` file, calls the driver of the statistics `stat_avg`. This subroutine is located at `./stats/statistics.f`, which contains the main code for both the 2D and 3D statistics. All the variables used in `statistics.f` are defined in the common block `./inc_src/STATS`, which contains a very important parameter: `STATS3D`. If this parameter is set to `1`, then the 3D version of the statistics is used, whereas if this parameter is set to `0` the 2D version will be used. Note that this parameter is important since it defines the sizes of the variables when compiling, *i.e.*, in the 2D version of the statistics the variables are two-dimensional since the averaging in the homogeneous direction is also performed while running the simulation.

Four run-time parameters are required by the statistics toolbox, and they are all set in the `.rea` file located at `./run/per_hill.rea`. The parameters under consideration are the following

(i) `p068 STAT_COMP`. This parameter determines the number of time steps between individual samples used to obtain the time-averaged statistics. The default value is `STAT_COMP=10`, and further insight on the recommended values of `STAT_COMP` to obtain converged statistics is given in Ref. Vinuesa *et al.* (2016).

(ii) `p069 STAT_OUTP`. This parameter determines the number of time steps before writing statistics files on disk. These files will be described in detail below, but a general rule is to write two statistics files per simulation. Thus, if `p011 NSTEPS` is `1000`, then `p069 STAT_OUTP` would be set to `500`.

(iii) `p070 CHKPTSTEP`. This parameter determines the number of time steps before writing restart files on disk. We will not describe in detail the restart procedure in the present document, but the general rule is as with `p069 STAT_OUTP`, namely to set this parameter so that two sets of restart files are written per simulation.

(iv) `p071 IFCHKPTRST`. This parameter determines whether the simulation is started from the initial condition provided in the `useric` subroutine, or it is started from a restart field from a previous simulation. This will not be discussed in detail in the present document, but the idea is that when `p071 IFCHKPTRST` is set to `1` then the simulation is initiated from the restart fields.

When the subroutine driving the computation of turbulence statistics, *i.e.* `stat_avg`, is called for the first time, all the variables are initialized together with the values of the parameters defined in the `.rea` file. If the current step is a multiple of `STAT_COMP` then the subroutine `stat_compute` is called to perform the statistic computation, and if the step is a multiple of `STAT_OUTP` then the statistics files are written to disk. Note that if the 3D version of the statistics is used then the subroutine `stat_mfo_outfld3D` is used to write the fields, and if the 2D version is used then a communication step is performed first through the subroutine `stat_gl_collation`, and afterwards the writing operation is carried out through the subroutine `stat_mfo_outfld2D`. The writing operations use the `.f` field structure within Nek5000.

Regarding the `stat_compute` subroutine, it is based on the idea of performing time-averaging through the following process:

1) The time interval between the last sample taken to compute statistics and the current one is calculated and stored in the variable `dtime`.
2) The total time period over which the current statistics file is being averaged is stored in the variable `stat_atime`.
3) The present sample represents a fraction of time with respect to the total accumulated statistics of `dtime/stat_atime`, and this fraction is stored in variable `beta`.
4) Thus, the previously accumulated statistical quantities represent a fraction of time corresponding to $1 -$ `beta`, a value that is stored in the variable `alpha`.
5) The running time-average of a quantity ϕ, for which an accumulated time-average ϕ_a has been obtained so far and for which a new statistical sample ϕ_n has just been obtained, is computed as: $\overline{\phi} = \alpha\phi_a + \beta\phi_n$. Note that here $\overline{\phi}$ denotes the running time-average of ϕ.

All the terms used to compute the various tensors and budgets, both in the 2D and the 3D versions of the statistics, are described in §4. Note that the process for time-averaging outlined above is the main operation performed in the 3D version of the statistics, whereas in the 2D version the averaging over the homogeneous direction is performed in addition to it. For instance, the following Fortran 90 code is used in the `stat_compute` subroutine to compute the average of the horizontal velocity component:

```
!   <u>t
if (STATS3D.eq.0) then
lnvar = lnvar + 1
npos = lnvar
call stat_compute_1Dav1(slvel(1,1,1,1,1),npos,alpha,beta)
else
call add2sxy(STAT(1,npos),alpha,vx,beta,ntot)
```

```
endif
```

If the 3D version of the statistics is considered, then the subroutine add2sxy from the math.f file in the core of Nek5000 is used to compute the time average of the vx variable as described above, in terms of alpha and beta. On the other hand, if the 2D version of the statistics is considered, then the stat_compute_1Dav1 subroutine is called. In this routine, the averaging in the homogeneous direction is performed first, then the time-averaging is carried out in a similar way as above, using the add2sxy subroutine. Note that this subroutine is currently compatible with meshes extruded using the n2to3 tool from Nek, and in the current version the homogeneous direction is assumed to be z. This subroutine essentially performs local averages inside the elements, seeks for all the elements aligned in the homogeneous direction, and then calculates the mean of the locally-averaged values. Note that a similar process is used in more complicated terms, such as for instance in the horizontal velocity fluctuation term. This term would be computed as: $\overline{u^2} = \overline{\tilde{u}\tilde{u}} - U^2$, where $\overline{\tilde{u}\tilde{u}}$ is the time-average of the square of the instantaneous signal $\tilde{u}$, minus the square of the mean value U. Since the mean value is only available after averaging all the statistic files, in the run-time statistics code we compute the term $\overline{\tilde{u}\tilde{u}}$, and the final value $\overline{u^2}$ is computed in a post processing step. The code employed to compute $\overline{\tilde{u}\tilde{u}}$ is the following:

```
!  <uu>t
if (STATS3D.eq.0) then
lnvar = lnvar + 1
npos = lnvar
call stat_compute_1Dav2(slvel(1,1,1,1,1),slvel(1,1,1,1,1),
$ npos,alpha,beta)
else
call col3(STAT_UU,vx,vx,ntot)
call add2sxy(STAT(1,npos),alpha,STAT_UU,beta,ntot)
endif
```

In the 3D version of the statistics, the subroutine col3 from the math.f file is used first to compute the square of vx, and then the subroutine add2sxy is used to perform the time-average operation as above. A similar process is carried out in the 2D version of the statistics, where the subroutine stat_compute_1Dav2 is a modified version of stat_compute_1Dav1 which performs the averaging in the homogeneous direction of the square of vx first, then it calculates the time average through add2sxy. In the basic version of the toolbox presented here, a total of 44 quantities are computed and stored, both in the 2D and 3D version of the statistics. Note that this is defined by the parameter STAT_LVAR in the STATS common block. The averaged quantities are stored in the variable STAT_RUAVG in the case of the 2D statistics, and in STAT in the 3D version. Note however that only one set of variables (2D or 3D) is compiled, as indicated by the parameter STATS3D. Moreover, the averaged quantities are stored in the same field number in both versions of the statistics, as summarized in Table 1. A more detailed description of the various terms under consideration is given in §4, and further information regarding turbulence statistics can be found in, for instance, Ref. Pope (2008).

Variable	Position	Variable	Position
<u>t	1	<pdwdz>t	23
<v>t	2	<uuu>t	24
<w>t	3	<vvv>t	25
<p>t	4	<www>t	26
<uu>t	5	<uuv>t	27
<vv>t	6	<uuw>t	28
<ww>t	7	<vvu>t	29
<pp>t	8	<vvw>t	30
<uv>t	9	<wwu>t	31
<vw>t	10	<wwv>t	32
<uw>t	11	<ppp>t	33
<pu>t	12	<pppp>t	34
<pv>t	13	<uvw>t	35
<pw>t	14	<uuuu>t	36
<pdudx>t	15	<vvvv>t	37
<pdudy>t	16	<wwww>t	38
<pdudz>t	17	<e11>t = (du/dx)**2 + (du/dy)**2 + (du/dz)**2	39
<pdvdx>t	18	<e22>t = (dv/dx)**2 + (dv/dy)**2 + (dv/dz)**2	40
<pdvdy>t	19	<e33>t = (dw/dx)**2 + (dw/dy)**2 + (dw/dz)**2	41
<pdvdz>t	20	<e12>t = (du/dx)*(dv/dx) + (du/dy)*(dv/dy) + (du/dz)*(dv/dz)	42
<pdwdx>t	21	<e13>t = (du/dx)*(dw/dx) + (du/dy)*(dw/dy) + (du/dz)*(dw/dz)	43
<pdwdy>t	22	<e23>t = (dv/dx)*(dw/dx) + (dv/dy)*(dw/dy) + (dv/dz)*(dw/dz)	44

Table 1: List of the 44 variables computed in the statistics toolbox, and position where they are stored in the corresponding variable, *i.e.*, STAT_RUAVG in the case of the 2D statistics, and STAT in the 3D version. Note that in the nomenclature employed here, <u>t denotes time-average of the time signal $\tilde{u}$ in the 3D version; in the 2D version of the code, averaging in the homogeneous direction is performed in addition to the time-averaging.

3. Postprocessing of the statistics files

3.1. *Postprocessing using the 3D version of the statistics*

When running the 3D version of the statistics toolbox, the result is the time-average of the 44 variables summarized in Table 1. Since no averaging is performed in any homogeneous direction, the resulting fields are three-dimensional. These are stored in a total of 11 files, named from s01per_hill0.f00001 to s11per_hill0.f00001. Note that one typically writes two sets of statistics files per simulation, which means that at the end of the run one would obtain 11 fields with .f00001 and 11 more with .f00002. The time-average of these two would constitute the time-average of the full run. If the complete case under consideration is split into, for instance, 10 runs, then the time-average of each of the 11 files would be obtained from 20 individual fields. The postprocessing code used to perform this operation is located at `./cpost3d/per_hill.f`. In this folder one sets up the Nek5000 with the same `SIZE` parameters as the original case, since the postprocessing files are also three-dimensional. In this code, the variable `mtimes` defines the beginning of the time-averaging period.

After setting up the case and compiling, the postprocessing code is run on the directory `./post3d`, where Nek5000 is set to postprocessing mode by setting the parameter `p011 NSTEPS` from the `.rea` file to `0`. The number of fields to read and average in time, *i.e.*, 20 in the example described above, is set through the parameter `p068` from the `.rea` file. This code computes all the statistics described in §4, including mean flow, Reynolds-stress tensor and budgets, and outposts all these terms in three-dimensional fields. The result is a total of 20 files, named `a01per_hill0.f00001` to `a22per_hill0.f00001`, containing all the three-dimensional turbulence statistics and some derivatives. These fields can then be visualized in VisIt, or further analyzed by interpolating them into a new mesh with the files located at `cinterp3d` and `interp3d`. The interpolation is done in the mesh defined through `x.fort`, `y.fort` and `z.fort`, which is created with the Matlab script `per_hill_geom.m` (located at `interp3d`) and saved at `./interp3d/ZSTAT`. The interpolation gives as result the file `./ZSTAT/int_fld` which contains the interpolated data. This file is read and further postprocessed with the matlab script `per_hill_stat.m`

3.2. *Postprocessing using the 2D version of the statistics*

In the 2D version of the statistics, the result is the average in time and in the homogeneous direction of the 44 variables summarized in Table 1, which implies that the resulting fields are two-dimensional. These are stored in a single file named `stsper_hill0.f00001`, and as in §3.1 the typical approach is to save two files per simulation: `.f00001` and `.f00002`. The postprocessing code for the 2D version of the statistics is located at `./post2d/per_hill.f`, where Nek5000 is run in postprocessing mode (namely with the parameter `p011 NSTEPS` from the `.rea` file set to `0`), but also with the dimension in the `SIZE` file set to `2`, given the fact that the statistics files are two-dimensional.

The variables in the `stsper_hill0.f` files are defined in the 2D version of the spectral-element mesh used to run the simulation. However, it is often useful to define a different mesh, where the following analysis can be performed more conveniently, for instance without the difficulties introduced by the Gauss–Lobatto–Legendre (GLL) distribution of points within elements. This mesh is defined in the Matlab script `mesh_G.m`, which produces two binary files with the mesh coordinates: `x.fort` and `y.fort`. These are stored in the folder `./ZSTAT`. As in §3.1, the number of statistics fields to average is set through the parameter `p068` from the `.rea` file, and the beginning of the time-averaging interval is defined through the variable `mtimes`. Then, when running this code all the `stsper_hill0.f` files will be read, averaged in time, and then interpolated to the mesh defined through `x.fort` and `y.fort`. The result is stored in the file `int_fld`, which is also located in the folder `./ZSTAT`. Finally, the Matlab script `per_hill_stat.m` reads the `int_fld`, and computes all the turbulence statistics, including mean flow, Reynolds-stress tensor and the corresponding budgets. The final result is therefore expressed as a series of 2D matrices in Matlab, which can be further analyzed.

4. Description of the terms

In the following, x, y and z denote the three spatial coordinates, whereas $\tilde{u}$, $\tilde{v}$ and $\tilde{w}$ are the corresponding instantaneous velocity components. Time-averages are denoted by $(\bar{\cdot})$ or capital letters (U, V and W are the mean velocity components). Lower case corresponds to fluctuating velocities, so the velocity fluctuation is defined as: $\overline{u^2} = \overline{\tilde{u}^2} - U^2$. Note that index summation notation is also used when required. Therefore, the mean velocities constitute a tensor of rank 1:

$$U_i = \begin{pmatrix} U \\ V \\ W \end{pmatrix}, \tag{1}$$

and the (symmetric) Reynolds-stress tensor has rank 2:

$$\overline{u_i u_j} = \begin{pmatrix} \overline{u^2} & \overline{uv} & \overline{uw} \\ \overline{uv} & \overline{v^2} & \overline{vw} \\ \overline{uw} & \overline{vw} & \overline{w^2} \end{pmatrix}. \tag{2}$$

Another second-order tensor corresponds to the velocity gradients of the mean velocities:

$$\partial_i U_j = \begin{pmatrix} \partial U/\partial x & \partial U/\partial y & \partial U/\partial z \\ \partial V/\partial x & \partial V/\partial y & \partial V/\partial z \\ \partial W/\partial x & \partial W/\partial y & \partial W/\partial z \end{pmatrix}. \tag{3}$$

The skewness is a third-order tensor related to the triple-product terms, and it is defined as:

$$S_{i,j,1} = \begin{pmatrix} \overline{u^3} & \overline{u^2 v} & \overline{u^2 w} \\ \overline{u^2 v} & \overline{uv^2} & \overline{uvw} \\ \overline{u^2 w} & \overline{uvw} & \overline{uw^2} \end{pmatrix}, \quad S_{i,j,2} = \begin{pmatrix} \overline{u^2 v} & \overline{uv^2} & \overline{uvw} \\ \overline{uv^2} & \overline{v^3} & \overline{v^2 w} \\ \overline{uvw} & \overline{v^2 w} & \overline{vw^2} \end{pmatrix}, \tag{4}$$

$$S_{i,j,3} = \begin{pmatrix} \overline{u^2 w} & \overline{uvw} & \overline{uw^2} \\ \overline{uvw} & \overline{v^2 w} & \overline{vw^2} \\ \overline{uw^2} & \overline{vw^2} & \overline{w^3} \end{pmatrix}. \tag{5}$$

The transport equation of the Reynolds-stress tensor can be written as:

$$\frac{\partial}{\partial t}\overline{u_i u_j} = P_{ij} + \varepsilon_{ij} + D_{ij} + T_{ij} - \Pi^s_{ij} + \Pi^t_{ij} - C_{ij}, \tag{6}$$

which involves the following tensors of rank 2: P_{ij} is the production, ε_{ij} is the pseudo dissipation (see the discussion in Ref. Pope (2008)), D_{ij} is the viscous diffusion, T_{ij} is the turbulent transport, Π^s_{ij} is the pressure strain, Π^t_{ij} is the pressure transport and C_{ij} is the convection. Time is denoted by t. The various tensors are defined as (where ρ and ν denote the fluid density and kinematic viscosity respectively, and p corresponds to the pressure):

$$P_{ij} = -\overline{u_i u_k}\frac{\partial U_j}{\partial x_k} - \overline{u_j u_k}\frac{\partial U_i}{\partial x_k}, \tag{7}$$

$$\varepsilon_{ij} = -2\nu\overline{\frac{\partial u_i}{\partial x_k}\frac{\partial u_j}{\partial x_k}}, \tag{8}$$

$$D_{ij} = \nu\frac{\partial^2}{\partial x_k^2}\overline{u_i u_j}, \tag{9}$$

$$T_{ij} = -\frac{\partial}{\partial x_k}\overline{u_i u_j u_k}, \tag{10}$$

$$\Pi^s_{ij} = -\frac{1}{\rho}\left(\overline{p\frac{\partial u_i}{\partial x_j}} + \overline{p\frac{\partial u_j}{\partial x_i}}\right), \tag{11}$$

$$\Pi^t_{ij} = -\frac{1}{\rho}\left(\frac{\partial}{\partial x_j}\overline{pu_i} + \frac{\partial}{\partial x_i}\overline{pu_j}\right). \tag{12}$$

$$C_{ij} = U_k\frac{\partial}{\partial x_k}\overline{u_i u_j}. \tag{13}$$

The pressure transport and the pressure strain tensors can be combined to form the velocity-pressure-gradient tensor Π_{ij}, defined as follows:

$$\Pi_{ij} = -\frac{1}{\rho}\left(\overline{u_i\frac{\partial p}{\partial x_j}} + \overline{u_j\frac{\partial p}{\partial x_i}}\right). \tag{14}$$

Using the chain rule:

$$\frac{\partial}{\partial x_i}\overline{pu_j} = \overline{u_j\frac{\partial p}{\partial x_i}} + \overline{p\frac{\partial u_j}{\partial x_i}}, \tag{15}$$

which implies that the velocity-pressure-gradient tensor can be computed as:

$$\Pi_{ij} = \Pi^t_{ij} - \Pi^s_{ij}. \tag{16}$$

It is also possible to write a transport equation for the turbulent kinetic energy (TKE), defined as $k = 1/2\left(\overline{u^2} + \overline{v^2} + \overline{w^2}\right)$:

$$\frac{\partial k}{\partial t} = P^k + \varepsilon^k + D^k + T^k - \Pi^{k,s} + \Pi^{k,t} - C^k, \tag{17}$$

where each of the terms can be obtained from the respective tensor, as exemplified here for the case of TKE production:

$$P^k = \frac{1}{2}\left(P_{xx} + P_{yy} + P_{zz}\right). \tag{18}$$

4.1. *Components of the production tensor*

The following terms constitute the various components of the production tensor P_{ij}, both in 2D and in 3D:

$$P_{xx}^{2D} = -2\left(\overline{u^2}\frac{\partial U}{\partial x} + \overline{uv}\frac{\partial U}{\partial y}\right), \tag{19}$$

$$P_{xx}^{3D} = -2\left(\overline{u^2}\frac{\partial U}{\partial x} + \overline{uv}\frac{\partial U}{\partial y} + \overline{uw}\frac{\partial U}{\partial z}\right). \tag{20}$$

$$P_{yy}^{2D} = -2\left(\overline{uv}\frac{\partial V}{\partial x} + \overline{v^2}\frac{\partial V}{\partial y}\right), \tag{21}$$

$$P_{yy}^{3D} = -2\left(\overline{uv}\frac{\partial V}{\partial x} + \overline{v^2}\frac{\partial V}{\partial y} + \overline{vw}\frac{\partial V}{\partial z}\right). \tag{22}$$

$$P_{zz}^{2D} = -2\left(\overline{uw}\frac{\partial W}{\partial x} + \overline{vw}\frac{\partial W}{\partial y}\right), \tag{23}$$

$$P_{zz}^{3D} = -2\left(\overline{uw}\frac{\partial W}{\partial x} + \overline{vw}\frac{\partial W}{\partial y} + \overline{w^2}\frac{\partial W}{\partial z}\right), \tag{24}$$

$$P_{xy}^{2D} = -\left(\overline{u^2}\frac{\partial V}{\partial x} + \overline{uv}\frac{\partial U}{\partial x} + \overline{uv}\frac{\partial V}{\partial y} + \overline{v^2}\frac{\partial U}{\partial y}\right), \tag{25}$$

$$P_{xy}^{3D} = -\left(\overline{u^2}\frac{\partial V}{\partial x} + \overline{uv}\frac{\partial U}{\partial x} + \overline{uv}\frac{\partial V}{\partial y} + \overline{v^2}\frac{\partial U}{\partial y} + \overline{uw}\frac{\partial V}{\partial z} + \overline{vw}\frac{\partial U}{\partial z}\right). \tag{26}$$

$$P_{xz}^{2D} = -\left(\overline{u^2}\frac{\partial W}{\partial x} + \overline{uw}\frac{\partial U}{\partial x} + \overline{uv}\frac{\partial W}{\partial y} + \overline{vw}\frac{\partial U}{\partial y}\right), \tag{27}$$

$$P_{xz}^{3D} = -\left(\overline{u^2}\frac{\partial W}{\partial x} + \overline{uw}\frac{\partial U}{\partial x} + \overline{uv}\frac{\partial W}{\partial y} + \overline{vw}\frac{\partial U}{\partial y} + \overline{uw}\frac{\partial W}{\partial z} + \overline{w^2}\frac{\partial U}{\partial z}\right). \tag{28}$$

$$P_{yz}^{2D} = -\left(\overline{uv}\frac{\partial W}{\partial x} + \overline{uw}\frac{\partial V}{\partial x} + \overline{v^2}\frac{\partial W}{\partial y} + \overline{vw}\frac{\partial V}{\partial y}\right), \tag{29}$$

$$P_{yz}^{3D} = -\left(\overline{uv}\frac{\partial W}{\partial x} + \overline{uw}\frac{\partial V}{\partial x} + \overline{v^2}\frac{\partial W}{\partial y} + \overline{vw}\frac{\partial V}{\partial y} + \overline{vw}\frac{\partial W}{\partial z} + \overline{w^2}\frac{\partial V}{\partial z}\right). \tag{30}$$

4.2. *Components of the pseudo dissipation tensor*

As discussed by Pope Pope (2008), the difference between the pseudo dissipation and the dissipation is very small in virtually all cases. Expressing the budget in terms of the pseudo dissipation leads to a small additional term, which would be added to the viscous diffusion discussed below. The following terms constitute the various components of the pseudo dissipation tensor ε_{ij}, which are the same in 2D and in 3D:

$$\varepsilon_{xx}^{2D,3D} = -2\nu\left[\overline{\left(\frac{\partial u}{\partial x}\right)^2} + \overline{\left(\frac{\partial u}{\partial y}\right)^2} + \overline{\left(\frac{\partial u}{\partial z}\right)^2}\right]. \tag{31}$$

$$\varepsilon_{yy}^{2D,3D} = -2\nu \left[\overline{\left(\frac{\partial v}{\partial x}\right)^2} + \overline{\left(\frac{\partial v}{\partial y}\right)^2} + \overline{\left(\frac{\partial v}{\partial z}\right)^2} \right]. \tag{32}$$

$$\varepsilon_{zz}^{2D,3D} = -2\nu \left[\overline{\left(\frac{\partial w}{\partial x}\right)^2} + \overline{\left(\frac{\partial w}{\partial y}\right)^2} + \overline{\left(\frac{\partial w}{\partial z}\right)^2} \right]. \tag{33}$$

$$\varepsilon_{xy}^{2D,3D} = -2\nu \left[\overline{\frac{\partial u}{\partial x}\frac{\partial v}{\partial x}} + \overline{\frac{\partial u}{\partial y}\frac{\partial v}{\partial y}} + \overline{\frac{\partial u}{\partial z}\frac{\partial v}{\partial z}} \right]. \tag{34}$$

$$\varepsilon_{xz}^{2D,3D} = -2\nu \left[\overline{\frac{\partial u}{\partial x}\frac{\partial w}{\partial x}} + \overline{\frac{\partial u}{\partial y}\frac{\partial w}{\partial y}} + \overline{\frac{\partial u}{\partial z}\frac{\partial w}{\partial z}} \right]. \tag{35}$$

$$\varepsilon_{yz}^{2D,3D} = -2\nu \left[\overline{\frac{\partial v}{\partial x}\frac{\partial w}{\partial x}} + \overline{\frac{\partial v}{\partial y}\frac{\partial w}{\partial y}} + \overline{\frac{\partial v}{\partial z}\frac{\partial w}{\partial z}} \right]. \tag{36}$$

4.3. *Components of the viscous diffusion tensor*

The following terms constitute the various components of the viscous diffusion tensor D_{ij}, which are the same in 2D and in 3D:

$$D_{xx}^{2D,3D} = \nu \left(\frac{\partial^2}{\partial x^2}\overline{u^2} + \frac{\partial^2}{\partial y^2}\overline{u^2} + \frac{\partial^2}{\partial z^2}\overline{u^2} \right). \tag{37}$$

$$D_{yy}^{2D,3D} = \nu \left(\frac{\partial^2}{\partial x^2}\overline{v^2} + \frac{\partial^2}{\partial y^2}\overline{v^2} + \frac{\partial^2}{\partial z^2}\overline{v^2} \right). \tag{38}$$

$$D_{zz}^{2D,3D} = \nu \left(\frac{\partial^2}{\partial x^2}\overline{w^2} + \frac{\partial^2}{\partial y^2}\overline{w^2} + \frac{\partial^2}{\partial z^2}\overline{w^2} \right). \tag{39}$$

$$D_{xy}^{2D,3D} = \nu \left(\frac{\partial^2}{\partial x^2}\overline{uv} + \frac{\partial^2}{\partial y^2}\overline{uv} + \frac{\partial^2}{\partial z^2}\overline{uv} \right). \tag{40}$$

$$D_{xz}^{2D,3D} = \nu \left(\frac{\partial^2}{\partial x^2}\overline{uw} + \frac{\partial^2}{\partial y^2}\overline{uw} + \frac{\partial^2}{\partial z^2}\overline{uw} \right). \tag{41}$$

$$D_{yz}^{2D,3D} = \nu \left(\frac{\partial^2}{\partial x^2}\overline{vw} + \frac{\partial^2}{\partial y^2}\overline{vw} + \frac{\partial^2}{\partial z^2}\overline{vw} \right). \tag{42}$$

4.4. *Components of the turbulent transport tensor*

The following terms constitute the various components of the turbulent transport tensor T_{ij}, both in 2D and in 3D:

$$T_{xx}^{2D} = -\left(\frac{\partial}{\partial x}\overline{u^3} + \frac{\partial}{\partial y}\overline{u^2 v}\right), \tag{43}$$

$$T_{xx}^{3D} = -\left(\frac{\partial}{\partial x}\overline{u^3} + \frac{\partial}{\partial y}\overline{u^2 v} + \frac{\partial}{\partial z}\overline{u^2 w}\right). \tag{44}$$

$$T_{yy}^{2D} = -\left(\frac{\partial}{\partial x}\overline{uv^2} + \frac{\partial}{\partial y}\overline{v^3}\right), \tag{45}$$

$$T_{yy}^{3D} = -\left(\frac{\partial}{\partial x}\overline{uv^2} + \frac{\partial}{\partial y}\overline{v^3} + \frac{\partial}{\partial z}\overline{v^2 w}\right). \tag{46}$$

$$T_{zz}^{2D} = -\left(\frac{\partial}{\partial x}\overline{uw^2} + \frac{\partial}{\partial y}\overline{vw^2}\right), \tag{47}$$

$$T_{zz}^{3D} = -\left(\frac{\partial}{\partial x}\overline{uw^2} + \frac{\partial}{\partial y}\overline{vw^2} + \frac{\partial}{\partial z}\overline{w^3}\right). \tag{48}$$

$$T_{xy}^{2D} = -\left(\frac{\partial}{\partial x}\overline{u^2 v} + \frac{\partial}{\partial y}\overline{uv^2}\right), \tag{49}$$

$$T_{xy}^{3D} = -\left(\frac{\partial}{\partial x}\overline{u^2 v} + \frac{\partial}{\partial y}\overline{uv^2} + \frac{\partial}{\partial z}\overline{uvw}\right). \tag{50}$$

$$T_{xz}^{2D} = -\left(\frac{\partial}{\partial x}\overline{u^2 w} + \frac{\partial}{\partial y}\overline{uvw}\right), \tag{51}$$

$$T_{xz}^{3D} = -\left(\frac{\partial}{\partial x}\overline{u^2 w} + \frac{\partial}{\partial y}\overline{uvw} + \frac{\partial}{\partial z}\overline{uw^2}\right). \tag{52}$$

$$T_{yz}^{2D} = -\left(\frac{\partial}{\partial x}\overline{uvw} + \frac{\partial}{\partial y}\overline{v^2 w}\right), \tag{53}$$

$$T_{yz}^{3D} = -\left(\frac{\partial}{\partial x}\overline{uvw} + \frac{\partial}{\partial y}\overline{v^2 w} + \frac{\partial}{\partial z}\overline{vw^2}\right). \tag{54}$$

4.5. *Components of the velocity-pressure-gradient tensor*

The following terms constitute the various components of the viscous diffusion tensor Π_{ij}, which are the same in 2D and in 3D:

$$\Pi_{xx}^{2D,3D} = -\frac{2}{\rho}\overline{u\frac{\partial p}{\partial x}}. \tag{55}$$

$$\Pi_{yy}^{2D,3D} = -\frac{2}{\rho}\overline{v\frac{\partial p}{\partial y}}. \tag{56}$$

$$\Pi_{zz}^{2D,3D} = -\frac{2}{\rho}\overline{w\frac{\partial p}{\partial z}}. \tag{57}$$

$$\Pi_{xy}^{2D,3D} = -\frac{1}{\rho}\left(\overline{u\frac{\partial p}{\partial y}} + \overline{v\frac{\partial p}{\partial x}}\right). \tag{58}$$

$$\Pi_{xz}^{2D,3D} = -\frac{1}{\rho}\left(\overline{u\frac{\partial p}{\partial z}} + \overline{w\frac{\partial p}{\partial x}}\right). \tag{59}$$

$$\Pi_{yz}^{2D,3D} = -\frac{1}{\rho}\left(\overline{v\frac{\partial p}{\partial z}} + \overline{w\frac{\partial p}{\partial y}}\right). \tag{60}$$

4.6. *Components of the pressure strain tensor*

The following terms constitute the various components of the pressure strain tensor Π_{ij}^{s}, which are the same in 2D and in 3D:

$$\Pi_{xx}^{s,2D,3D} = -\frac{2}{\rho}\overline{p\frac{\partial u}{\partial x}}. \tag{61}$$

$$\Pi_{yy}^{s,2D,3D} = -\frac{2}{\rho}\overline{p\frac{\partial v}{\partial y}}. \tag{62}$$

$$\Pi_{zz}^{s,2D,3D} = -\frac{2}{\rho}\overline{p\frac{\partial w}{\partial z}}. \tag{63}$$

$$\Pi_{xy}^{s,2D,3D} = -\frac{1}{\rho}\left(\overline{p\frac{\partial u}{\partial y}} + \overline{p\frac{\partial v}{\partial x}}\right). \tag{64}$$

$$\Pi_{xz}^{s,2D,3D} = -\frac{1}{\rho}\left(\overline{p\frac{\partial u}{\partial z}} + \overline{p\frac{\partial w}{\partial x}}\right). \tag{65}$$

$$\Pi_{yz}^{s,2D,3D} = -\frac{1}{\rho}\left(\overline{p\frac{\partial v}{\partial z}} + \overline{p\frac{\partial w}{\partial y}}\right). \tag{66}$$

4.7. *Components of the pressure transport tensor*

The following terms constitute the various components of the pressure transport tensor Π_{ij}^{t}, which are the same in 2D and in 3D:

$$\Pi_{xx}^{t,2D,3D} = -\frac{2}{\rho}\frac{\partial}{\partial x}\overline{pu}. \tag{67}$$

$$\Pi_{yy}^{t,2D,3D} = -\frac{2}{\rho}\frac{\partial}{\partial y}\overline{pv}. \tag{68}$$

$$\Pi_{zz}^{t,2D,3D} = -\frac{2}{\rho}\frac{\partial}{\partial z}\overline{pw}. \tag{69}$$

$$\Pi_{xy}^{t,2D,3D} = -\frac{1}{\rho}\left(\frac{\partial}{\partial y}\overline{pu} + \frac{\partial}{\partial x}\overline{pv}\right). \tag{70}$$

$$\Pi_{xz}^{t,2D,3D} = -\frac{1}{\rho}\left(\frac{\partial}{\partial z}\overline{pu} + \frac{\partial}{\partial x}\overline{pw}\right). \tag{71}$$

$$\Pi_{yz}^{t,2D,3D} = -\frac{1}{\rho}\left(\frac{\partial}{\partial z}\overline{pv} + \frac{\partial}{\partial y}\overline{pw}\right). \tag{72}$$

4.8. *Components of the convection tensor*

The following terms constitute the various components of the convection tensor C_{ij}, both in 2D and in 3D:

$$C_{xx}^{2D} = U\frac{\partial}{\partial x}\overline{u^2} + V\frac{\partial}{\partial y}\overline{u^2}, \tag{73}$$

$$C_{xx}^{3D} = U\frac{\partial}{\partial x}\overline{u^2} + V\frac{\partial}{\partial y}\overline{u^2} + W\frac{\partial}{\partial z}\overline{u^2}. \tag{74}$$

$$C_{yy}^{2D} = U\frac{\partial}{\partial x}\overline{v^2} + V\frac{\partial}{\partial y}\overline{v^2}, \tag{75}$$

$$C_{yy}^{3D} = U\frac{\partial}{\partial x}\overline{v^2} + V\frac{\partial}{\partial y}\overline{v^2} + W\frac{\partial}{\partial z}\overline{v^2}. \tag{76}$$

$$C_{zz}^{2D} = U\frac{\partial}{\partial x}\overline{w^2} + V\frac{\partial}{\partial y}\overline{w^2}, \tag{77}$$

$$C_{zz}^{3D} = U\frac{\partial}{\partial x}\overline{w^2} + V\frac{\partial}{\partial y}\overline{w^2} + W\frac{\partial}{\partial z}\overline{w^2}. \tag{78}$$

$$C_{xy}^{2D} = U\frac{\partial}{\partial x}\overline{uv} + V\frac{\partial}{\partial y}\overline{uv}, \tag{79}$$

$$C_{xy}^{3D} = U\frac{\partial}{\partial x}\overline{uv} + V\frac{\partial}{\partial y}\overline{uv} + W\frac{\partial}{\partial z}\overline{uv}. \tag{80}$$

$$C_{xz}^{2D} = U\frac{\partial}{\partial x}\overline{uw} + V\frac{\partial}{\partial y}\overline{uw}, \tag{81}$$

$$C_{xz}^{3D} = U\frac{\partial}{\partial x}\overline{uw} + V\frac{\partial}{\partial y}\overline{uw} + W\frac{\partial}{\partial z}\overline{uw}. \tag{82}$$

$$C_{yz}^{2D} = U\frac{\partial}{\partial x}\overline{vw} + V\frac{\partial}{\partial y}\overline{vw}, \tag{83}$$

$$C_{yz}^{3D} = U\frac{\partial}{\partial x}\overline{vw} + V\frac{\partial}{\partial y}\overline{vw} + W\frac{\partial}{\partial z}\overline{vw}. \tag{84}$$

5. Description of the volume-force tripping

The tripping used in this case to trigger the turbulent boundary layer is implemented as a weak random volume forcing which acts in the vertical direction. This tripping is based on the work by Schlatter and Örlü Schlatter & Örlü (2012) where a detailed description of the parameters and functions defining the tripping is included, and it has been used in complex simulations such as the direct numerical simulation (DNS) of a wing by Hosseini et al. Hosseini *et al.* (2016).

The tripping code is located in the `compile` folder. In particular, in the `per_hill.usr` file which calls the driver of the tripping `tripf`. This subroutine is located at `./tripf.f`, which contains the main code for the tripping except some subroutines which are included in `per_hill.usr`. All the variables used in `tripf.f` are defined in the common block `./inc_src/TRIPF`.

The parts of the tripping code that need to be added in the `.usr` file are:

- The definition of the Gaussian forcing in the subroutine `userf`. Note that this definition must be changed according to the case under study (*e.g.* tripping along the 4 walls of a square duct) since the routine in the example is only valid for two tripping lines in the z direction.
- Two lines in `userchk` to call the subroutines that read the tripping parameters, `readtrip_par`, and drive the tripping, `tripf`.
- In `usrdat3`, the code that reads the tripping parameters again (`call readwallfile`) and finds the GLL points in which the tripping is applied.

In order to run with tripping, two files are required in the `run` folder: `per_hill.wall` and `forparam.i`. Both of these files contain the parameters needed for tripping. Regarding the `per_hill.wall` file, the included parameters are:

- `nwalls`: determines the number of tripping lines to be used in the problem.
- `nwallpar`: number of global parameters (*i.e.* defined in the same file).
- `npwallpar`: number of parameters rows for each wall (coordinates of the line are not considered as a row). In the current version of the tripping this value is equal to 1 referring to the line in which the smoothing lengths in each direction are defined.
- Global parameters: these parameters are computed following the guidelines from Ref. Schlatter & Örlü (2012). The parameters correspond to: time-independent amplitude, time-dependent amplitude, temporal cut-off scale (*i.e.* $t_s = 4\delta^*/U_\infty$), number of modes for the top wall in the periodic hill (*i.e.* $z_s = L/(1.7\delta^*)$ where L is the length of the tripping line) and the number of modes of the bottom wall of the periodic hill. Note that δ^* is the displacement thickness of the laminar boundary layer at the inflow.

- Parameters per wall: usually consists of 3 rows for each wall in which the following parameters are defined: the direction of the tripping (*i.e.* x, y or z), the coordinates of position for the tripping line (note that the coordinates of the points in the direction of the tripping line are not needed) and the smoothing parameters of the Gaussian, function which are determined as $\sigma_x = 4\delta^*$, $\sigma_y = \delta^*$ and $\sigma_z = 0$ for a case with x as the streamwise direction, y as the vertical direction and z as the direction in which the tripping is applied.

The other file in which the tripping parameters are defined is `forparam.i`. These parameters correspond, in sequential order, to:

1. x coordinate of the first tripping line (it must match the first tripping line defined in `per_hill.wall`).
2. y coordinate of the first tripping line.
3. x coordinate of the second tripping line.
4. y coordinate of the second tripping line.
5. σ_x for the first tripping line.
6. σ_y for the first tripping line.
7. σ_x for the second tripping line.
8. σ_y for the second tripping line.
9. Temporal cutoff scale t_s.
10. Time-independent amplitude.

References

Fischer, P. F., Lottes, J. W. & Kerkemeier, S. G. 2008 Nek5000: open source spectral element cfd solver (2008). available at `http://nek5000.mcs.anl.gov`.

Hosseini, S. M., Vinuesa, R., Schlatter, P., Hanifi, A. & Henningson, D. S. 2016 Direct numerical simulation of the flow around a wing section at moderate Reynolds number. *Int. J. Fluid Mech.* **61**, 117–128.

Pope, S. B. 2008 Turbulent flows.

Schlatter, P. & Örlü, R. 2012 Turbulent boundary layers at moderate reynolds numbers: inflow length and tripping effects. *Int. J. Fluid Mech.* **710**, 5–34.

Vinuesa, Ricardo, Prus, Cezary, Schlatter, Philipp & Nagib, Hassan M 2016 Convergence of numerical simulations of turbulent wall-bounded flows and mean cross-flow structure of rectangular ducts. *Meccanica* **51** (12), 3025–3042.