PREPRINT, under review for publication, 2015

A Web Services-accessible database of turbulent channel flow and its use for testing a new integral wall model for LES

J. Graham[1], K. Kanov[1], X.I.A. Yang[1], M. K.Lee[2], N. Malaya[2], C.C. Lalescu[1], R. Burns[1], G. Eyink[1], A. Szalay[1], R.D. Moser[2], and C. Meneveau[1]

[1]The Johns Hopkins University, Baltimore, MD 21218, USA
[2]University of Texas, Austin, TX 78712, USA

August 22, 2015

Abstract

The output from a direct numerical simulation (DNS) of turbulent channel flow at $Re_\tau \approx 1000$ is used to construct a publicly and Web-services accessible, spatio-temporal database for this flow. The simulated channel has a size of $8\pi h \times 2h \times 3\pi h$, where h is the channel half-height. Data are stored at $2048 \times 512 \times 1536$ spatial grid points for a total of 4000 time samples every 5 time steps of the DNS. These cover an entire channel flow-through time, i.e. the time it takes to traverse the entire channel length $8\pi h$ at the mean velocity of the bulk flow. Users can access the database through an interface that is based on the Web-services model and perform numerical experiments on the slightly over 100 terabytes (TB) DNS data on their remote platforms, such as laptops or local desktops. Additional technical details about the pressure calculation, database interpolation and differentiation tools are provided in several appendices. As a sample application of the channel flow database, we use it to conduct an *a-priori* test of a recently introduced integral wall model for Large Eddy Simulation of wall-bounded turbulent flow. The results are compared with those of the equilibrium wall model, showing the strengths of the integral wall model as compared to the equilibrium model.

1 Introduction

Direct numerical simulation (DNS) of turbulent flows at ever higher Reynolds number has become feasible [1–4] thanks to the rapid expansion of high performance computing infrastructures. Still, community access to the large datasets that typically arise from DNS poses a challenge.Publicly accessible DNS-derived databases of turbulence fields [5–12] have demonstrated the benefits of sharing DNS-generated field data to the turbulence research community, in addition to the sharing of turbulence statistics as is done e.g. in Refs. [13–16]. The database approach can provide easy access and help in translating vast DNS data into meaningful knowledge. In particular, Web-services equipped databases for DNS have been developed in recent years [5, 6, 17–19]. The Johns Hopkins Turbulence Databases (JHTDB, at *http://turbulence.pha.jhu.edu*) has been providing public access to large scale DNS data of isotropic homogeneous turbulence [6, 20] at Taylor-scale Reynolds number $Re_\lambda \approx 430$ and forced MHD turbulence at $Re_\lambda \approx 170$ [21] through a specially adapted web services interface. Subroutine-like tools (*getfunctions*) are provided such as *getVelocityAndPressure, getVelocityGradient, getPressureLaplacian*, etc. [6] and users can perform numerical experiments on multiple terabyte datasets from platforms such as laptops using familiar programming languages (C, Fortran, Matlab, and Python). Since its inception, JHTDB has facilitated a number of research tasks. In order to give an idea of the range of possible applications of such databases, we point out that users have leveraged the database in studies of extreme events [22] and shape evolution [23] in turbulence, in the study of particles-turbulence interactions [24], turbulent dispersion [21, 25–27], for calibration of experimental techniques [28–32], for the study of gradient tensor properties [33–40], subgrid-scale model assessments [41, 42] and many others [18, 19, 43–55].

In this paper we describe the addition of a large-scale DNS of turbulent channel flow to JHTDB. The channel flow DNS is conducted at a friction velocity-based Reynolds number of $Re_\tau \approx 1000$, and velocity and pressure fields of a whole flow-through time of size $\approx$ 100 TB are stored. The data volume of each time-step is subdivided into small chunks or database atoms, which are temporally and spatially indexed and stored in a SQL Server database. Detailed description of the database layout and temporal and spatial partitioning of the data is provided in Kanov et al. [17] and the supplementary material of Eyink et al. [21]. The databases storing the data occupy several of the nodes of the DataScope cluster at JHU [56].

The organization of the article is as follows. We first discuss the DNS flow parameters in section 2. Since the pressure had to be computed separately for ingestion into the database, some details about the pressure calculation are also presented. Additional information concerning the usage of the database and the generation of the DNS data is included in appendices. Then, as a sample application we use the data to conduct an *a-priori* test of a new wall model for Large Eddy Simulations (LES), the integral wall model (iWMLES) introduced in [57], and compare it to a traditional equilibrium wall model.

2 DNS approach and simulation parameters

The data are obtained from a direct numerical simulation (DNS) of fully developed wall-bounded channel flow with periodic boundary conditions in the longitudinal (x) and transverse (z) directions, and no-slip conditions at the top and bottom walls ($y/h = \pm 1$, where h is the half-channel height). The simulation is performed using the petascale DNS channel flow code PoongBack [58]. The code solves the incompressible Navier-Stokes equations using a wall–normal, velocity–vorticity formulation [59]. This formulation uses a vertical velocity and vorticity representation by which the pressure calculation is eliminated and continuity is explicitly satisfied. The benefit of this approach is that the need for specifying the correct pressure boundary condition is avoided, especially when explicit time integration is used. The governing equations are solved using a Fourier-Galerkin pseudo-spectral method for the longitudinal and transverse directions. A seventh-order Basis-splines (B-splines) collocation method is used to discretize in the wall normal direction (see Appendix A for more information). Dealiasing is performed using the 3/2-rule [60]. Temporal integration is performed using a low-storage, third-order Runge-Kutta method. In order to increase the time-step without violating the CFL condition, the Navier-Stokes equations are solved in a frame of reference moving in the +x direction at a speed of $0.45U_b$, where U_b is the bulk channel mean velocity. In this frame, the Dirichlet boundary condition used for the streamwise velocity at the wall is $u(y/h = \pm 1) = -0.45U_b$ while the other components still vanish.

Initially, the flow is driven using a constant volume flux control (imposing U_b in the wall-attached frame of reference, or $0.55U_b$ in the moving frame of the simulation) until stationary conditions are reached. Then the control is changed to a constant applied mean pressure gradient forcing term equivalent to the time-averaged shear stress resulting from the prior steps. Additional iterations, amounting to four flow-through times, are then performed to further achieve statistical stationarity before outputting velocity and pressure fields for ingestion into the database. Before ingesting to the database, the simulation frame velocity of $0.45U_b$ is added to the streamwise component, although the grid-points at which data are stored are those moving at speed $0.45U_b$, i.e the x-coordinate of grid points obeys $x = x_0 + 0.45U_b t$, where t is the elapsed time.

The DNS parameters are shown in Table 2. The simulation is well resolved, as the mesh is designed using standard resolution heuristics for wall-bounded simulations [61]. In particular, the x and z-directions, $\Delta x^+ \approx 13$, $\Delta z^+ \approx 7$. The y-direction uses a non-uniform mesh, with the first point at $\Delta y^+ < 1$, and $\Delta y^+ \approx \Delta z^+$ towards the center of the channel (a detailed listing is provided in the next section). A recent study [62] found that meshes generated conforming with these criterion result in discretization errors that are quite small. This indicates that the commonly used heuristics should provide adequate resolution for a well-resolved simulation.

In order to obtain the pressure field for the database, the pressure solver was implemented in the Poong-

Simulation Parameters	Values
Domain Length ($L_x \times L_y \times L_z$)	$8\pi h \times 2h \times 3\pi h$
Grid ($N_x \times N_y \times N_z$)	$2048 \times 512 \times 1536$ (wavemodes, also database spatial resolution)
	$3072 \times 512 \times 2304$ (collocation points, for the 3/2 dealiasing step)
Viscosity	$\nu = 5 \times 10^{-5} U_b h$
Mean pressure gradient	$dP/dx = 0.0025 U_b^2/h$
DNS Time step	$\Delta t = 0.0013 h/U_b$
Database time step	$\delta t = 0.0065 h/U_b$
Time stored	$t = [0, 26] h/U_b$

Table 1: Simulation parameters in units used in the simulation. There are more collocation points than wave modes in the x and z directions due to the use of the 3/2-dealiasing. Data is stored in physical space without the zero-padding, i.e. on $2048 \times 512 \times 1536$ grid-points.

Back code. One solves the pressure Poisson equation

$$\nabla^2 p = -\nabla \cdot [\nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u})] \tag{1}$$

where p is the pressure divided by density, and $\boldsymbol{u}$ the velocity. The Neumann boundary condition, expressed as

$$\left[\frac{\partial p}{\partial y} - \nu \frac{\partial^2 v}{\partial y^2}\right]_{\text{wall}} = 0 \tag{2}$$

where ν is the molecular kinematic viscosity and v the wall-normal velocity component, is used at the top and bottom walls ($y/h = \pm 1$). This calculation is performed independently from the velocity field solution only when outputting fields. The implementation and testing of the pressure solver is discussed in more detail in Appendix B.

3 Flow properties and database content

After reaching statistically stationary conditions, the simulation is continued for approximately one flow-through time. The 3 component velocity vector and pressure fields are stored every 5 time steps, resulting in 4000 frames of data. Information regarding the resulting statistical quantities are listed in table 3 and the effective resolution is indicated in Table 3. Note that the averaging operation for mean and other statistical quantities is applied in time between $t = 0$ and $t = 26$, and over x–z planes.

Statistics	Values
Bulk velocity (U_b)	0.99994
Centerline velocity (U_c)	1.1312
Friction velocity (u_τ)	4.9968×10^{-2}
Viscous length scale ($\delta_\nu = \nu/u_\tau$)	1.0006×10^{-3}
Bulk velocity Reynolds number ($Re_b = U_b 2h/\nu$)	3.9998×10^4
Centerline Reynolds number ($Re_c = U_c h/\nu$)	2.2625×10^4
Friction velocity Reynolds number ($Re_\tau = u_\tau h/\nu$)	9.9935×10^2

Table 2: Flow statistics averaged over $t = [0, 26]$. Values are in units used in the simulation.

Grid Spacing	Values
x direction (Δx^+)	12.2639
y direction at first point (Δy_1^+)	1.65199×10^{-2}
y direction at center (Δy_c^+)	6.15507
z direction (Δz^+)	6.13196

Table 3: Collocation grid spacing in viscous units.

In the following figures several basic flow characteristics are presented. In Figure 1 the friction Reynolds

number is shown as a function of simulation time, indicating statistical stationarity near the nominal value $Re_\tau \approx 1000$.

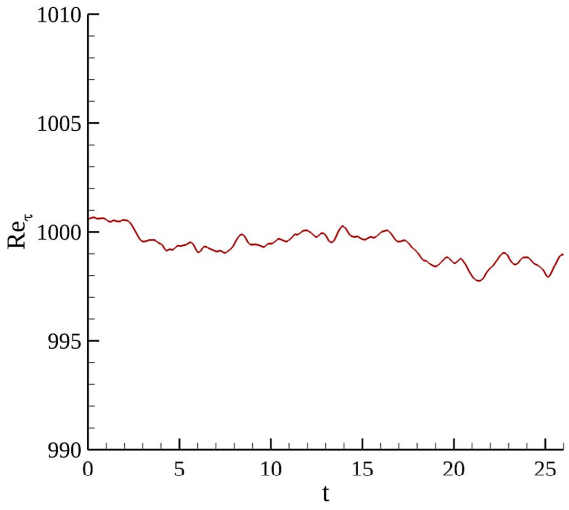


Figure 1: Friction velocity Reynolds number during the channel flow simulation during the time interval available in the database.

Figure 2 shows the mean velocity profiles for the current DNS (solid red line) and Ref. [2] (dashed red line), together with the standard $U^+ = y^+$ profile in the viscous sublayer and the log-layer with parameters $\kappa = 0.4$ and $B = 5$ (black dashed lines), which are in the commonly accepted range. It is readily apparent that the mean velocity profiles agree markedly well. The viscous and turbulent shear stresses, Reynolds normal stresses, mean pressure, pressure variance, and velocity–pressure covariances are shown in Figures 3(a)–3(d). The Reynolds stresses and pressure variance for the current DNS (solid lines) are plotted along with those from Ref. [2] (dashed lines). Minor differences between the second-order moments can be attributed to the slightly different Reynolds numbers of both DNS.

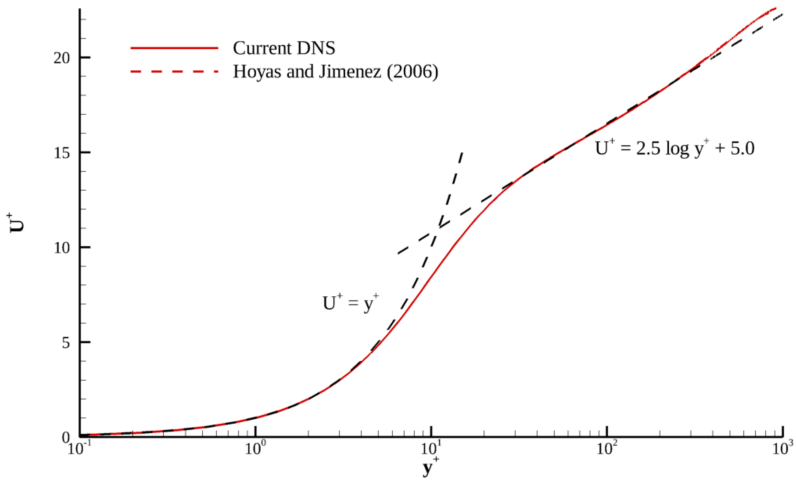
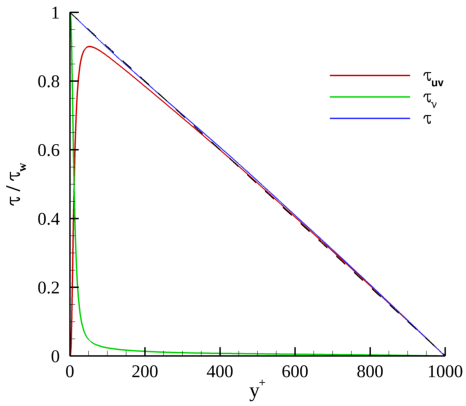
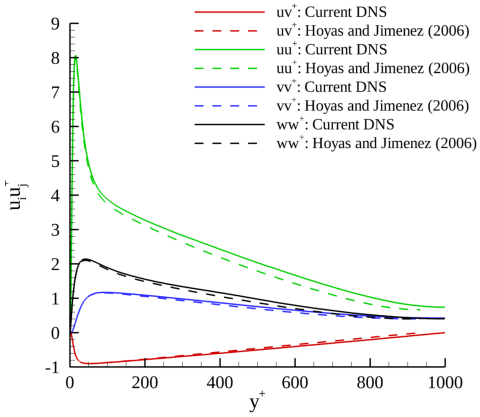


Figure 2: Mean velocity profiles in viscous units for the current DNS and Hoyas and Jiménez (2006) at $Re_\tau = 934$ [2]. Standard values of $\kappa = 0.4$ and $B = 5$ are used in the log-law (black dashed line) for reference.

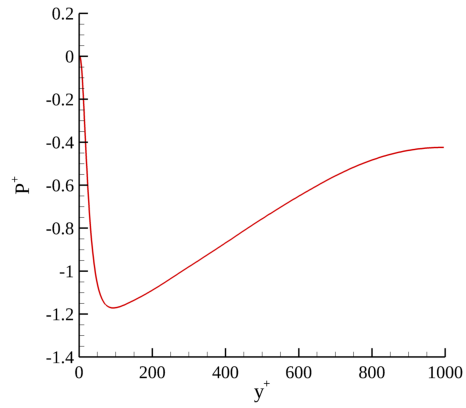
In the remaining plots, we show power spectral densities of velocity and pressure are presented for various y^+ locations, in order to clearly demonstrate that the simulations are spatially well resolved, i.e. that at high



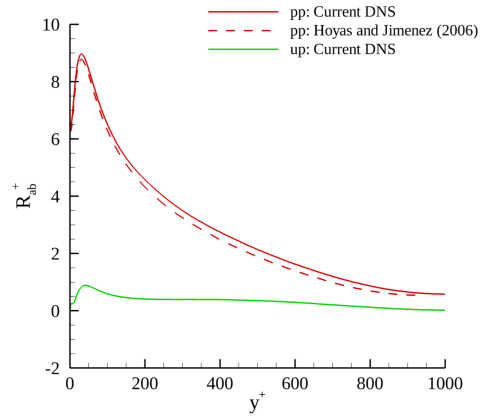
(a) Mean viscous, turbulent, and total shear stress normalized by the wall stress for the current DNS. Dashed line has slope of -1.



(b) Velocity covariances in viscous units for the current DNS and Hoyas and Jiménez (2006) at $Re_\tau = 934$ [2].



(c) Mean pressure profile in viscous units for the current DNS.



(d) Pressure variance and pressure-velocity covariance in viscous units for the current DNS and pressure variance of Hoyas and Jiménez (2006) at $Re_\tau = 934$ [2].

Figure 3: Profiles of statistical quantities from the channel flow DNS.

wavenumbers the spectra decay sufficiently fast and smoothly. Streamwise spectra are shown in Figure 4, whereas spanwise spectra are shown in Figure 5. Similarly to Figures 2 and 3(a)–3(d), the plots in Figures 4–5 also agree very well with prior published profiles, e.g. those in provided by Ref. [9].

The fields (3 velocity components and pressure) are ingested into the JHTDB following similar approach as for the isotropic turbulence datasets [6], namely indexing according to the Z-index. In a previous implementation [6] data were subdivided into data atoms consisting of 64^3 grid points with 4 additional points on all sides for edge replication, thus leading to 72^3 point data atoms. Subsequent analysis of actual database usage has shown that 8^3-point atoms without edge replication is in general more efficient. Therefore, all datasets now included in the JHTDB use 8^3-point data atoms including the channel flow dataset.

Appendix C describes the interpolation and differentiation formulae provided by the system. The methods described in [6] were adapted for the non-uniform, non-periodic grid in the wall-normal direction. In addition, spline interpolations are also implemented, to provide smooth interpolants for the applications where continuity of derivatives is desired.

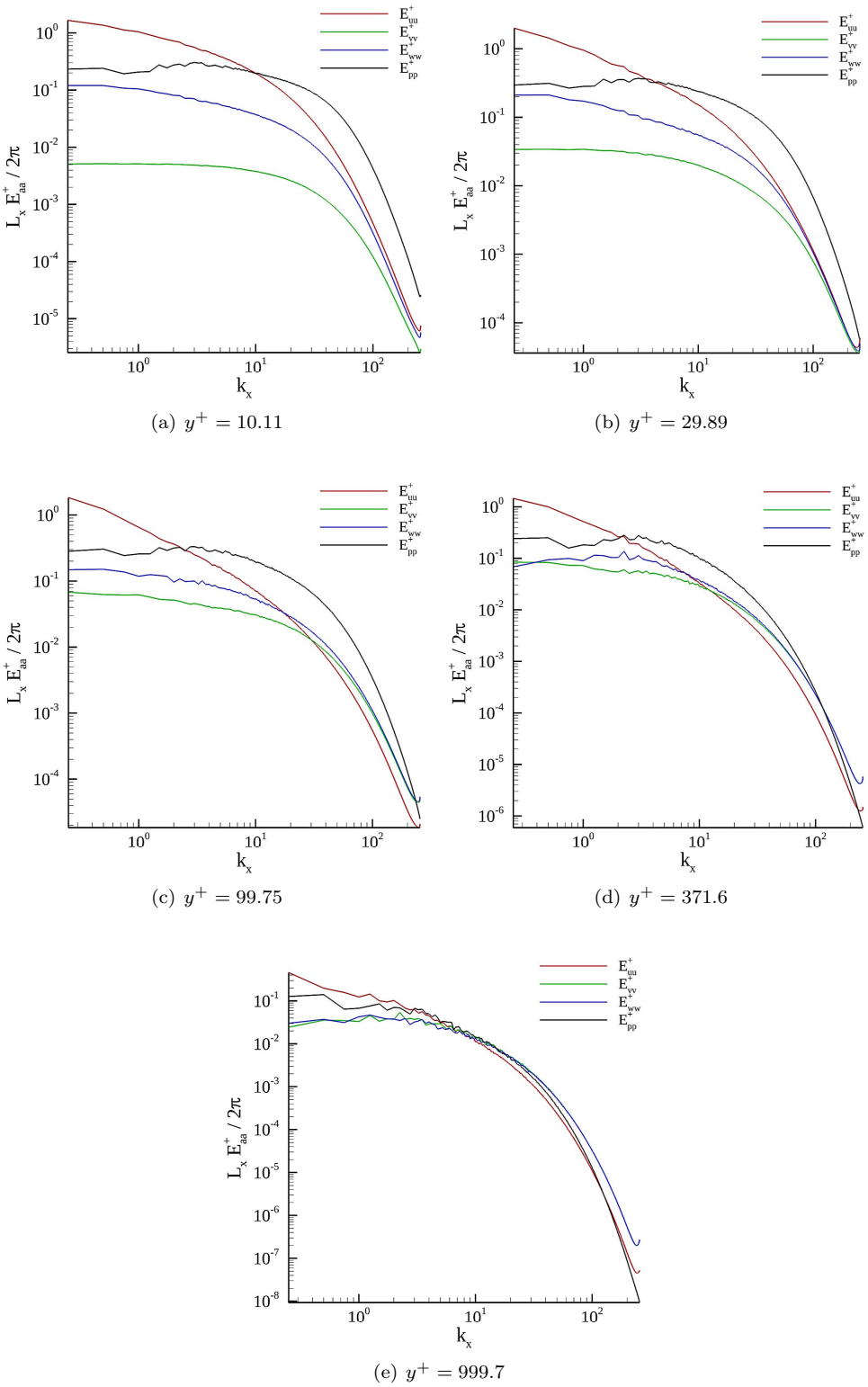
4 Sample Application to testing a new wall model

In this section we use the channel flow database to conduct an *a-priori* test of an integral wall model for Large Eddy Simulations (the iWMLES model), that has been recently proposed in another paper [57]. This wall model is briefly summarized here for the convenience of discussion, while for general reviews on wall modeling for LES, see [63, 64]. The new iWMLES model aims to model the wall stress given information about the flow away from the wall so that near-wall resolution can be relaxed during LES. iWMLES adopts the classical integral method of von Karman and Pohlhausen (VKP) [65] in which a velocity profile with various parameters is assumed. This assumption replaces numerical integration of the Reynolds-averaged boundary layer equations in the near-wall zone as is commonly done in zonal or hybrid wall models. The method also shares several features with the integral method proposed in Chung & Pullin [66].

The proposed velocity profile to be used in iWMLES contains a viscous or roughness sub-layer, and a logarithmic layer. The latter is augmented with a linear term to account for inertial and pressure gradient effects. Similar to the VKP method, the assumed velocity profile coefficients must be determined from matching and consistency conditions and then wall-normal integrals may be done analytically. The assumed functional form is for the filtered velocity $\langle u \rangle$. In order to justify the assumed profile as a possible class of solutions to the Reynolds-averaged (RANS) boundary layer equations, $\langle u \rangle$ must be thought of as a spatially filtered velocity, filtered at the LES spatial resolution in horizontal directions. In addition a temporal filter must be applied over an averaging time-scale T_{wall}, in order to damp out any fast fluctuations and allow turbulent or viscous diffusive processes to propagate towards the wall. Only then can a RANS type model be invoked and a boundary layer profiles assumed. We envision a situation in which LES is performed using a grid-spacing on the order of Δ_y, and that we therefore know from LES the wall-parallel velocity at a distance $y = \Delta_y$ away from the wall. The iWMLES model involves an assumed profile with a two-layer form:

$$\begin{aligned} \langle u \rangle &= u_\nu \frac{y}{\delta_\nu}, && 0 \le y \le \delta_i, \\ \langle u \rangle &= u_\tau \left[C + \frac{1}{\kappa} \log \frac{y}{\Delta_y} \right] + u_\tau A \frac{y}{\Delta_y}, && \delta_i < y \le \Delta_y. \end{aligned} \tag{3}$$

where $\langle u \rangle$ is the assumed profile for the filtered velocity as function of y, the wall normal direction. Δ_y is the distance from the wall where the velocity is assumed to be available if one were to perform an LES with the first vertical grid point located at a distance Δ_y away from the wall. Also, δ_i is the inner layer height, and u_ν and δ_ν are the "inner layer" velocity and length scales, respectively. u_τ is the velocity scale for the "meso layer". A is the coefficient of the linear correction term and C is the constant from the log law. We restrict the discussion to 1-D for simplicity, i.e. we do not include the spanwise velocity $\langle w \rangle$ component in the discussion. Then the 6 unknown parameters, u_ν, δ_ν, δ_i, u_τ, C and A, must be determined from the local flow conditions through additional conditions and constraints. 5 such constraints can be easily stated. Moreover,

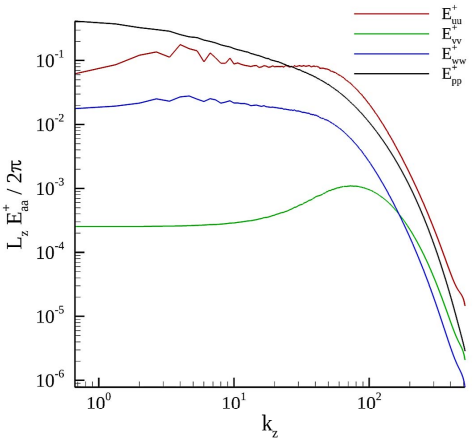


(a) $y^+ = 10.11$ (b) $y^+ = 29.89$

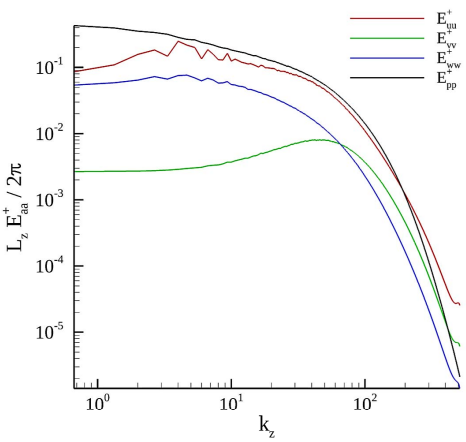
(c) $y^+ = 99.75$ (d) $y^+ = 371.6$

(e) $y^+ = 999.7$

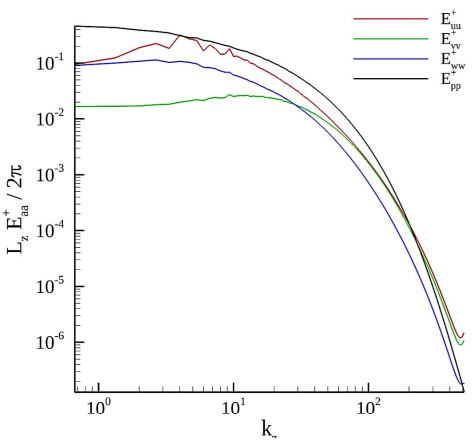
Figure 4: Streamwise power spectral densities at various y^+ locations as function of k_x



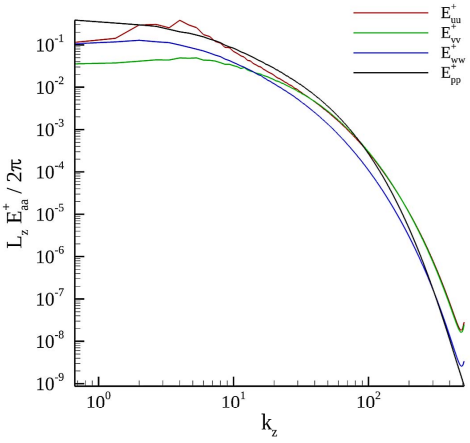
(a) $y^+ = 10.11$



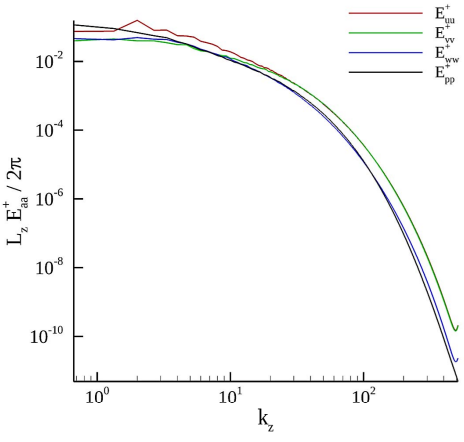
(b) $y^+ = 29.89$



(c) $y^+ = 99.75$



(d) $y^+ = 371.6$



(e) $y^+ = 999.7$

Figure 5: Spanwise power spectral densities at various y^+ locations as function of k_z

as is in VKP method, the profile shape function is substituted into the vertically integrated momentum equation, which provides the 6th condition to fully determine the 6 parameters as function of the local flow conditions.

We briefly discuss the 6 conditions, more details can be found in Ref. [57]. u_τ^2 is the overall momentum loss through the wall, and u_ν is momentum loss due to viscous effects. Since in this application the channel surface is smooth, we have $u_\tau = u_\nu$. The overall momentum loss u_τ^2 should be consistent with $\nu \partial u/\partial y|_{y=0}$. With this constraint, we can express in this case δ_ν via u_τ, obtaining $\delta_\nu = \nu/u_\tau$ (this is not generally valid for the case of rough walls). The height of inner layer δ_i is set to be $11\nu/u_\tau$, which is the intercept between the linear viscous profile $\langle u \rangle = y u_\tau^2/\nu$ and the standard log-law $\langle u \rangle = u_\tau/\kappa \ln(y u_\tau/\nu) + B$ with $B = 5$ and $\kappa = 0.4$. In the case that we find that $11\nu/u_\tau > \Delta_y$ we would conclude that we have a wall resolving situation. Then a linear profile is used from $y = 0$ to $y = \Delta_y$, with no "meso-layer". We also must use the condition of velocity matching at Δ_y, i.e. matching the assumed profile with the velocity available from the LES, U_{LES}, and impose velocity continuity at $y = \delta_i$. We can use those two conditions to express C, A in terms of u_τ. Finally, u_ν, δ_ν, C, A, δ_i all can be expressed via u_τ as follows:

$$
\begin{aligned}
u_\nu &= u_\tau, \\
\delta_\nu &= \frac{\nu}{u_\tau}, \\
\delta_i &= 11\frac{\nu}{u_\tau}, \\
A &= \left(1 - 11\frac{\nu}{u_\tau \Delta_y}\right)^{-1} \left[\frac{1}{\kappa}\log\left(11\frac{\nu}{u_\tau \Delta_y}\right) + \frac{U_{LES}}{u_\tau} - 11\right] \\
C &= \frac{U_{LES}}{u_\tau} - A.
\end{aligned}
\tag{4}
$$

The assumed profile with these parameters (that still depend on the unknown u_τ) is substituted into the momentum equation that will be integrated in the vertical direction, to solve for u_τ. A vertical, wall-normal integral of the momentum equation is also used in the wall model of Chung & Pullin [66]. The momentum equation reads, in forward Euler time-discretized form, as follows:

$$
\frac{L_x^{n+1} - L_x^n}{dt} + M_x^n = \tau_{\Delta_y}^n - \tau_w^n, \tag{5}
$$

where $L_x = \int_0^{\Delta_y} \langle u \rangle \, dy$, M_x contains the vertically integrated convective term and pressure gradient term (see below), τ_{Δ_y}, τ_w are the momentum flux at $y = \Delta_y$ and at the wall respectively. The time step size is dt. The superscript denotes the time step. All information is assumed to be known at step n. In iWMLES, the wall stress is calculated for step $n+1$ and fed back to LES.

The term M_x is defined according to:

$$
M_x = \frac{1}{\rho}\frac{\partial \langle p \rangle_{LES}}{\partial x}\Delta_y + \left[\frac{\partial L_{xx}}{\partial x} - U_{LES}\left(\frac{\partial L_x}{\partial x}\right)\right] \tag{6}
$$

where $L_{xx} = \int_0^{\Delta_y} \langle u \rangle^2 \, dy = \int_0^{\delta_i} \langle u \rangle^2 \, dy + \int_{\delta_i}^{\Delta_y} \langle u \rangle^2 \, dy$. Thus, L_x and L_{xx} (and thus M_x) and τ_{Δ_y} can all be calculated analytically from the assumed velocity profile.

Since the wall model is intended to be applied in the context of LES, we first filter the DNS data to typical LES resolution. Data are extracted from the database using the cutout service, in which users can specify indices of data to be extracted. Then, spatial filtering is performed outside of the database, namely in horizontal $x - z$ planes. We use two filtering kernels, i.e. the Gaussian kernel and the tophat filtering kernel. The assumed resolution of the LES is $\Delta x \approx 0.2h$, $\Delta z \approx 0.1h$ and we assume the first grid point away from the wall in LES is located at $\Delta_y = 0.1h$ (i.e. assuming LES has 20 grid points to resolve the channel height of $2h$). In other words, in the *a-priori* analysis, we use filtering at scales $\Delta x^+ = 192$, $\Delta_z^+ = 97$ and Δ_y is set to be 0.1. On the grid, it turns out that 75 grid points are used in DNS to resolve the layer between

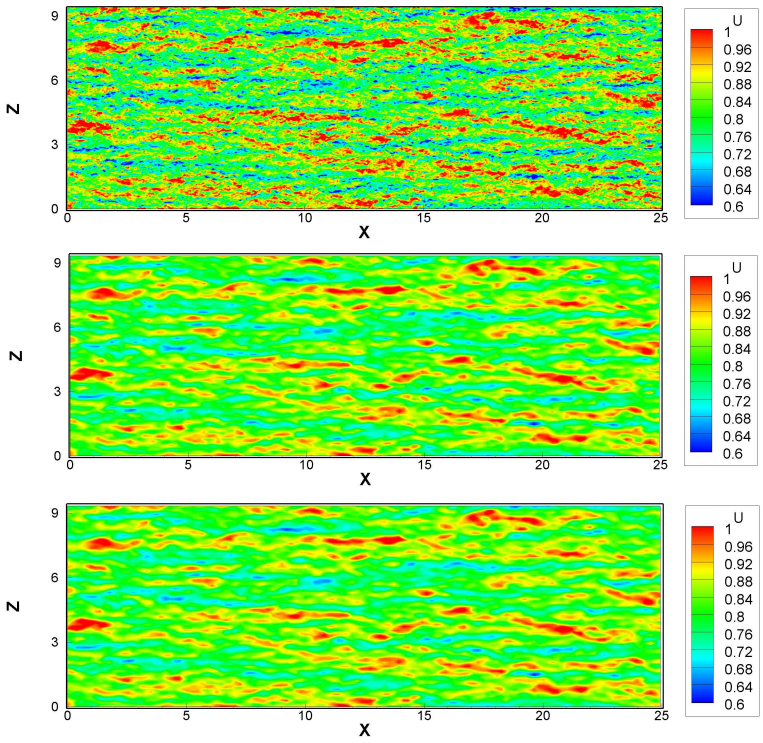


Figure 6: Contour plots of u at $y = \Delta_y$ (taking $y = 0$ to be the wall) for the unfiltered DNS, Gaussian filtered, and tophat filtered fields. The horizontal filtering is done at $\Delta x^+ = 192$ (or $\Delta_x \approx 2\Delta_y$), $\Delta_z^+ = 97$ (or $\Delta_z \approx \Delta_y$). The resolution of original DNS is $\Delta x^+ = 12$, $\Delta_z^+ = 6$.

the wall at $y = 0$ and $y = \Delta_y$. Figure 6 shows a contour plot of u at $y = \Delta_y$ from the unfiltered DNS data, the Gaussian filtered field, and the tophat filtered field. Barely any difference is visible between the tophat filtered field and Gaussian filtered field.

To test the performance of the integral wall model, we must in addition perform time averaging. As in the practical implementations of iWMLES, temporal filtering over the time scale $T_{\text{wall}} = \Delta_y/\kappa u_\tau$ is applied to the velocity field using a one-sided exponential filter:

$$U_{LES}^{n+1} = U_{LES}^{n}\left(1 - \frac{\Delta t}{T_{\text{wall}}}\right) + \tilde{u}^n \frac{\Delta t}{T_{\text{wall}}}, \tag{7}$$

where $\tilde{u}^n$ is the spatially filtered DNS velocity at time step n, and Δt is the simulation time step size (database timestep δt in the analysis of the database fields). A similar filtering is performed to obtain the pressure gradient from LES, $\partial p_{LES}^{n+1}/\partial x$. For time n, we also obtain u_τ directly from the DNS by spatially and temporally filtering $\nu \partial u/\partial y|_{y=0}$. Then we calculate u_ν, δ_ν, δ_i, C, A using Eq.(4), with the u_τ obtained from the filtered velocity field at time-step n. Then, using the new U_{LES}^{n+1} and $\partial p_{LES}^{n+1}/\partial x$, and the integrated momentum equation, we can solve for u_ν, δ_ν, δ_i, C, A and u_τ at the next time-step $n+1$. This stress can then be compared to the wall stress at time $n+1$, which is also determined from the DNS.

Because the model is active in predicting the temporal change in the wall stress and we are using the exact DNS stress at the prior time step n, we compare the time-derivative rather than the stress itself. In figure 7, we compare the time derivative of the wall stress obtained from the integral wall model and from DNS.

We also provide comparison with the predictions of the equilibrium wall model, which assumes instead that the log-law holds instantaneously and locally. For the equilibrium wall model, the wall stress is obtained simply by solving:

$$\frac{U_{LES}^{n+1}}{u_\tau} = \frac{1}{\kappa}\log\left(\frac{\Delta_y u_\tau}{\nu}\right) + B \tag{8}$$

to determine u_τ (and thus τ_w). While in practice this model would be applied locally, to provide a meaningful comparison with iWMLES that only attempts to model the 'low-frequency' components of the wall stress at time-scales longer than T_{wall}, we also apply such time filtering when evaluating the equilibrium wall model. Specifically, we use the time-filtered velocity U_{LES} as input to the equilibrium model and then compare the predicted stress (solution to Eq. 8 for u_τ and thus τ_w) to the space and time filtered wall stress ($\nu \partial u/\partial y|_{y=0}$) from the DNS.

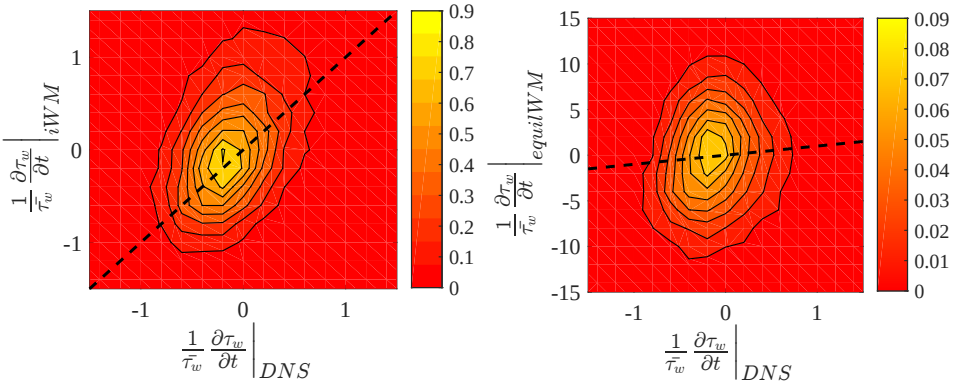


Figure 7: Scatter plots of the time derivative of the wall stress. The correlation between iWM and DNS is 0.44 for the tophat filtered flow field and 0.40 for the Gaussian filtered flow field. The correlation drops to 0.10, 0.12 for the equilibrium wall model respectively. Note the excessive range of fluctuations of predicted wall stresses for the equilibrium wall model.

Figure 8 shows the probability distribution function of $\partial \tau_w/\partial t$ from DNS, iWM, and the equilibrium wall model. We observe that by taking into consideration the effects of near wall acceleration, pressure gradient, the integral wall model can better represent the near wall physics than the equilibrium wall model. We point out that if we compared the stresses themselves, the iWMLES model shows correlation coefficients above 99%. However, this would be misleading since the stress at time-step $n+1$ is very close to that at time-step n, and we used the exact DNS value at time-step n. Similarly to the model of Chung & Pullin [66], the iWMLES model makes a prediction on the *rate of change* of the wall stress via the integrated momentum equation and that is what a fair *a-priori* test must compare in this case. Note that the PDF of $\partial \tau_w/\partial t$ suggests a non-zero mean value which would not be consistent with stationary statistics. This is because the *a-priori* test is done only on a single snapshot.

Commenting on the resulting averaging time-scale, substituting the calculated u_τ into the expression of T_{wall}, we found that 766×5 DNS time steps amounts to T_{wall}. The time filtering is carried on until about half a flow over time (which amounts to 2000×5 DNS time steps) to reduce possible effects from the finite time of the data.

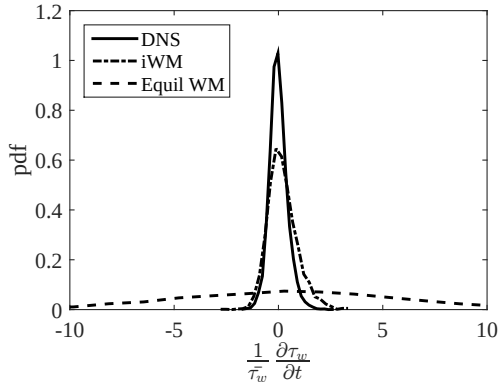


Figure 8: PDF of $\partial \tau_w/\partial t$ from DNS, the integral wall model and the equilibrium wall model. The tophat filtered flow field is used to generate this plot.

We can also directly compare the assumed profile from the model with the parameters obtained from the equations to the actual filtered profile from the DNS. We can perform the time filtering of the spatially filtered velocity at any y at some representative x and z locations using, again,

$$\langle u \rangle^{n+1}(y) = \langle u \rangle^{n}(y) \left(1 - \frac{dt}{T_{\text{wall}}}\right) + \tilde{u}^{n}(y) \frac{dt}{T_{\text{wall}}}. \tag{9}$$

Figure 9 shows a random selection of four snapshots of profiles comparing between the filtered DNS profile and the modeled profile. As can be seen, in each case the slope at the wall (the desired stress) is well reproduced. Also, a range of values of the parameter A is generated depending on the local time-smoothed pressure gradient and other flow conditions.

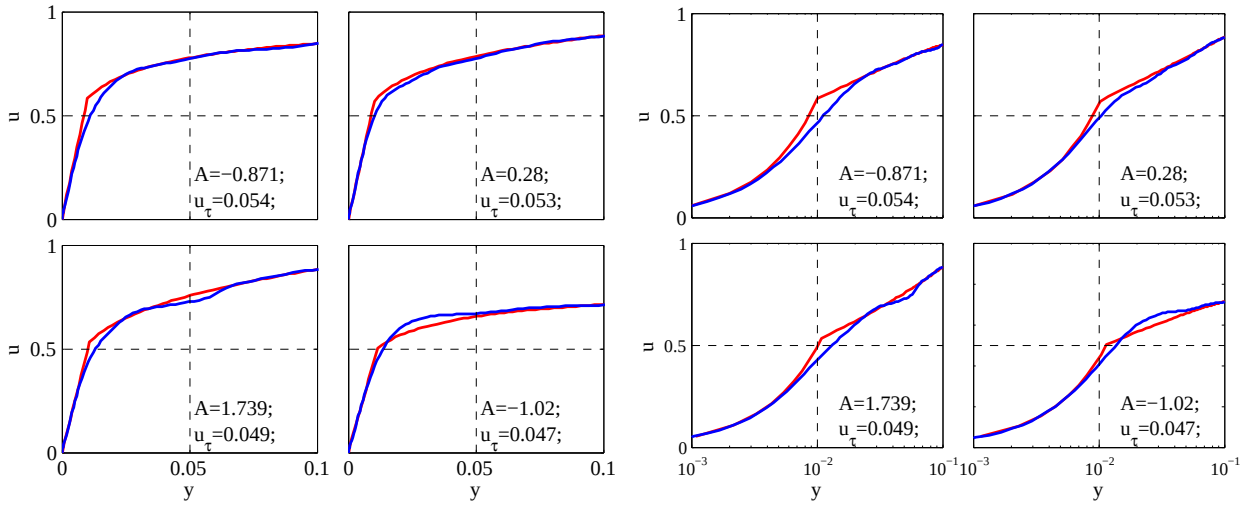


Figure 9: Four sample comparisons of snapshots of the spatially and temporally filtered DNS velocity profiles near the wall and the wall modeled profile in linear-linear scale (left samples) and log-linear scale (right samples). The red lines are modeled profile and the blue lines are filtered profile from DNS. The coefficients A of the linear correction term and the predicted friction velocity are listed within the figure. The units of velocity and length are as provided in the database. Vertical (at $y = 0.01$) and horizontal (at $u = 0.5$) dashed lines are included as visual reference to better see differences between the individual plots.

Present results show that the channel flow database can be used to study wall models for LES, and that the new integral wall model contains non-trivial physics that can lead to more accurate predictions of wall stress evolution in wall-modeled LES.

5 Conclusions

In this paper, a new public database for turbulent channel flow is described. The database facilitates remote public access by using Web-services, including access using familiar programming languages such as Fortran, C or Matlab. A cutout service that enables extracting raw data (velocities and pressure) on grid points is also provided. The new channel flow database is used to gain new insights into wall modeling for LES. Specifically, the data are used to conduct an *a-priori* test of the recently proposed integral wall model for LES. Wall stresses and their rates of change predicted from the integral wall model are compared with the predictions of the traditional equilibrium wall model. It is seen that the iWMLES provides more accurate predictions of the wall stress compared to the equilibrium model. The fact that the data are at high Reynolds number enabled us to probe a significant scale separation, so that the distance to the "first LES grid point" away from the wall Δ_y was both small compared to the channel size ($\Delta_y = 0.1h$) while being far from the viscous region ($\Delta_y = 100\nu/u_\tau$). And, the accessibility of data over an extended time period, as opposed to only a few snapshots, enabled us to perform the time filtering that the iWMLES model calls for.

In practice, the most challenging applications of the iWMLES will not be in LES of equilibrium channel

flow. This flow is not the most demanding test case for a wall model since the overall mean stress will be matched correctly to the imposed pressure gradient forcing. Even so, the *a-priori* tests show that slow temporal fluctuations around the mean value are in fact captured well by the iWMLES. Within these fluctuations, there are sometimes more favorable or adverse pressure gradients, and these modify the local logarithmic law as characterized by the parameter A_x. Reference [57] describes a number of *a-posteriori* tests of the iWMLES approach, such as flow over arrays of wall mounted cubes or truncated cones, showing good results. Further tests in even more demanding flows such as diverging channels are desirable to probe the ability of the model to predict separation.

Acknowledgements:

The work of the group at the Johns Hopkins University was supported by the US NSF grant CDI-II: CMMI 0941530, the database infrastructure was supported by US NSF grant OCI-108849 and JHU's Institute for Data Intensive Engineering & Science. The work on the integral wall model was supported by the Office of Naval Research (grant # N00014-12-1-0582, Dr. R. Joslin, program director). The channel flow simulations on TACC supercomputers were made possible through XSEDE allocation # AST110057. The University of Texas team acknowledges funding from the National Science Foundation PetaApps Grant OCI-0749223 and PRAC Grant 0832634, which supported the development of the PoongBack code. The initial condition field preparation was supported by the Argonne Leadership Computing Facility at Argonne National Laboratory under Early Science Program(ESP) and Innovative and Novel Computational Impact on Theory and Experiment Program(INCITE) 2013. We thank Suzanne Werner, Victor Paul and Jan Vandenberg for their continuing work on the underlying hard and software, Perry Johnson for Matlab coding, and Stephen Hamilton for the website development.

Appendix A: B-Spline collocation method

Basis splines (B-splines) are piecewise polynomials that possess special continuity properties within a given interval (see [67] for an excellent treatise on B-splines.) The construction of B-splines begins with a given, non-decreasing breakpoint distribution on the interval $\Xi = [a, b]$ expressed as $\vec{\xi} = \{\xi_i : i = 0, \ldots, N\}$. From the breakpoint distribution a set of B-splines which guarantee continuous derivatives everywhere in Ξ up to order $k-1$, where k is the order of the B-spline, may be constructed by properly selecting a set of knot locations within Ξ [67, 68]. These knot locations are denoted by $\vec{t} = \{t_i : i = 0, \ldots, N+k-1\}$. The $N+1$ Marsden-Schoenberg collocation points defined on Ξ are then determined as the mean location of $k-1$ consecutive knots in $\vec{t}$ and are expressed by $\vec{x} = \{x_i : i = 0, \ldots, N\}$ [69]. In order to ensure C^{k-1} on Ξ, the number of B-splines required is equal to the knot number [68], i.e. $n = k$ where n are the number of B-splines.

For the given knot vector $\vec{t}$, the n B-splines of order k are then defined according to the recurrence relation as

$$B_i^k(x) = \frac{x - t_i}{t_{i+k-1}} B_i^{k-1}(x) + \frac{t_{i+k} - x}{t_{i+k} - t_{i+1}} B_{i+1}^{k-1}(x) \tag{10}$$

where

$$B_i^1(x) = \begin{cases} 1 & \text{if } t_i \leq x < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

and $i = 0, \ldots, n-1$. The set of B-splines form a complete basis and may be used to express a function $u(x)$ on Ξ as

$$u(x) = \sum_{i=0}^{n-1} c_i B_i^k(x) \tag{12}$$

where c_i are the B-spline coefficients (or de Boor points [69]). For collocation methods, u will be given at

the discrete collocation points. At these locations the field u may be written (in discrete form) as

$$u_j = \sum_{i=0}^{n-1} c_i B_{ij} \tag{13}$$

and $B_{ij} = B_i^k(x_j)$ is the B-spline matrix. The coefficients c_i are then readily expressed as

$$c_i = \sum_{j=0}^{N-1} B_{ij}^{-1} u_j \ . \tag{14}$$

where B_{ij}^{-1} is the inverse of the B-spline matrix from (13).

The derivatives of the field u are carried by the B-splines such that the dth derivative of u may be expressed as

$$u_j^{(d)} = \sum_{i=0}^{N-1} c_i B_{ij}^{(d)} \tag{15}$$

where $B_{ij}^{(d)} = B_i^{(d),k}(x_j)$. Therefore, derivatives are computed by first determining the coefficients c_i given by (14). Then the derivatives are obtained by taking the inner product of the appropriate B-spline matrix with the coefficients.

It should be noted that B-spline matrix is compactly supported over n splines (represented by the inner index i). As a result, the B-spline matrix may be represented as a sparse, banded matrix. This feature is especially important for the linear algebra operations needed for computing the coefficients c_i in (14) and applying the associated B-spline matrix when computing u or its derivatives.

Appendix B: Pressure solver

B.1 Solution procedure

In this Appendix, the solution procedure used in the pressure solver for the channel flow DNS is presented. As mentioned in section 2 the pressure field is not determined during the evaluation of the velocity field. However, it is required for computing certain turbulence quantities of interest (e.g. the pressure strain correlation of the turbulent kinetic energy equation) and thus is desired for storage in the JHTDB. The vertical velocity and vorticity formulation of [59] invokes the PPE given by Eq.(1). From this equation the pressure can be solved for as a boundary-value problem with Neumann boundary conditions.

In the pressure solver, the PPE given by (1) is expressed in wavespace along the x and z directions. An ordinary differential equation (ODE) is then solved for each k_x and k_z wavemode. The solution procedure for the non-zero and zero wavemodes are discussed in the following sections.

We first apply Fourier transforms along the x and z directions to obtain the transformed PPE as

$$\left(-k_m^2 + \frac{d^2}{dy^2}\right)\widehat{p} = ik_x\widehat{H}_x + ik_z\widehat{H}_z + \frac{d\widehat{H}_y}{dy} \tag{16}$$

where $\widehat{\cdot}$ denotes Fourier transform, $\boldsymbol{H} = -\nabla\cdot(\boldsymbol{u}\otimes\boldsymbol{u}) + \boldsymbol{F}$ with $\boldsymbol{F}$ being the mean pressure gradient, i is the imaginary unit, $k_m^2 = k_x^2 + k_z^2$ with k_x and k_z the wave numbers along the x and z directions, respectively. The Fourier transformed Neumann boundary condition is then expressed as

$$\left[\frac{\partial\widehat{p}}{\partial y} - \nu\frac{\partial^2\widehat{v}}{\partial y^2}\right]_{\text{wall}} = 0 \tag{17}$$

For all but the $0-0$ wavemode, these equations provides a second order ODE along y that may be solved using standard techniques. The solver implemented in PoongBack uses a banded LU-decomposition approach for the B-spline linear algebra operations.

Now applying the B-spline representation presented in appendix 5 for p, the transformed PPE in (16) becomes

$$\vec{c}_p \cdot \left(-k_m^2 \boldsymbol{B} + \boldsymbol{B}^{(2)}\right) = \boldsymbol{R} \tag{18}$$

where $\boldsymbol{c}_p$ are the B-spline coefficients for the pressure and $R_j = (ik_x\widehat{H}_x + ik_z\widehat{H}_z + \frac{d\widehat{H}_y}{dy})_j$ where j denotes the index of the collocation points. The above equation may then be expressed as

$$\boldsymbol{c}_p \cdot \boldsymbol{L} = \boldsymbol{R} \tag{19}$$

where $L_{ij} = \left(-k_m^2 B_{ij} + B_{ij}^{(2)}\right)$. Applying the boundary conditions given in (17), the left and right hand side matrices become $L_{ij} = B_{ij}^{(1)}$ and $R_j = \left(\nu \frac{\partial^2 \widehat{v}}{\partial y^2}\right)_j$ become at the top and bottom walls, i.e. $y = \pm h$. The pressure is then computed as

$$\boldsymbol{p} = \boldsymbol{c}_p \cdot \boldsymbol{B} \tag{20}$$

where $\boldsymbol{c}_p = \boldsymbol{L}^{-1} \cdot \boldsymbol{R}$.

The $0-0$ wavemode can be obtained by considering the N-S equation in vertical direction, and for $k_x = k_z = 0$ one obtains

$$\frac{d\widehat{p}}{dy} = \widehat{H}_y. \tag{21}$$

The nonlinear term $\widehat{H}_y$ can be written as $\widehat{H}_y(y;0,0) = -\frac{d\widehat{vv}}{dy}$ such that we can integrate (21) directly to obtain

$$\widehat{p} = -\widehat{vv}. \tag{22}$$

where an arbitrary integration constant has been set to zero.

B.2 Validation

In order to test the numerical implementation of the pressure solver, two validation approaches are taken. In the first approach, two test cases compare numerical results are against the analytical solutions of an inhomogeneous Helmholtz equation. In these tests, the same equations are solved, but Dirichlet boundary conditions are applied in one test and Neumann boundary conditions are applied in the other. The numerical solutions for these tests are obtained using the pressure solution algorithm in appendix B1. Another set of tests are performed which serve to check how well the pressure solver implicitly satisfies the Dirichlet boundary condition while only enforcing the Neumann boundary condition in the PPE. The details of this test are discuss below.

We first solve the inhomogenous Helmholtz equation given as

$$\left(-k^2 + \frac{d^2}{dy^2}\right) p = g(y) \tag{23}$$

where $g(y)$ and $k \neq 0$ are known. The boundary conditions are specified as $p(-1) = c$ and $p(1) = d$. The solution for p may be obtained by writing p as $p = p_h + p_p$ where the homogeneous solution p_h carries the boundary conditions and a particular solution p_p has homogenous boundary conditions. The resulting homogeneous solution is then given as

$$p_h = a\ \cosh(ky) + b\ \sinh(ky) \tag{24}$$

where $a = (c+d)/(2\cosh(k))$ and $b = (d-c)/(2\sinh(k))$. The particular solution may be obtained by using an appropriate basis projection. Since the particular solution has homogeneous boundary conditions a sine series is chosen such that

$$p_p = \sum_{n=1}^{\infty} a_n \sin(\pi n y)\ . \tag{25}$$

The coefficients a_n are then obtained by inserting p_p into (23) and taking the appropriate scalar product to obtain

$$a_n = -\frac{1}{(\pi n)^2 + k^2} \int_{-1}^{1} g(y) \sin(\pi n y) dy \ . \tag{26}$$

The final solution is then given as

$$p(y) = \frac{c+d}{2\ \cosh(k)} \cosh(ky) + \frac{d-c}{2\ \sinh(k)} \sinh(ky) + \sum_{n=1}^{\infty} a_n \sin(\pi n y) \tag{27}$$

where a_n is given by (26).

For the second test, we solve the same inhomogeneous Helmholtz equation but in this case Neumann boundary conditions are applied. The boundary conditions for this case are $\frac{dp}{dy}(-1) = c$ and $\frac{dp}{dy}(1) = d$. Following the same solution procedure as for the Dirichlet boundary condition case, the same particular solution as given by (25) and (26). The homogenous solution is also given by (24), however, the coefficients a and b are modified. The resulting final solution is given as

$$p(y) = \frac{d-c}{2k\ \sinh(k)} \cosh(ky) + \frac{d+c-2\gamma}{2k\ \cosh(k)} \sinh(ky) + \sum_{n=1}^{\infty} a_n \sin(\pi n y) \tag{28}$$

where a_n is given by (26) with

$$\gamma = \sum_{n=1}^{\infty} (-1)^n \frac{\pi n g_n}{(\pi n)^2 + k^2} \tag{29}$$

and $g_n = -\int_{-1}^{1} g(y) \sin(\pi n y) dy$.

The two tests for the different boundary conditions were performed with $g(y) = ky$ and $c = -1$ and $d = 2$. The results for the Dirichlet and Neumann boundary conditions are shown in Figures 10 and 11, respectively, for two values of k. From these plots it is readily observed that the numerical solution computes the correct solution.

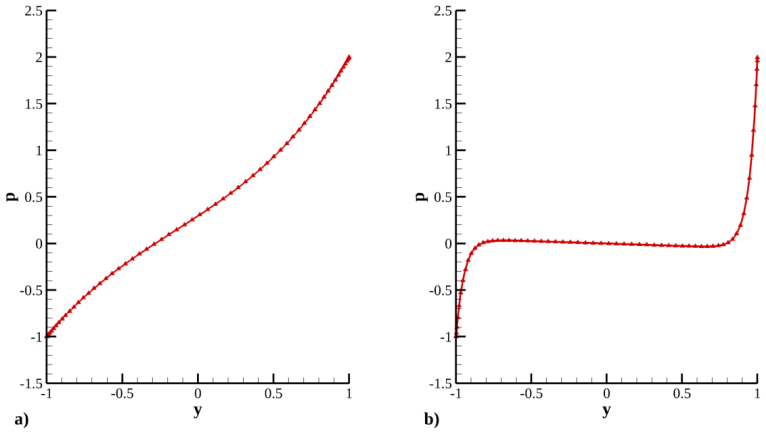


Figure 10: Comparison between the numerical (lines) and analytical solutions (symbols) of (23) with Dirichlet boundary conditions and two values of k: a) $k = 1.1180$ and b) $k = 18.901$.

In the pressure solver, the Neumann boundary condition, as given by (17), is explicitly enforced. There is, however, a Dirichlet boundary condition (obtained by projecting the momentum equation along the channel wall) that may be applied that is just as valid as (17). They both cannot be enforced simultaneously however. For this validation test the Dirichlet boundary condition is computed after the pressure field is obtained. We then perform a posteriori checks to see how well the momentum equation along the channel wall is satisfied.

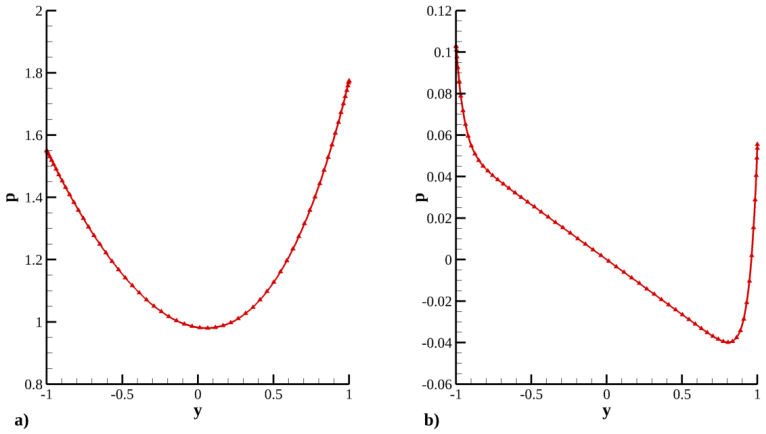


Figure 11: Comparison between the numerical (lines) and analytical solutions (symbols) of (23) with Neumann boundary conditions and two values of k: a) $k = 1.1180$ and b) $k = 18.901$.

For the tests, we use the Dirichlet condition given as:

$$\frac{\partial p}{\partial \tau} = \nu \boldsymbol{\tau} \cdot \nabla^2 \boldsymbol{u} + \boldsymbol{\tau} \cdot \boldsymbol{\Pi} \tag{30}$$

where $\boldsymbol{\Pi}$ is the mean pressure gradient force. We simply choose $\boldsymbol{\tau}$ along the $x - z$ diagonal resulting in

$$\frac{\partial p}{\partial x} + \frac{\partial p}{\partial z} - \Pi_x = \nu \left(\nabla^2 \boldsymbol{u} + \nabla^2 \boldsymbol{w} \right) \tag{31}$$

In wavespace, we write the residual for the Dirichlet condition as

$$\widehat{R_D} = \underbrace{\frac{\partial \widehat{p}}{\partial x} + \frac{\partial \widehat{p}}{\partial z} - \widehat{\Pi}_x}_{\widehat{R_D^L}} \underbrace{- \nu \left(-k_m^2 + \frac{d^2}{dy^2} \right) (\widehat{u} + \widehat{w})}_{\widehat{R_D^R}} \tag{32}$$

This quantity is computed once the pressure field is obtained from the PPE. We then transform $\widehat{R_D^L}$ and $\widehat{R_D^R}$ to physical space and compute to point-wise residuals. The first pointwise residual uses a local normalization given as:

$$R(\boldsymbol{x}, t) = \frac{|R_D^L(\boldsymbol{x}, t) - R_D^R(\boldsymbol{x}, t)|}{2 \max \left(R_D^L(\boldsymbol{x}, t), R_D^R(\boldsymbol{x}, t) \right)} \tag{33}$$

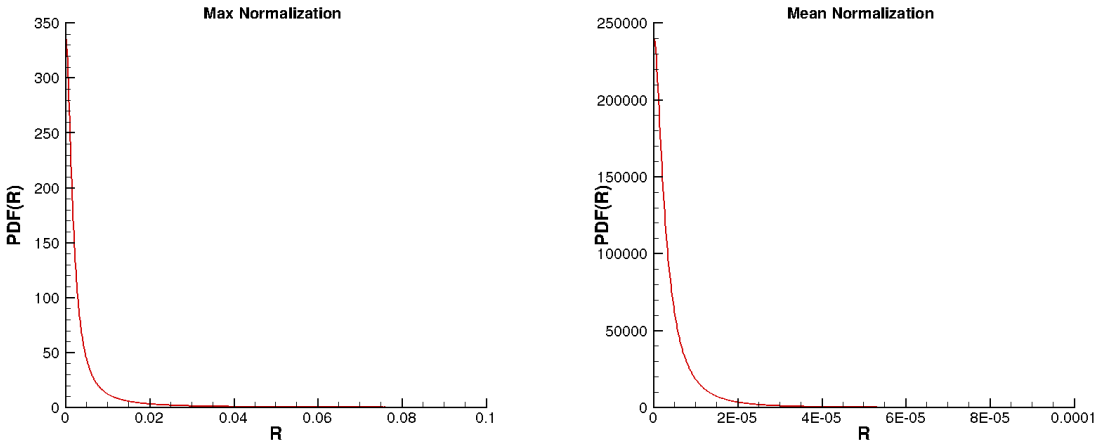
while the second uses the planar average of R_D^L such that:

$$R(\boldsymbol{x}, t) = \frac{|R_D^L(\boldsymbol{x}, t) - R_D^R(\boldsymbol{x}, t)|}{\langle |R_D^L(\boldsymbol{x}, t)| \rangle_{xz}} \tag{34}$$

The PDFs for the residuals given by (33) and (34) are shown. From these results it may be concluded that the Dirichlet boundary condition is reasonably satisfied. This is a result of correct evaluation of the momentum equation by the PoongBack code.

Appendix C: Interpolation and differentiation methods

The JHTDB Web services provide methods for performing spatial interpolation and differentiation within the database. Discussed in the following sections are the formulation of these methods used for the channel flow database. For detailed information about the the subroutine-like calls that can be made from MATLAB, Fortran, C/C++ programs and the appropriate interfaces, see Ref. [6].



(a) Local normalization as defined by (33)

(b) Mean normalization as defined by (34)

Figure 12: PDF of the PPE Dirichlet boundary condition residual with the two normalization types

C.1: Spatial interpolation

Spatial interpolation for domains with non-uninform grid spacing (e.g. the channel flow domain) is applied using multivariate polynomial interpolation of the barycentric Lagrange form from [70]. Using this approach, we are interested in interpolating the field f at point $\boldsymbol{x}'$. The point $\boldsymbol{x}'$ is known to exist within the grid cell at location (x_m, y_n, z_p) where (m, n, p) are the cell indices. The cell indices are obtained for the x and z directions, which are uniformly distributed, according to

$$\begin{aligned} m &= \text{floor}(x'/dx) \\ p &= \text{floor}(z'/dz)\ . \end{aligned} \tag{35}$$

In the y direction the grid is formed by Marsden-Schoenberg collocation points which are not uniformly distributed. Along this direction we perform a search to obtain n such that

$$\begin{aligned} y_n \leq y' < y_{n+1} \quad &\text{if } y' \leq 0 \\ y_{n-1} < y' \leq y_n \quad &\text{if } y' > 0 \end{aligned} \tag{36}$$

The cell indices are also assured to obey the following:

$$\begin{aligned} 0 \leq m \leq N_x - 2 & \\ 0 \leq n \leq N_y/2 - 1 \quad &\text{if } y' \leq 0 \\ N_y/2 \leq n \leq N_y - 1 \quad &\text{if } y' > 0 \\ 0 \leq p \leq N_z - 2 & \end{aligned} \tag{37}$$

where N_x, N_y, and N_z are the number of grid points along the x, y, and z directions, respectively. In the case that $x' = x_{N_x-1}$ the cell index set to be $m = N_x - 2$; likewise for the z direction. The interpolation stencil contains q points in each direction for an order q interpolant (with degree $q-1$). The resulting interpolated value is expressed as:

$$f(\boldsymbol{x}') = \sum_{i=i_s}^{i_e} \sum_{j=j_s}^{j_e} \sum_{k=k_s}^{k_e} l_x^i(x') l_y^j(y') l_z^k(z') f(x_i, y_j, z_k) \tag{38}$$

where the starting and ending indices are given as

$$\begin{aligned} i_s &= m - \text{ceil}(q/2) + 1 \\ i_e &= i_s + q - 1 \\ j_s &= \begin{cases} n - \text{ceil}(q/2) + 1 + j_o & \text{if } n \leq N_y/2 - 1 \\ n - \text{floor}(q/2) + j_o & \text{otherwise} \end{cases} \\ j_e &= j_s + q - 1 \\ k_s &= p - \text{ceil}(q/2) + 1 \\ k_e &= k_s + q - 1 \end{aligned} \tag{39}$$

and j_o is the index offset for the y direction depending on the distance from the top and bottom walls. The ceil() function ensures that stencil remains symmetric about the interpolation point when q is odd. In the case for j_s, the separate treatments for the top and bottom halves of the channel is done to ensure that the one-sided stencils remain symmetric with respect to the channel center. The value for j_o may be evaluated based upon the y cell index and the interpolation order as

$$j_o = \begin{cases} \max(\text{ceil}(q/2) - n - 1, 0) & \text{if } n \leq N_y/2 - 1 \\ \min(N_y - n - \text{ceil}(q/2), 0) & \text{otherwise} \end{cases} . \tag{40}$$

The interpolation weights, l_x, l_y, and l_z, are given as

$$l_\theta^\xi(\theta') = \frac{\frac{w_\xi}{\theta' - \theta_\xi}}{\sum_{\eta=\xi_s}^{\xi_e} \frac{w_\eta}{\theta' - \theta_\eta}} \tag{41}$$

where θ may either be x, y, or z. The barycentric weights, w_ξ, in (41) are given as

$$w_\xi = \frac{1}{\prod_{\eta=\xi_s, \eta\neq\xi}^{\xi_e} \theta_\xi - \theta_\eta} \tag{42}$$

The weights are computed by applying a recursive update procedure as in [70]. A slightly modified version of the algorithm in [70] is given below (Greg Eyink, personal communication, 2013):

for $\xi = \xi_s$ to ξ_e **do**
  $w_\xi = 1$
end for
for $\xi = \xi_s + 1$ to ξ_e **do**
  for $\eta = \xi_s$ to $\xi - 1$ **do**
    $w_\eta = (\theta_\eta - \theta_\xi) w_\eta$
    $w_\xi = (\theta_\xi - \theta_\eta) w_\xi$
  end for
end for
for $\xi = \xi_s$ to ξ_e **do**
  $w_\xi = 1/w_\xi$
end for

To account for the periodic domain along the x and z directions we adjust the i and k indices when referencing f in (38) such that

$$f(\boldsymbol{x}') = \sum_{i=i_s}^{i_e} \sum_{j=j_s}^{j_e} \sum_{k=k_s}^{k_e} l_x^i(x') l_y^j(y') l_z^k(z') f(x_{i\%Nx}, y_j, z_{k\%Nz}) \tag{43}$$

and % is the modulus operator. The indices for the interpolation coefficients remain the same, however, we

use the fact that the grid points are uniformly spaced such that (41) becomes

$$l_\theta^\xi(\theta') = \frac{\frac{w_\xi}{\theta' - \xi\Delta\theta}}{\sum_{\eta=\xi_s}^{\xi_e} \frac{w_\eta}{\theta' - \eta\Delta\theta}} \tag{44}$$

and similarly for the barycentric weights, (42) becomes

$$w_\xi = \frac{1}{\prod_{\eta=\xi_s, \eta\neq\xi}^{\xi_e} (\xi - \eta)\Delta\theta} \tag{45}$$

for the x and z directions. The computation of the barycentric weights for the x and z directions are carried out once (for a given interpolation order) for all grid points using (45); for the y direction the barycentric weights are computed for each point using (42).

C.2: Spatial differentiation

Spatial differentiation for grids with non-uniform spacing is performed using the barycentric method of the interpolating polynomial. In one dimension (assuming the x direction; the same applies for the y and z directions), the interpolant for the field f is given as

$$f(x) = \sum_{j=i_s}^{i_e} l_x^j(x) f(x_j) \ . \tag{46}$$

It follows that the r^{th} derivative may be computed as

$$\frac{d^r f}{dx^r}(x) = \sum_{j=i_s}^{i_e} \frac{d^r l_x^j}{dx^r}(x) f(x_j) \ . \tag{47}$$

Within the database we compute the derivatives at the grid sites for the $FD4NoInt$, $FD6NoInt$, and $FD8NoInt$ differencing methods where no interpolation is performed. If a sample point is given that does not coincide with a grid point, the derivative at the *nearest* grid point is computed and returned. For the $FD4Lag4$ method we compute the derivatives with the $FD4NoInt$ method (at the grid sites) and then these data are interpolated to the interpolation point using the $Lag4$ interpolation method presented in §C.1.

For evaluating derivatives at the grid sites we follow the method presented in [70] such that

$$\frac{d^r f}{dx^r}(x_i) = \sum_{j=i_s}^{i_e} D_{x,ij}^{(r)} f(x_j) \ . \tag{48}$$

where $D_{x,ij}^{(r)} = \frac{d^r l_x^j}{dx^r}(x_i)$ is the differentiation matrix [70]. The differentiation matrices for $r = 1$ and $r = 2$ are given, respectively, as

$$D_{x,ij}^{(1)} = \frac{w_j/w_i}{x_i - x_j} \tag{49}$$

$$D_{x,ij}^{(2)} = -2\frac{w_j/w_i}{x_i - x_j}\left[\sum_{k\neq i} \frac{w_k/w_i}{x_i - x_k} + \frac{1}{x_i - x_j}\right] \tag{50}$$

for $i \neq j$ and

$$D_{x,jj}^{(r)} = -\sum_{i\neq j} D_{x,ji}^{(r)} \tag{51}$$

when $i = j$ for all $r > 0$. We note that in (50) and (51) fixes have been applied to the respective equations presented in [70], i.e., (9.4) and (9.5). As with the interpolation schemes, the grid point locations for the uniformly distributed directions are expressed as $\theta_\xi = \xi\Delta\theta$, where θ may either be x or z.

For second order mixed derivatives (such as for the pressure Hessian) we compute the derivatives at the grid sites within the respective plane. When computing the mixed partials along x and y we have

$$\frac{d^2 f}{dxy}(x_m, y_n) = \sum_{i=i_s}^{i_e} \sum_{j=j_s}^{j_e} D^{(1)}_{x,mi} D^{(1)}_{y,nj} f(x_i, y_j) \ . \tag{52}$$

Similar formulae exist for mixed partials along x and z, and y and z.

The differencing stencil size depends on the required order of the differencing method and the derivative order, r. In general, the resulting stencil size is determined as

$$q = \begin{cases} q' + r & \text{for non–symmetric grid distribution about evaluation point} \\ q' + r - (r+1)\%2 & \text{for symmetric grid distribution about evaluation point} \end{cases} \tag{53}$$

where q' is the order of the differencing method. For example, to obtain a 6^{th} order differencing method for the first derivative of f along the x, y, and z directions, a value of $q = 7$ is required. For the second derivative, the x or z directions require a value of $q = 7$ where the y direction requires $q = 8$ to achieve a 6^{th} order differencing method.

C.3: Spline interpolation

While the Lagrange interpolation techniques provide a fast and accurate result, they have discontinuous derivatives at grid cell faces, which is undesirable in some scenarios, for instance for the integration of particle trajectories [71]. Therefore, high order spline interpolations were additionally implemented, in order to provide interpolants with continuous derivatives up to desired orders. Specifically, the methods discussed in [71, 72] were generalized to the case of nonuniform grids, and were subsequently used. In the following, we provide a brief overview of the method, and a preliminary study of the interpolation errors.

Hermite splines Assuming that the values of a function $f(x)$ and its derivatives are known at the points 0 and 1, the polynomial $s(x)$ can be built, such that

$$s(x) = \sum_{k=0}^{p} a_k x^k \tag{54}$$

$$\left[\frac{d^l s}{dx^l}(x) = f^{(l)}(x) \right]_{x \in \{0,1\}} , \forall l \in \{0, 1, \ldots, m\} \tag{55}$$

where p is the order of the polynomial; if the linear system of equations (55) is to have a unique solution a_k, p must be equal to $2m + 1$.

From this problem a sequence of "base" polynomials can be derived, such that the expression of $s(x)$ can be written as follows:

$$s(x) = \sum_{l=0}^{m} \sum_{i=0,1} f^{(l)}(i) \alpha_i^{(m,l)}(x) \tag{56}$$

where the α polynomials (of order p each one) have the following explicit expressions (see [72] for details):

$$\alpha_0^{(m,l)}(x) = \frac{x^l}{l!}(1-x)^{m+1} \sum_{k=0}^{m-l} \frac{(m+k)!}{m!k!} x^k \tag{57}$$

$$\alpha_1^{(m,l)}(x) = \frac{(x-1)^l}{l!} x^{m+1} \sum_{k=0}^{m-l} \frac{(m+k)!}{m!k!} (1-x)^k \tag{58}$$

Grid splines In [72] centered differences are used to approximate the derivatives of f, since it would be impractical to store the derivatives of f on the grid for the general multidimensional case. A naive algorithm is then provided to construct "grid splines":

$$s^{(m,q)}(x) = \sum_{i=-n}^{1+n} f(i)\beta_i^{(m,q)}(x) \tag{59}$$

where the β polynomials can be constructed from the α polynomials and the centered difference formulae. Also, n is the number of immediate neighbours that are needed for the computation, while $q = 2n + 2$ is the total number of points used (i.e. the stencil size). Unlike in previous work, the pair (m, q) is chosen for the characterization of each formula, since the interest lies mainly in the smoothness m of the interpolant, and the computational cost is mostly related to q. Note that the order $p = 2m + 1$ also has an influence on the computational cost, but this becomes less important in the multidimensional case.

This previous result is perfectly adequate for the case of periodic, uniform grids, since the coordinate transforms are trivial. Furthermore, the β polynomials themselves owe their form to the fact that the original function is known at points in $\mathbb{Z}$. However, these results cannot be directly used for nonuniform grids, or even for uniform grids that are not periodic.

The basis of (59) is that the β polynomials contain information about the centered differences. It is crucial that centered differences are used, since that means that, whether we compute the interpolation in the interval $[i, i + 1]$ or $[i + 1, i + 2]$, the derivatives will be approximated with the same value at the point $i + 1$, therefore the derivative of the interpolant will be continuous. For the case of generic grids, it is this constraint that must be kept: the value of the finite difference approximation used for some grid point, embedded in the form of the polynomials used, must be the same whether we approach the grid point from the left or from the right.

In [73, 74] a systematic way to construct all the possible finite difference formulas for a given grid is provided. This allows the construction of specific β polynomials for individual grid points on a nonuniform, nonperiodic grid. I.e. instead of having a number q of β polynomials for each formula, there are now on the order of Nq, where N is the total number of grid points. Obviously, the generation and storage of these many polynomials is only reasonable in cases when there are many different fields that must be evaluated on the same grid.

Assume that a grid of points $x_i, 0 \leq i \leq N - 1$ is given, as well as a smoothness m and a stencil size q (thus a number of neighbours $n = (q - 2)/2$. This is the outline of the algorithm that is then followed for a generic one dimensional grid to construct the β polynomials:

1. For each $0 \leq i \leq N - 1$, construct the Fornberg coefficients δ_{ij}^l for the l-th derivative approximated at the point x_i, using the grid nodes $\nu_i(j), 0 \leq j \leq q - 1$, where

   (a) if the grid is periodic:
   $\nu_i(0) = i - n$, $\nu_i(1) = i - n + 1$, ..., $\nu_i(q - 2) = i + n$
   Note that since the distances between successive points on the grid is constant, the resulting δ_{ij}^l will in fact be independent of i.

   (b) otherwise:

      i. case $i < n$:
      $$\nu_i(0) = 0, \nu_i(1) = 1, \ldots, \nu_i(q - 2) = q - 2$$

      ii. case $n \leq i < N - n$:
      $$\nu_i(0) = i - n, \nu_i(1) = i - n + 1, \ldots, \nu_i(q - 2) = i + n$$

      iii. case $N - n \leq i$:
      $$\nu_i(0) = N - q + 1, \nu_i(1) = N - q + 2, \ldots, \nu_i(q - 2) = N - 1$$

2. For each $0 \leq i < N-1$, construct the corresponding sequence of β polynomials, and the sequence of compute nodes μ:

   (a) if the grid is periodic:

$$\beta_0(\tilde{x}) = \sum_{l=0}^{m} (x_1 - x_0)^{-l} \delta_{00}^l \alpha_0^{(m,l)}(\tilde{x}) \tag{60}$$

$$\beta_j(\tilde{x}) = \sum_{l=0}^{m} (x_1 - x_0)^{-l} \left[\delta_{0j}^l \alpha_0^{(m,l)}(\tilde{x}) + \delta_{0(j-1)}^l \alpha_1^{(m,l)}(\tilde{x}) \right], \forall 1 \leq j < q-1 \tag{61}$$

$$\beta_{q-1}(\tilde{x}) = \sum_{l=0}^{m} (x_1 - x_0)^{-l} \delta_{0(q-1)}^l \alpha_1^{(m,l)}(\tilde{x}) \tag{62}$$

$$\mu_i(0) = i - n, \mu_i(1) = i - n + 1, \ldots, \mu_i(q-2) = i + n, \mu_i(q-1) = i + 1 + n \tag{63}$$

   (b) otherwise:

      i. case $i < n$

$$\beta_{ij}(\tilde{x}) = \sum_{l=0}^{m} (x_{i+1} - x_i)^{-l} \left[\delta_{ij}^l \alpha_0^{(m,l)}(\tilde{x}) + \delta_{(i+1)j}^l \alpha_1^{(m,l)}(\tilde{x}) \right], \forall 0 \leq j < q-1 \tag{64}$$

$$\mu_i(j) = \nu_i(j) \tag{65}$$

      ii. case $n \leq i < N - n$:

$$\beta_{i0}(\tilde{x}) = \sum_{l=0}^{m} (x_{i+1} - x_i)^{-l} \delta_{i0}^l \alpha_0^{(m,l)}(\tilde{x}) \tag{66}$$

$$\beta_{ij}(\tilde{x}) = \sum_{l=0}^{m} (x_{i+1} - x_i)^{-l} \left[\delta_{ij}^l \alpha_0^{(m,l)}(\tilde{x}) + \delta_{(i+1)(j-1)}^l \alpha_1^{(m,l)}(\tilde{x}) \right], \forall 1 \leq j < q-1 \tag{67}$$

$$\beta_{i(q-1)}(\tilde{x}) = \sum_{l=0}^{m} (x_{i+1} - x_i)^{-l} \delta_{(i+1)(q-1)}^l \alpha_1^{(m,l)}(\tilde{x}) \tag{68}$$

$$\mu_i(0) = i - n, \mu_i(1) = i - n + 1, \ldots, \mu_i(q-2) = i + n, \mu_i(q-1) = i + 1 + n \tag{69}$$

      iii. case $N - n \leq i$:

$$\beta_{ij}(\tilde{x}) = \sum_{l=0}^{m} (x_{i+1} - x_i)^{-l} \left[\delta_{ij}^l \alpha_0^{(m,l)}(\tilde{x}) + \delta_{(i+1)j}^l \alpha_1^{(m,l)}(\tilde{x}) \right], \forall 0 \leq j < q-1 \tag{70}$$

$$\mu_i(j) = \nu_i(j) \tag{71}$$

After the polynomials are generated, given some random point $x_0 \leq x < x_{N-1}$, do the following:

1. find the c (cell) index, such that $x_c \leq x < x_{c+1}$, and compute $\tilde{x} = \frac{x - x_c}{x_{c+1} - x_c}$
2. compute the sum

$$\sum_{j=0}^{q-1} \beta_{cj}(\tilde{x}) f(\mu_c(j)) \tag{72}$$

   where f is the field that is to be interpolated, known at the grid points, and imposing that β_{cj} is 0 if undefined.

Implementation for channel flow database For the channel flow database, the simple uniform and periodic grid results from [71] are used for the x and z directions, and the above generalization for the y

direction. This leads to the following tensor product spline

$$\sum_{i=0}^{q-1}\sum_{j=0}^{q-1}\sum_{k=0}^{q-1}\beta_i(\tilde{x})\beta_{c_yj}(\tilde{y})\beta_k(\tilde{z})f(\mu(i),\mu_c(j),\mu(k)) \tag{73}$$

where β_j are the uniform-periodic grid β polynomials, and likewise μ gives the compute nodes for the uniform-periodic grid.

Derivatives can be interpolated by differentiating the β polynomials directly. This leads to interpolants which coincide with finite difference approximations at grid nodes by construction. In practice, the different derivatives of the β polynomials are also stored alongside the polynomials, and they are simply replaced into (73) as needed:

$$\frac{d^{o_x}}{dx^{o_x}}\frac{d^{o_y}}{dx^{o_y}}\frac{d^{o_z}}{dx^{o_z}}f(x,y,z)\approx\sum_{i=0}^{q-1}\sum_{j=0}^{q-1}\sum_{k=0}^{q-1}\beta_{c_x}^{(o_x)}(\tilde{x})\beta_{c_yj}^{(o_y)}(\tilde{y})\beta_{c_z}^{(o_z)}(\tilde{z})f(\mu(i),\mu_c(j),\mu(k)) \tag{74}$$

The maximum order of differentiation is m, since continuity can only be guaranteed up to order m.

Tests based on divergence-free condition By construction, the different spline interpolations should converge to a set limit as the smoothness m and the stencil size q grow to infinity. However, there is no reason to believe that the limit they reach is the "exact" field, i.e. the spectral representation of the DNS.

As a limited test on the accuracy of derivatives and interpolations provided by the spline methods described above, we evaluate relative errors in complying with the divergence-free condition. A set of points was chosen at random in the lower half of the simulation domain (at a randomly chosen time): 2^{12} points distributed randomly in the xz plane for each 8^{th} y cell. The velocity field was interpolated at these points with different methods, and the results compared, resulting in an "error"; this error was then averaged over the 2^{12} points for each value of y, to obtain the dependency of the error on the y component. In this sense, we can compare all the interpolation methods to one chosen as a reference ($(m,q)=(2,14)$ in our case), as in figure 13; we can study the interpolation error for the different quantities of interest, as in figure 14; and we can study how well interpolation methods respect certain constraints on the data, as in figure 15.

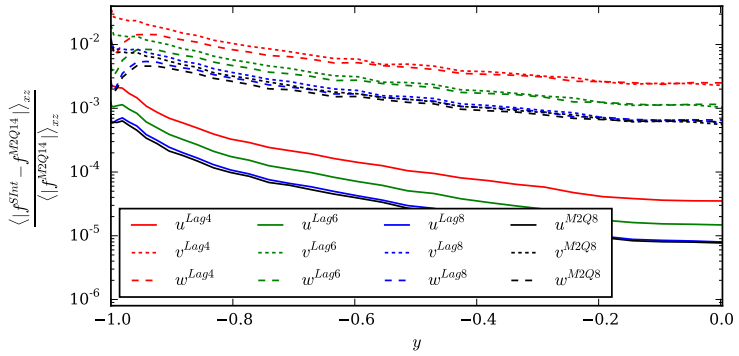


Figure 13: Distance between various methods and the $(m,q)=(2,14)$ method, for the different components of the velocity field.

Taking into account the results seen in figure 15, a convergence study was carried out. Ideally, the divergence should approach 0 when computed for the interpolated velocity field, since the divergence-free property was imposed in Fourier space in the original DNS. A comparison of the divergence error, as a function of the size of the stencils used in the interpolation, was made between the channel flow data for $y=0$ and the isotropic turbulence data (described in [6, 20]), for some arbitrarily chosen fixed y value. Since the channel flow data is close to isotropic at the channel center, this should provide a meaningful comparison.

2^{12} points were chosen with random x and z coordinates, and the divergence was computed for four different datasets: raw channel flow data (labeled with 'c' in the figure), raw isotropic turbulence data

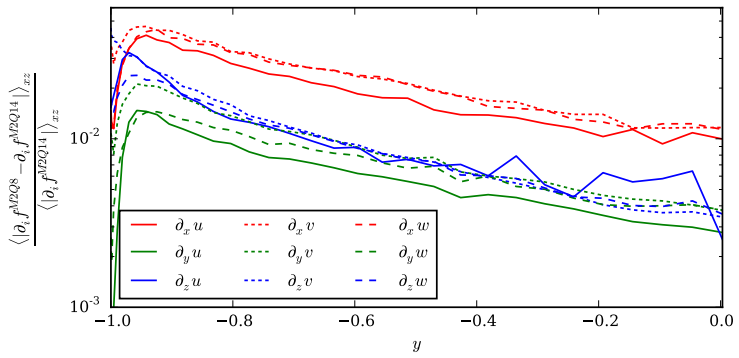


Figure 14: Distance between $(2, 8)$ and $(2, 14)$ for the different components of the velocity gradient.

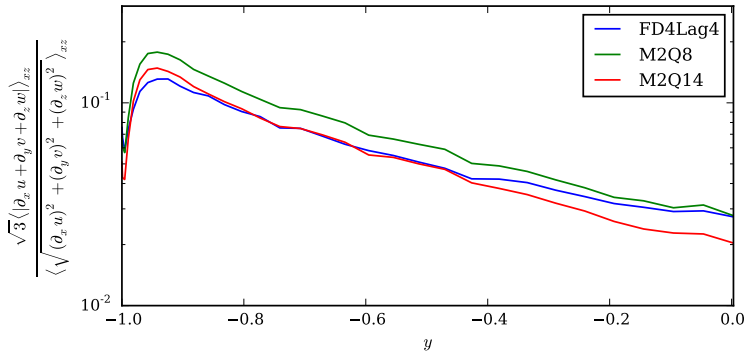


Figure 15: Divergence of the velocity field, computed with the results of different interpolations.

(labeled with 'i' in the figure), filtered channel flow data ('fc') and filtered isotropic turbulence data ('fi'). The points are identical for the raw and filtered versions of the same dataset. Filtering was performed by convolution with the array $(1/4, 1/2, 1/4)$ in the x and z directions, i.e. a trapezoidal rule with width twice the grid-size was applied (no filtering on the y direction for either dataset).

It can be seen that the divergence error does not converge for the raw channel flow dataset, while it does converge (for the range of q values we used) for the isotropic turbulence dataset. This is most likely related to the number of grid points used for the real space grid for the two datasets. In the case of the channel flow, the minimum number of points was chosen, so that all the Fourier modes would be represented (in the x and z directions). Since a 2/3 truncation was used for dealiasing, this means that the space required for the database was decreased by more than half (4/9), as opposed to the case when the real space grid used in the original DNS would have been used. For the isotropic turbulence dataset, the full DNS real space grid was used, even though a $2\sqrt{2}/3$ truncation had also been applied.

Figure 16 seems to imply that it is this additional smoothness that allows the spline interpolations to converge when the stencil size is increased. Interestingly, the $m = 1$ methods do not converge for the channel flow even for the filtered data. However, it is quite clear that not only is the divergence error much smaller for the filtered data, but it also converges (for $m = 2$ and $m = 3$) with increasing q.

References

[1] T. Ishihara, T. Gotoh, and Y. Kaneda, "Study of high-reynolds number isotropic turbulence by direct numerical simulation," *Annual Review of Fluid Mechanics*, vol. 41, pp. 165–180, 2009.

[2] S. Hoyas and J. Jiménez, "Scaling of the velocity fluctuations in turbulent channels up to Reτ= 2003," *Physics of Fluids (1994-present)*, vol. 18, no. 1, p. 011702, 2006.

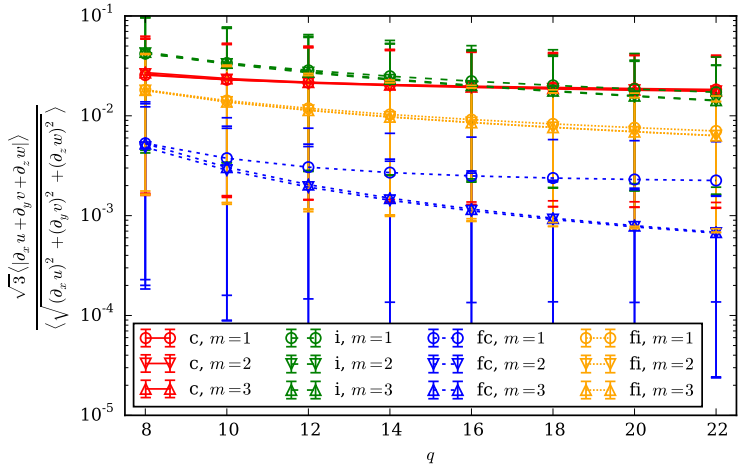


Figure 16: Convergence study for divergence error; average values are shown against different values of the stencil q. Error bars represent intervals containing 80% of the values.

[3] P. K. Yeung, S. B. Pope, and B. L. Sawford, "Reynolds number dependence of lagrangian statistics in large numerical simulations of isotropic turbulence," *Journal of Turbulence*, no. 7, 2006.

[4] J. Schumacher, K. R. Sreenivasan, and V. Yakhot, "Asymptotic exponents from low-reynolds-number flows," *New Journal of Physics*, vol. 9, no. 4, p. 89, 2007.

[5] E. Perlman, R. Burns, Y. Li, and C. Meneveau, "Data exploration of turbulence simulations using a database cluster," in *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, p. 23, ACM, 2007.

[6] Y. Li, E. Perlman, M. Wan, Y. Yang, C. Meneveau, R. Burns, S. Chen, A. Szalay, and G. Eyink, "A public turbulence database cluster and applications to study lagrangian evolution of velocity increments in turbulence," *Journal of Turbulence*, no. 9, 2008.

[7] F. Toschi, "ICFD database 2." `http://mp0806.cineca.it/icfd.php`, 2014. examined October 2014.

[8] J. C. Del Álamo and J. Jiménez, "Spectra of the very large anisotropic scales in turbulent channels," *Physics of Fluids*, vol. 15, no. 6, p. L41, 2003.

[9] R. D. Moser, "ICES database." `http://turbulence.ices.utexas.edu`, 2014. examined October 2014.

[10] M. K. Lee and R. D. Moser, "Direct numerical simulation of turbulent channel flow up to re_tau = 5200," *Journal of Fluid Mechanics*, 2014.

[11] J. C. del Alamo and J. Jimenez, "Direct numerical simulation of the very large anisotropic scales in a turbulent channel," *arXiv preprint arXiv:1309.2322*, 2013.

[12] J. Jimenez, "UPM database." `http://torroja.dmt.upm.es/channels/data/`, 2014. examined October 2014.

[13] Langley Research Center, "Turbulence Modeling Resource." `http://turbmodels.larc.nasa.gov/`, 2015. examined April 2015.

[14] ERCOFTAC, "Turbulence modeling database." `http://www.ercoftac.org/fileadmin/user_upload/bigfiles/sig15/database/index.html`, 2015. examined April 2015.

[15] University of Tokyo, "DNS and heat transfer statistics database." `http://thtlab.jp/DNS/dns_database.html`, 2015. examined April 2015.

[16] S. Pirozzoli, M. Bernardini, and P. Orlandi, "Turbulent channel flow statistics DNS database." `http://newton.dma.uniroma1.it/channel`, 2015. examined April 2015.

[17] K. Kanov, R. Burns, G. Eyink, C. Meneveau, and A. Szalay, "Data-intensive spatial filtering in large numerical simulation datasets," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, p. 60, IEEE Computer Society Press, 2012.

[18] K. Kanov, E. Perlman, R. Burns, Y. Ahmad, and A. Szalay, "I/o streaming evaluation of batch queries for data-intensive computational turbulence," in *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, p. 29, ACM, 2011.

[19] X. Wang, E. Perlman, R. Burns, T. Malik, T. Budavári, C. Meneveau, and A. Szalay, "Jaws: Job-aware workload scheduling for the exploration of turbulence simulations," in *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–11, IEEE Computer Society, 2010.

[20] H. Yu, K. Kanov, E. Perlman, J. Graham, E. Frederix, R. Burns, A. Szalay, G. Eyink, and C. Meneveau, "Studying lagrangian dynamics of turbulence using on-demand fluid particle tracking in a public turbulence database," *Journal of Turbulence*, no. 13, 2012.

[21] G. Eyink, E. Vishniac, C. Lalescu, H. Aluie, K. Kanov, K. Bürger, R. Burns, C. Meneveau, and A. Szalay, "Flux-freezing breakdown in high-conductivity magnetohydrodynamic turbulence," *Nature*, vol. 497, no. 7450, pp. 466–469, 2013.

[22] H. Xu, A. Pumir, G. Falkovich, E. Bodenschatz, M. Shats, H. Xia, N. Francois, and G. Boffetta, "Flight–crash events in turbulence," *Proceedings of the National Academy of Sciences*, vol. 111, no. 21, pp. 7558–7563, 2014.

[23] J. Jucha, H. Xu, A. Pumir, and E. Bodenschatz, "Time-reversal-symmetry breaking in turbulence," *Phys. Rev. Lett.*, vol. 113, p. 054501, 2014.

[24] K. Gustavsson, J. Einarsson, and B. Mehlig, "Tumbling of small axisymmetric particles in random and turbulent flows," *Physical Review Letters*, vol. 112, no. 1, p. 014501, 2014.

[25] G. L. Eyink, "Stochastic flux freezing and magnetic dynamo," *Physical Review E*, vol. 83, no. 5, p. 056405, 2011.

[26] G. L. Eyink and D. Benveniste, "Diffusion approximation in turbulent two-particle dispersion," *Physical Review E*, vol. 88, no. 4, p. 041001, 2013.

[27] D. Benveniste and T. Drivas, "Asymptotic results for backwards two-particle dispersion in a turbulent flow," *Physical Review E*, vol. 89, no. 4, p. 041003, 2014.

[28] N. A. Buchmann, C. E. Willert, and J. Soria, "Pulsed, high-power LED illumination for tomographic particle image velocimetry," *Experiments in Fluids*, vol. 53, no. 5, pp. 1545–1560, 2012.

[29] X. Liu and J. Katz, "Vortex-corner interactions in a cavity shear layer elucidated by time-resolved measurements of the pressure field," *Journal of Fluid Mechanics*, vol. 728, pp. 417–457, 2013.

[30] D. Xu and J. Chen, "Accurate estimate of turbulent dissipation rate using PIV data," *Experimental Thermal and Fluid Science*, vol. 44, pp. 662–672, 2013.

[31] D. Fiscaletti, J. Westerweel, and G. E. Elsinga, "Long-range μPIV to resolve the small scales in a jet at high Reynolds number," *Experiments in Fluids*, vol. 55, no. 9, pp. 1–15, 2014.

[32] J. M. Lawson and J. R. Dawson, "A scanning piv method for fine-scale turbulence measurements," *Experiments in Fluids*, vol. 55, no. 12, pp. 1–19, 2014.

[33] B. Luethi, M. Holzner, and A. Tsinober, "Expanding the Q–R space to three dimensions," *Journal of Fluid Mechanics*, vol. 641, pp. 497–507, 2009.

[34] J. I. Cardesa, D. Mistry, L. Gan, and J. R. Dawson, "Invariants of the reduced velocity gradient tensor in turbulent flows," *Journal of Fluid Mechanics*, vol. 716, pp. 597–615, 2013.

[35] H. Yu and C. Meneveau, "Lagrangian refined kolmogorov similarity hypothesis for gradient time evolution and correlation in turbulent flows," *Physical review letters*, vol. 104, no. 8, p. 084502, 2010.

[36] C. Meneveau, "Lagrangian dynamics and models of the velocity gradient tensor in turbulent flows," *Annu. Rev. Fluid Mech.*, vol. 43, pp. 219—245, 2011.

[37] H. Yu and C. Meneveau, "Scaling of conditional Lagrangian time correlation functions of velocity and pressure gradient magnitudes in isotropic turbulence," *Flow, Turbulence and Combustion*, vol. 85, no. 3-4, pp. 457–472, 2010.

[38] Y. Li, L. Chevillard, C. Meneveau, and G. L. Eyink, "Matrix exponential-based closures for the turbulent subgrid-scale stress tenso," *Phys. Rev. E*, vol. 79, p. 016305, 2009.

[39] H. Yu, K. Kanov, E. Perlman, J. Graham, E. Frederix, R. Burns, A. Szalay, G. Eyink, and C. Meneveau, "Studying lagrangian dynamics of turbulence using on-demand fluid particle tracking in a public turbulence database," *Journal of Turbulence*, vol. 13, no. 12, 2012.

[40] L. Moriconi and R. M. Pereira, "Vorticity statistics and the time scales of turbulent strain," *Physical Review E*, vol. 88, no. 1, p. 013005, 2013.

[41] H. Lu, "Assessment of the modulated gradient model in decaying isotropic turbulence," *Theoretical and Applied Mechanics Letters*, vol. 1, no. 4, p. 041004, 2011.

[42] A. G. Gungor and S. Menon, "A new two-scale model for large eddy simulation of wall-bounded flows," *Progress in Aerospace Sciences*, vol. 46, no. 1, pp. 28–45, 2010.

[43] F. F. Grinstein, A. A. Gowardhan, and A. J. Wachtor, "Simulations of Richtmyer–Meshkov instabilities in planar shock-tube experiments," *Physics of Fluids*, vol. 23, no. 3, p. 034106, 2011.

[44] W. Liu and E. Ribeiro, "Scale and rotation invariant detection of singular patterns in vector flow fields," in *Structural, Syntactic, and Statistical Pattern Recognition*, pp. 522–531, Springer, 2010.

[45] P. Bhat and K. Subramanian, "Fluctuation dynamos and their Faraday rotation signatures," *Monthly Notices of the Royal Astronomical Society*, vol. 429, no. 3, pp. 2469–2481, 2013.

[46] C. J. Keylock, T. E. Tokyay, and G. Constantinescu, "A method for characterising the sensitivity of turbulent flow fields to the structure of inlet turbulence," *Journal of Turbulence*, vol. 12, no. 45, 2011.

[47] M. Holzner, M. Guala, B. Lüthi, A. Liberzon, N. Nikitin, W. Kinzelbach, and A. Tsinober, "Viscous tilting and production of vorticity in homogeneous turbulence," *Physics of Fluids*, vol. 22, no. 6, p. 061701, 2010.

[48] C. C. Wu and T. Chang, "Rank-ordered multifractal analysis (roma) of probability distributions in fluid turbulence," *Nonlinear Processes in Geophysics*, vol. 18, no. 2, pp. 261–268, 2011.

[49] C. J. Keylock, K. Nishimura, and J. Peinke, "A classification scheme for turbulence based on the velocity-intermittency structure with an application to near-wall flow and with implications for bed load transport," *Journal of Geophysical Research: Earth Surface (2003–2012)*, vol. 117, no. F1, 2012.

[50] W. Liu and E. Ribeiro, "Detecting singular patterns in 2D vector fields using weighted Laurent polynomial," *Pattern Recognition*, vol. 45, no. 11, pp. 3912–3925, 2012.

[51] M. Mishra, X. Liu, M. Skote, and C.-W. Fu, "Kolmogorov spectrum consistent optimization for multi-scale flow decomposition," *Physics of Fluids*, vol. 26, no. 5, p. 055106, 2014.

[52] F. F. Grinstein, A. A. Gowardhan, J. R. Ristorcelli, and A. J. Wachtor, "On coarse-grained simulations of turbulent material mixing," *Physica Scripta*, vol. 86, no. 5, p. 058203, 2012.

[53] T. Chang, C. C. Wu, M. Echim, H. Lamy, M. Vogelsberger, L. Hernquist, and D. Sijacki, "Complexity phenomena and ROMA of the earth's magnetospheric cusp, hydrodynamic turbulence, and the cosmic web," *Pure and Applied Geophysics*, pp. 1–19, 2014.

[54] A. Pumir, H. Xu, G. Boffetta, G. Falkovich, and E. Bodenschatz, "Redistribution of kinetic energy in turbulent flows," *Physical Review X*, vol. 4, no. 4, p. 041006, 2014.

[55] M. Treib, K. Burger, F. Reichl, C. Meneveau, A. Szalay, and R. Westermann, "Turbulence visualization at the Terascale on desktop PCs," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 12, pp. 2169–2177, 2012.

[56] "Datascope." http://idies.jhu.edu/datascope.

[57] X. I. A. Yang, J. Sadique, R. Mittal, and C. Meneveau, "Integral wall model for large eddy simulations of wall-bounded turbulent flows," *Phys. Fluids*, 2015. in press.

[58] M. Lee, N. Malaya, and R. D. Moser, "Petascale direct numerical simulation of turbulent channel flow on up to 786K cores," in *Proc. Int. Conf. High Perform. Comput. Networking, Storage Anal. - SC '13*, (New York, New York, USA), pp. 1–11, ACM Press, 2013.

[59] J. Kim, P. Moin, and R. D. Moser, "Turbulence statistics in fully developed channel flow at low Reynolds number," *J. Fluid Mech.*, vol. 177, pp. 133–166, Apr. 1987.

[60] S. A. Orszag, "On the elimination of aliasing in finite-difference schemes by filtering high-wavenumber components," *J. Atmos. Sci.*, vol. 28, pp. 1074–1074, Sept. 1971.

[61] S. Hoyas and J. Jiménez, "Reynolds number effects on the reynolds-stress budgets in turbulent channels," *Physics of Fluids (1994-present)*, vol. 20, no. 10, p. 101511, 2008.

[62] T. Oliver, N. Malaya, R. Ulerich, and R. D. Moser, "Estimating uncertainties in statistics computed from direct numerical simulation," *Physics of Fluids (1994-present)*, vol. 26, no. 3, p. 035101, 2014.

[63] U. Piomelli and E. Balaras, "Wall-layer models for large-eddy simulations," *Annual review of fluid mechanics*, vol. 34, no. 1, pp. 349–374, 2002.

[64] U. Piomelli, "Wall-layer models for large-eddy simulations," *Progress in aerospace sciences*, vol. 44, no. 6, pp. 437–446, 2008.

[65] R. L. Panton, *Incompressible flow*. John Wiley & Sons, 2013.

[66] D. Chung and D. Pullin, "Large-eddy simulation and wall modelling of turbulent channel flow," *Journal of Fluid Mechanics*, vol. 631, pp. 281–309, 2009.

[67] C. de Boor, *A Practical Guide to Splines*. Springer, 2001.

[68] O. Botella, "A velocity-pressure Navier-Stokes solver using a B-spline collocation method," *CTR Annu. Res. Briefs*, 1999.

[69] R. W. Johnson, "Higher order B-spline collocation at the Greville abscissae," *Appl. Numer. Math.*, vol. 52, pp. 63–75, Jan. 2005.

[70] J. P. Berrut and L. N. Trefethen, "Barycentric lagrange interpolation," *SIAM Rev.*, vol. 46, pp. 501–517, jan 2004.

[71] C. C. Lalescu, B. Teaca, and D. Carati, "Implementation of high order spline interpolations for tracking test particles in discretized fields," *Journal of Computational Physics*, vol. 229, no. 17, pp. 5862 – 5869, 2010.

[72] C. C. Lalescu, "Two hierarchies of spline interpolations. practical algorithms for multivariate higher order splines," *ArXiv e-prints*, 2009.

[73] B. Fornberg, "Generation of finite difference formulas on arbitrarily spaced grids," *Mathematics of Computation*, vol. 51, pp. 699–706, Oct. 1988.

[74] B. Fornberg, "Classroom note: Calculation of weights in finite difference formulas," *SIAM review*, vol. 40, no. 3, pp. 685–691, 1998.