Journal of Physics: Conference Series **506** (2014) 012003 doi:10.1088/1742-6596/506/1/012003

Numerical issues in Lagrangian tracking and topological evolution of fluid particles in wall-bounded turbulent flows

C Atkinson[1], J Hackl[2], P Stegeman[1], G Borrell[2] and J Soria[1,3]

[1] Laboratory for Turbulence Research in Aerospace and Combustion, Dept. Mechanical and Aerospace Engineering, Monash University, Victoria, 3800, Australia
[2] School of Aeronautics, Universidad Politécnica de Madrid, 28040 Madrid, Spain
[3] Department of Aeronautical Engineering, King Abdulaziz University, Jeddah, Kingdom of Saudi Arabia

E-mail: `callum.atkinson@monash.edu`

Abstract. The determination of the local Lagrangian evolution of the flow topology in wall-bounded turbulence, and of the Lagrangian evolution associated with entrainment across the turbulent / non-turbulent interface into a turbulent boundary layer, require accurate tracking of a fluid particle and its local velocity gradients. This paper addresses the implementation of fluid-particle tracking in both a turbulent boundary layer direct numerical simulation and in a fully developed channel flow simulation. Determination of the sub-grid particle velocity is performed using both cubic B-spline, four-point Hermite spline and higher-order Hermite spline interpolation. Both wall-bounded flows show similar oscillations in the Lagrangian tracers of both velocity and velocity gradients, corresponding to the movement of particles across the boundaries of computational cells. While these oscillation in the particle velocity are relatively small and have negligible effect on the particle trajectories for time-steps of the order of CFL = 0.1, they appear to be the cause of significant oscillations in the evolution of the invariants of the velocity gradient tensor.

1. Introduction

The study of the temporal evolution of wall-bounded flows is complicated by the range of motion experienced by the near-wall and outer flow, and the dependence of many fluid properties on the relative motion of the observer. In order to overcome this challenge many studies have focused on Galilean invariant quantities such as the velocity gradient tensor A_{ij} (VGT) which governs the formation of intense vortical filaments and whose invariants (Q_A, R_A) can be used to classify the local topology at any point in the flow, within the framework of critical-point theory [1], as indicated in Figure 1. Following this methodology a number of researchers have calculated spiralling conditional mean trajectories (CMT) in Q_A-R_A space similar to that shown in Figure 2, with associated time-scales, which represent the non-local mean topological evolution [2]. Typically these calculations involve the solution of the Lagrangian evolution equations for the Q_A, R_A invariants using the right-hand side of Navier–Stokes equations [3] in an Eulerian frame of reference, evaluated from direct numerical simulations (DNS) [2, 4, 5]. Alternatively, experimental measurements of the Eulerian evolution of the invariants can be used, and the convective terms estimated [6]. This enables an investigation of the mean evolution of the

invariants, conditioned by their location in the Q_A-R_A space, for a given fluid domain, but does not allow the investigation of individual fluid particle traces, or account for the movement of particles throughout the domain. Given that the evolution of the invariants and their associated time-scales have been shown to be strongly dependent on the wall-normal position and, in the outer region of a turbulent boundary layer at $\mathrm{Re}_\tau \approx 1000$, can have evolution times of the order of 2.5 eddy-turnover times and a convected distance of approximately 20 boundary layer thickness (δ) [5], it remains to be seen how representative these CMT are of the true mean Lagrangian evolution of the flow, and the extent to which individual fluid particles follow this mean.

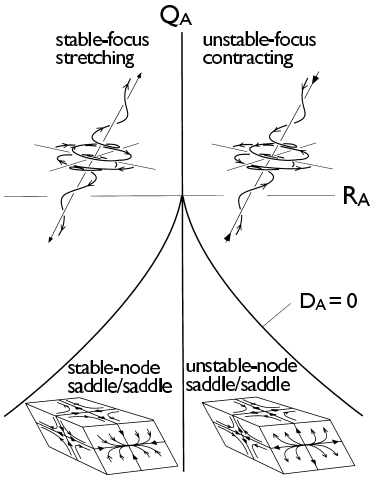


Figure 1. Three-dimensional local topologies (R_A, Q_A)-plane for incompressible flow. Taken from Soria et al. [7]

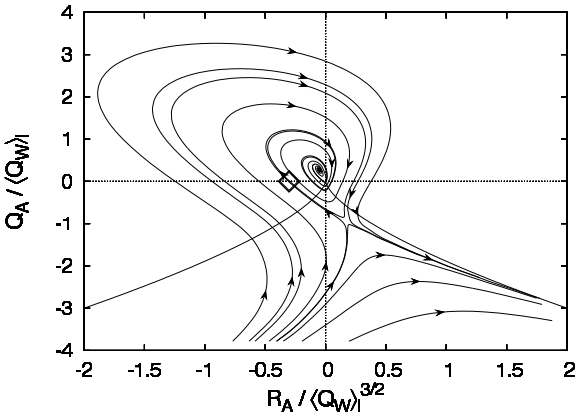


Figure 2. Conditional mean trajectories in the (R_A, Q_A)-plane over a domain corresponding to the most probable gradients in the logarithmic layer and wake region of a turbulent boundary layer from $Re_\theta = 730$ to 1954. Taken from Atkinson et al. 2012 [5].

The Lagrangian evolution of individual fluid particles is also of interest when it comes to the study of the entrainment of irrotational fluid into a turbulent flow, such as the interface between turbulent and non-turbulent (T/NT) regions of a turbulent boundary layer, which have been shown to extend far into the core of the boundary layer [8]. The dynamics of this interface play an important role in the entrainment of mass, momentum and passive scalars into a turbulent flow and in the growth of the boundary layer. Advances in experimental measurement techniques and access to the high-resolution data of direct numerical simulations has lead to an increasing interest in the geometry, role of coherent structures and mechanisms associated with T/NT interfaces in both free and wall-bounded shear flows [9–13]. To date, most of the research regarding the T/NT interface has focused on the investigation and the classification of the interface and the surrounding flow, with less attention given to the Lagrangian view of the entrainment process. For instance, what happens to fluid particles in an irrotational region? how do they interact with the rotational flow and gain enstrophy? what is the topological evolution and time-scale associated with this process?

In order to answer these questions about the true topological evolution experienced by fluid particles in various regions of a turbulent wall-bounded flow, or the evolution and dynamics of a fluid particle as it is entrained across the T/NT interface, it is necessary to implement fluid-particle tracking and thereby determine the Lagrangian evolution of the VGT as it follows a particle at each time step of a DNS of the wall-bounded flow. Fluid-particle tracking has received significant attention in DNS of homogeneous isotropic turbulence [14, 15] but, with the exception of [16], presumably owing to the higher computational demands, it has received relatively little

attention in DNS of wall-bounded flows. In this paper we investigate the implementation of particle tracking in both a turbulent boundary layer and a fully developed channel flow, and examine the influence of different interpolation and particle time-marching approaches on the calculated particle paths, and the Lagrangian evolution of the VGT.

2. Particle Tracking in Direct Numerical Simulations of Wall-bounded flows

Codes that perform direct numeric simulations of fluids flow can be modified to enable the calculation of the Lagrangian evolution of the fluid by treating element or packets of the fluid as if they were particles that exactly follow the fluid dynamics, so that

$$\mathrm{d}\vec{x}/\mathrm{d}t = \vec{u}(\vec{x}), \tag{1}$$

where $\vec{x}$ is the particle's position and $\vec{u}$ is the fluid flow velocity [14]. Given the simplicity of the above equation there should be no need to modify the time-marching of the principle code, assuming that it resolves all the relevant time-scales of the fluid flow. However, if during the time-marching the code uses an implicit or predictor step, a decision must be made as to whether or not to increment particle position at each intermediate step, assuming continuity is enforced at each step, and then re-calculate the particle velocity, or to only update the particle position at the end of the time-step. This later option only involves the use of the higher-order-accurate velocity fields, but will likely require a small overall time-step and more computation in order to achieve an accurate particle trajectory. Ultimately the accuracy of the fluid particle dynamics and the calculated particle trajectory will depend on both the magnitude of the time-step and the accuracy with which the velocity $\vec{u}(\vec{x})$ at the particle location can be determined from the computational cells of the fluid simulation. Assuming that the velocity components at the faces of each cell are calculated to a sufficient accuracy to capture the dynamics of the fluid, the accuracy of the particle velocity will still depend on the numerical interpolation technique used to compute the sub-cell velocity components. Owing to the expense of direct three-dimensional interpolation, such interpolation is typically determined via successive one-dimensional interpolations of the required grid points along each axis, which can also be advantageous when it comes to the interpolation of particle velocity from a computational domain that is split across a series of processors nodes.

Optimal implementation of particle tracking depends on the DNS code in which the tracking is to be performed. In the present case, fluid-particle tracking is to be implemented both in the high-resolution incompressible boundary layer code, and in the fully developed channel code developed by the School of Aeronautics at the Universidad Politécnica de Madrid. Simulation of the turbulent boundary layer is performed using a fractional-step method with a three-step Runge–Kutta scheme for time-marching that is second-order accurate for velocity, and enables the use of variable pressure gradients and time-dependent boundary conditions [17]. Convective and viscous terms in the streamwise (x) and wall-normal (y) directions are calculated using staggered three-point compact finite differences. The computational domain is periodic in the span-wise (z) direction with velocity and pressure components expanded in a Fourier series so that the span-wise gradient can be calculated using spectral methods with the 2/3 rule to prevent aliasing. A pressure correction is applied using the Poisson equation at each sub-step of the Runge–Kutta. Velocity gradients and divergence for the Poisson correction are calculated by centred second-order finite differences in the x, y directions and spectral in the z direction. The present version of the code uses a combination of multi-processing (OMP) and message passing (MPI) for hybrid multi-thread and multi-core parallel processing [18] with the decomposition of the computational domain alternating between cross-stream y, z slabs and stream-wise pencils. Tracking is carried out in the slab decomposition, so that communication is only required in the streamwise direction.

The channel-flow simulations are performed using the established pseudospectral method of Kim *et al* [19] with particle tracking incorporated into the parallel code of Hoyas & Jiménez [20]. Homogeneity in both the stream-wise and span-wise directions allows the pseudospectral solution of the Navier-Stokes equations rewritten in terms of wall-normal vorticity ω_y and the Laplacian of v, hence eliminating the need for pressure. While computationally efficient in terms of the fluid phase, this formulation requires the generation of a realisable velocity field $u_i(x, y, z)$ in order to interpolate for the velocity at each particle. As in the case of the boundary layer, particle positions and velocities are updated at each substage of a semi-implicit third-order Runge–Kutta time marching scheme, hence requiring the generation of the fluid velocity field at each sub-step. In this formulation, derivatives in both x and z directions are handled spectrally, while wall-normal derivatives are computed via a fourth-order seven-point compact finite difference.

3. Sub-grid Interpolation methods

For structured grids, a variety of techniques can be used for sub-cell interpolation. If the parent simulation code involves periodic dimensions in an incompressible flow field, a spectral interpolation can be performed with no interpolation error for all scales resolved by the grid. For non-periodic dimensions compact (spectral-like) interpolation schemes such as those found in [21] and [22] may be used. It is important to note that, while providing excellent interpolation accuracy, both the spectral and compact methods involve significant computational cost that must be repeated for every particle in the domain at each time-step or Runge–Kutta sub-step. Previous applications of fluid particle tracking in DNS of turbulent flows have typically used more computationally efficient and scalable explicit Basis- or Hermite-spline based interpolation schemes [14, 16] with Choi et al. [16] demonstrating that a four-point Hermite interpolation (degree = 8) provides results similar to the far more expensive spectral interpolation, albeit with derivatives at each grid point evaluated spectrally. Recent work of Stegeman et al. [23] indicates that even higher-order Hermite interpolation is required if the interpolation error is to be limited to only a few percent at the smallest wavenumber of the computation. Only the use of Basis and Hermite splines will be consider in this paper.

While both Basis- and Hermite-spline interpolation involves fitting and evaluating a local piecewise polynomial, Hermite splines require the value of the function (in this case velocity) and its derivatives at the two grid points nearest to the target location of the interpolation. In contrast Basis or B-splines represent a recursion of only the target quantity at multiple grid points.

Consider a scalar function (e.g. a single component of the fluid velocity) $f(x)$ sampled with N points over a uniform grid $\{\{x_i\} : i \in [0, N)\}$ with spacing Δ such that $f_i = f(x_i)$. A Hermite spline interpolation operator of degree M is defined as

$$f_I(\check{x}) = \sum_{p=0}^{M-1} a_p \check{x}^p, \tag{2}$$

where the parametric position $\check{x}$ between two consecutive nodes is defined from the spatial position x as

$$\check{x} = \frac{x - x_n}{\Delta} \quad \text{where} \quad x_n \leq x < x_{n+1}. \tag{3}$$

The polynomial coefficients a_p are determined from the boundary conditions of the scalar and its derivatives at $\check{x} = 0$ and 1:

$$\frac{\partial^\alpha f_I(\check{x}_b)}{\partial \check{x}^\alpha} = \frac{\partial^\alpha f(x_n + \Delta \check{x}_b)}{\partial \check{x}^\alpha} \quad \text{where} \quad \check{x}_b = \{0, 1\}, \; \alpha = \{0, ..., K-1\}. \tag{4}$$

This ensures that the derivatives of the fitted spline correspond to the derivative of $f(x)$ and are continuous across grid points up to an order $K = M/2$. The boundary derivatives of the sampled function are determined by a central differencing method with an order K such that the spline coefficients a_p are a linear function of the set of sampled points $\left\{f_{i-K/2}, ..., f_{i+K/2+1}\right\}$,

$$a_p = \sum_{q=0}^{K+1} c_{p,q} f_{n+q-K/2}. \tag{5}$$

Substituting the above expression for the spline coefficients into Equation (2) the spline can be presented as

$$f_I(\check{x}) = \sum_{p=0}^{M-1} \left(\sum_{q=0}^{K+1} c_{p,q} f_{n+q-K/2} \right) \check{x}^p, \tag{6}$$

so that each spline of degree M has an array of $(M-1) \times (K+1) = (M^2 + M - 2)/2$ unique coefficients $c_{p,q}$. Interpolation of a three-dimensional function can then be performed via successive one-dimensional interpolations of the required grid points along each axis. Given that the spline represents a piecewise polynomial, it can also be used to determine the gradient of a scalar at a given interpolated location, albeit at lower order of accuracy. Both this method and the interpolation of the velocity gradient from scalar fields of each component of $\partial u_i/\partial x_j$ are examined in section 5.

From the above definition, a cubic Hermite spline consists of order M=4, and requires the calculation to second-order accuracy of derivatives up to first order at the nearest grid points. This can be obtained from a second-order central difference. Following the formulation of Equation (6) a cubic B-spline will have pre-determined coefficients

$$c_{p,q} = \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}.$$

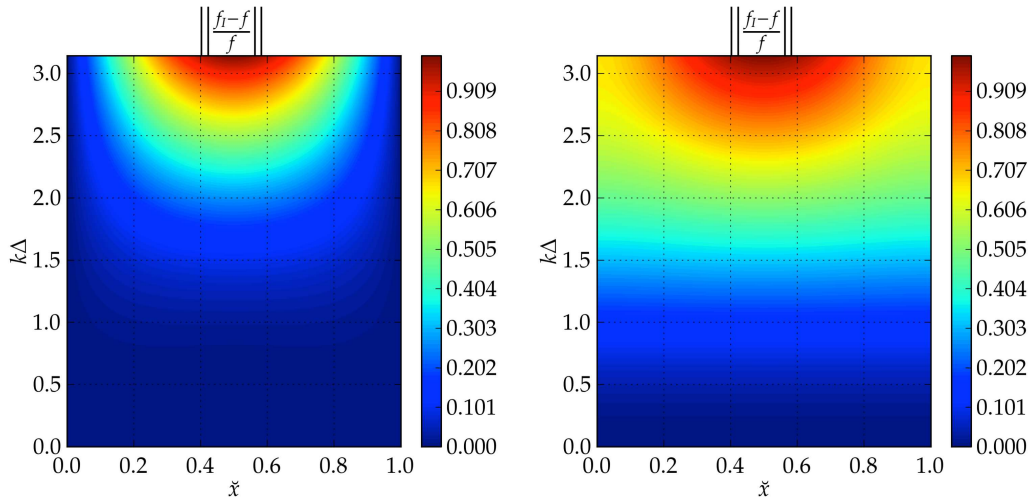
An alternative form to this Hermite interpolation was considered in the turbulent channel flow, where rather than using higher-order derivative boundary conditions at the two nearest grid points, a higher-order interpolation was instead performed using the zero- and first-order derivatives at the four nearest grid points [24]. In one dimension, a four-point Hermite spline can be represented as

$$f_I(\check{x}) = \sum_{p=1}^{4} H_p(\check{x}) f_q + \sum_{p=1}^{4} G_p(\check{x}) \frac{df}{d\check{x}}|_p, \tag{7}$$

where H and G are basis functions and f and all its first-order derivatives are evaluated at the grid points.

The relative error of both Basis- and Hermite-spline interpolation is due to the truncation error of the polynomial approximation as well as the error in the finite-difference approximation of the boundary derivatives in the case of the Hermite spline. This error can be explored using a plane-wave error analysis which provides the transfer function of the spline interpolation operator. A continuous plane-wave with a biased phase can be expressed in terms of the parametric coordinate system as

$$f(\check{x}, k\Delta, \phi) = A e^{\mathrm{i}k(x_n + \Delta\check{x}) + \phi} \quad \text{where} \quad \phi \in [-\pi, \pi). \tag{8}$$



(a) Hermite (b) Basis

Figure 3. Relative magnitude of interpolation error for Cubic (Degree = 4) Hermite and B-spline interpolation.

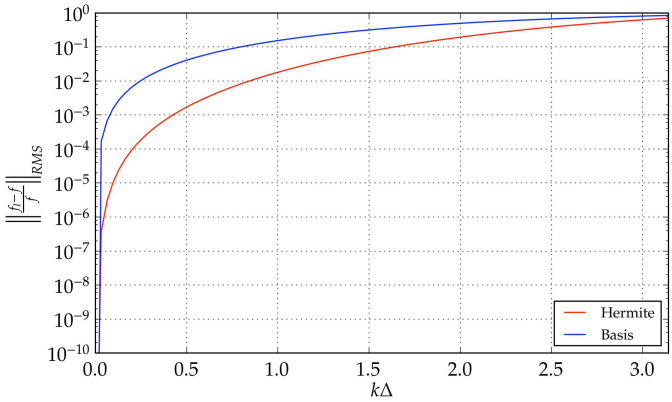


Figure 4. Root-mean-squared relative error over the range of interpolation positions for Cubic (Degree = 4) Hermite and B-spline interpolation.

Substituting Equation (8) into Equation (6), the interpolated value can be expressed as

$$f_I\left(\check{x}, k\Delta, \phi\right) = Ae^{\mathrm{i}k(x_n+\Delta\check{x})}e^{\mathrm{i}\phi}\sum_{p=0}^{M-1}\left(\sum_{q=0}^{K+1}c_{p,q}e^{\mathrm{i}k\Delta(q-K/2)}\right)\check{x}^p. \tag{9}$$

The difference between the true value of the plane wave and its interpolated value can be determined from a Monte Carlo simulation in the parameter space consisting of the parametric wavenumber $k\Delta$, the phase ϕ, and the location of the interpolated point $\check{x}$ relative to the nearest grid point. Figure 3 shows the magnitude of the interpolation error, defined as the magnitude of the difference between the interpolated value $f_I(\check{x})$ and the true value of the scalar $f(\check{x})$ at the same location with respect to the grid points. Not surprisingly the error increases as the distance between the interpolation point and the grid points increases, for both Hermite and B-splines, particular at higher wavenumbers. This change is far more gradual in the case of the B-spline but, as shown in Figure 4, the total error in the Hermite spline is lower than that of the B-spine for the same wavenumber.

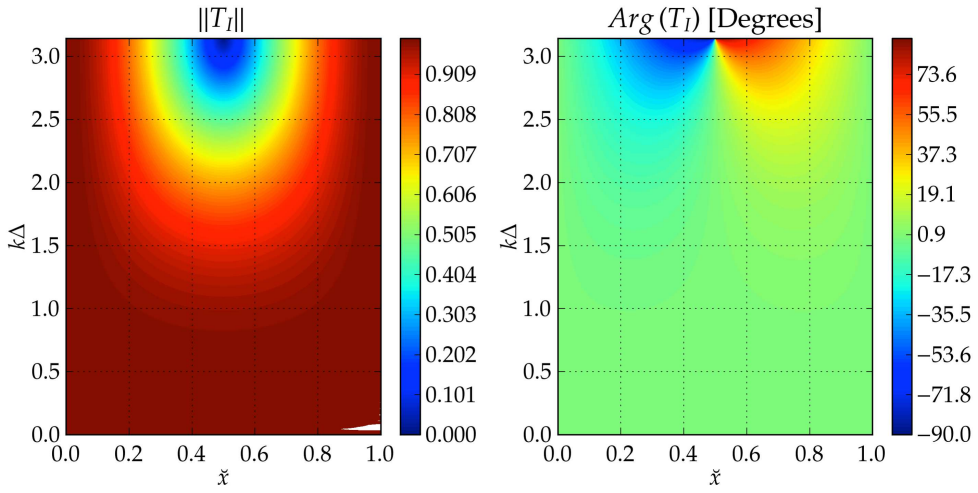


Figure 5. Transfer Function for a Cubic (degree = 4) Hermite Spline

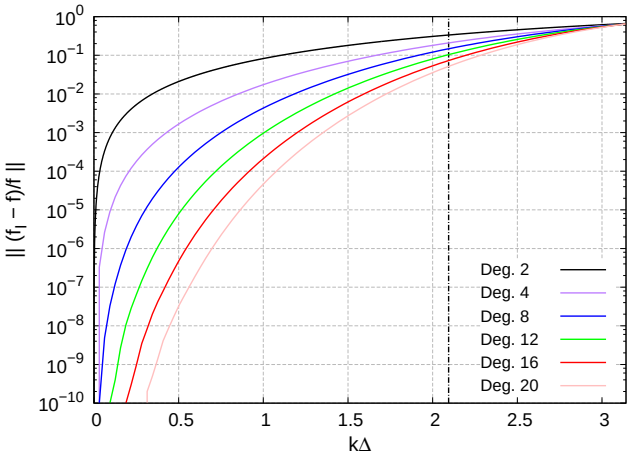


Figure 6. Root mean square relative error over the range of interpolation positions for different degree Hermite splines.

From Equation 9, a complex transfer function for this interpolation can be defined as

$$T_I\left(\check{x}, k\Delta\right) = \frac{f_I\left(\check{x}, \phi\right)}{f\left(\check{x}, \phi\right)} = \sum_{p=0}^{M-1} \left(\sum_{q=0}^{K+1} c_{p,q} e^{\mathrm{i}k\Delta(q-K/2)} \right) \check{x}^p, \tag{10}$$

which is only a function of the interpolation position $\check{x}$ and the relative wavenumber $k\Delta$. The modulus and argument of the transfer function defines the relative difference in magnitude and absolute difference in phase between the interpolated plane-wave and the original plane-wave, respectively. The contour plots in Figure 5 show the modulus and argument for the Hermite spline. The transfer function of the B-spline is similar, but shows less variation in interpolation position, as observed by the relative magnitude of the interpolation error. In both cases the argument of this transfer function suggests that both interpolation schemes may introduce oscillations in the higher wavenumbers.

The absolute magnitude of these interpolation errors and the effect on the particle trajectories and the evolution the of VGT and its invariants will depend on the energy content of the scalar (the fluid-velocity field) at different wavenumbers, the grid spacing, and the chosen degree of the fitted spline. Figure 6 shows the effect of varying the degree of Hermite interpolation. Assuming

negligible error in the scalar function at the grid points, a higher-order spline reduces the error at all wavenumbers. While the reduction in error from a linear (degree = 2) to a cubic (degree = 4) spline is significant, the incremental reduction in error decreases with higher order. If the effective cut-off wavenumber of the simulation is taken at $k\delta = 2/3$, following the 2/3 rule for de-aliasing, Figure 6 indicates that the error at this wavenumber should be approximately 22% for a cubic Hermite spline, decreasing to 5% for a Hermite spline of degree 20.

Higher-order interpolation requires the use of higher-order gradients, either directly in the case of Hermite interpolation, or through a larger coefficient matrix in the case of B-splines. Both cases ultimately require a wider interpolation kernel of size $N_P = M$, equal to the degree of the spline. This significantly increases the computation required for each particle. Given that the entire fluid phase must be computed for each time-step and that Lagrangian statistics require the calculation of numerous particles, it is therefore desirable to run the DNS with as many particles as possible. Naturally, this requires that the additional computation imposed by the particles be kept to a minimum. The benefits of higher-order interpolation are assessed in the context of the particle tracers in section 5.

4. Fluid-Particle Tracking Implementation

In order to test the benefits and possible need for high-order interpolation methods and to enable the processing of high-resolution boundary-layer simulations, the DNS code was modified to enable the use of Hermite spline interpolation of any specified order for both uniform and irregular grid spacing. Following the formulation for Hermite splines discussed in section 3, derivatives up to the required order are calculated at grid points nearest to the particle location using central or forward- or backward-biased finite-difference schemes, depending on the location of the particle with respect to the domain boundaries. Coefficients for these finite-difference schemes were computed for different-order schemes during runtime, following [25].

The velocity of each particle was determined at their initial location before the first Runge–Kutta time-step by converting one velocity component into physical space, and then performing three-dimensional interpolation of that velocity component at each particle location, before considering the next component. Interpolation was performed by first fitting one-dimensional Hermite splines of order M in the y, z cross-stream directions over a three-dimensional interpolation kernel of $M \times M \times M$ points around each particle, with final interpolation performed in the x direction. Structuring the interpolation in this format has the advantage that when a particle's interpolation kernel extends across the boundaries between computational nodes, the cross-stream interpolation can be handled by separate nodes. This reduces the communication between nodes to at most $M/2$ quantities for each velocity component.

Particle positions where updated at the start of each Runge–Kutta sub-step using their velocity from the end of the previous step. Velocities were then re-interpolated at this updated particle location after the fluid velocity was updated by the right-hand side of the governing equation, and after the Poisson correction for the divergence. The positions of all master (centre of particle is contained in the node) and slave particles (part of the interpolation kernel of a particle from an adjacent node) are stored for each computational domain, as done by Uhlmann [26]. With each update of the particle position, a test is performed to see whether a particle or its interpolation kernel crosses the boundary of its parent node, forcing the transition of a slave to a master or the formation of a new slave, respectively.

The velocity gradient tensor (VGT) at each particle was evaluated by first determining the velocity gradients at the centre of each cell using second-order central finite differences, consistent with the methodology used for the Poisson correction. It was assumed that the error associated with this scheme would be insignificant in comparison to the error introduction by the interpolation (at least for lower-order schemes), although this assumption has yet to be tested. Each component of the VGT was then interpolated following the same procedure used for the

Table 1. Parameters of the zero-pressure gradient boundary layer DNS. N_x, N_y and N_z are the grid sizes along the three axes, expressed for z in terms of collocation points, with Δ representing the corresponding resolutions at their coarsest points.

Re_τ	$(L_x, L_y, L_z)/\delta$	$\Delta x^+, \Delta y^+, \Delta z^+$	N_x, N_y, N_z	Particles
$126 - 190$	$7.6 \times 9.0 \times 10.5$	$13.4 \times 0.33 \times 6.2$	$257 \times 315 \times 768$	1200
$137 - 800$	$20.7 \times 15.7 \times 55.4$	$13.6 \times 0.18 \times 12.5$	$4097 \times 315 \times 768$	28800

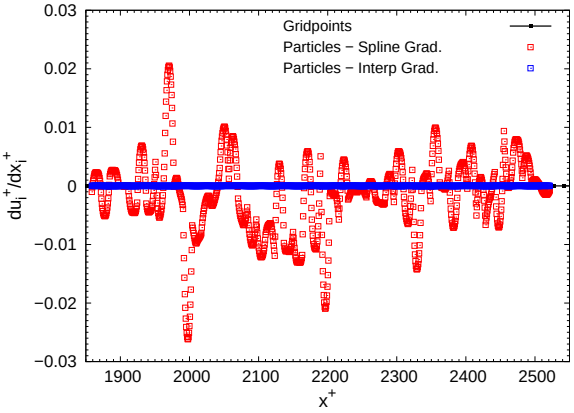


Figure 7. Comparison of the divergence in the particle velocity calculated from the gradient of a degree-4 Hermite spline fitted to the velocity field or by fitting Hermite splines to the gradients of the velocity field at each grid point. Values are shown for a series of particles distributed along a stream-wise line though the boundary layer simulation at $y^+ = 53$.

velocity field. Tests were also performed to see if the velocity gradients could be determined from differentiation of the spline fits as suggested by Yeung and Pope [14], which does not require the differentiation of the velocity field or additional interpolation, and is therefore far more efficient.

A similar process was performed in the channel flow using both cubic (degree = 4) B-splines and the four-point Hermite interpolation discussed in section 3. Particle positions and velocities were similarly computed at each Runge–Kutta sub-step. Pseudo-spectral methods were used to compute the velocity gradient at each grid point, from which each component of the velocity gradients tensor was calculated at each particle location via the same spline interpolation.

5. Results

To test the effect of the Hermite spline order and the size of the time-step on the particle trajectories and on the Lagrangian evolution of the invariants of the VGT, a turbulent boundary layer DNS was run over two moderate resolution grids whose dimensions and grid spacing are listed in Table 1. Initial testing was performed on the smaller grid in order to test the basic implementation of the code and the interpolation before extending the code to more realistic simulations.

In the absence of known particle trajectories, the accuracy of a given interpolation scheme was tested based on how much the particle position differed from that produced by a higher-order interpolation or a shorter time-step Δt, given that position error is a function of both the interpolation error and the time-step. The time-step from this point on will be quoted in terms of Courant-Friedrichs-Lewy number $CFL = U\Delta t/\Delta x$, where U is the free-stream velocity and Δx is the length of the computational cell. In addition to this, for an incompressible flow, the divergence $\partial u_i/\partial x_i$ measured at the particle location can be used as an indication of the accuracy of the velocity gradients calculated for each particle. Figure 7 shows the difference in the divergence obtained when calculating the velocity gradients at each particle from the gradient of a spline fitted to the velocity field (as suggested by [14]) compared to fitting a

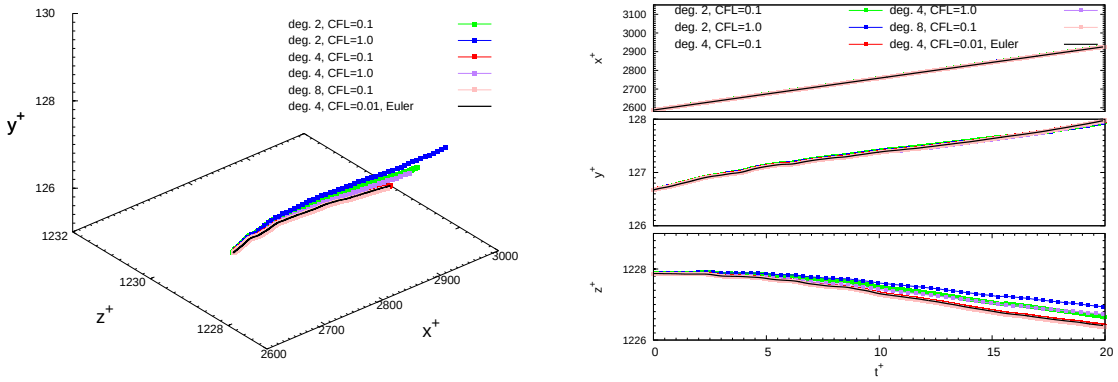


Figure 8. Particle trajectories calculated in the boundary layer simulation using Hermite splines of varying order over a series of different time-steps.

separate spline to the gradients of the velocity field at each grid point. Separate interpolation of each component of the VGT provides a divergence that is typically five orders of magnitude smaller than that obtained by differentiating the spline. Interpolating each component of the VGT is considerable more expensive, but provides a far more accurate estimation of the sub-grid particle velocity gradients.

The behaviour for the different interpolation and time-steps can be illustrated by considering the behaviour of an individual particle trajectory. Similar patterns were observed for most particles although, naturally, the degree to which a given particle trajectory diverges for different interpolation schemes will ultimately depend on the dynamics of the flow field at that time and position. Figure 8 shows the trajectory followed by a particle started at a wall-normal position of $y^+ = 126.7$ using a series of Hermite interpolations. Particle positions and velocities are updated for each Runge–Kutta time-step with the exception of the final trajectory, where the particle positions are instead updated using a first-order Euler time marching with CFL = 0.01. This final trajectory was generated in order to ensure that the use of the velocity fields inside the Runge–Kutta time-step did not have an adverse effect on the calculation of the particle's velocity. Comparison of the trajectories indicates that, for the same interpolation order, updating the particle position within the Runge–Kutta substeps enables time-steps at least an order of magnitude longer than for the Euler method, with negligible influence on the particle trajectory.

Figure 9 shows the variation in particle velocity as a function of time for each of the interpolation methods discussed above, demonstrating that both velocity and the consequent position are highly sensitive to the order of interpolation scheme and to the size of the time-step used in the simulation. The use of linear (degree = 2) interpolation results in significant oscillations in each component of the particle velocity, which is ultimately responsible for the difference in the particle trajectory. The use of cubic (degree = 4) interpolation reduces the amplitude of these oscillations, but the benefit of moving to a degree = 8 is less clear. In all cases these oscillations maintain approximately the same period and are strongly correlated with the particle position relative to the Eulerian simulation grid points.

The Lagrangian evolution of the Q_A, R_A invariants of the VGT following the same particle are shown in Figure 10 where the trajectories originate in the top left corner of the domain, which corresponds to the particle being in a region of stable focus stretching (see Figure 1). The invariants shown have been normalised by the mean of the second invariant of the rate of rotation tensor Q_W, which is directly proportional to the enstrophy [5]. While it is not clear what the

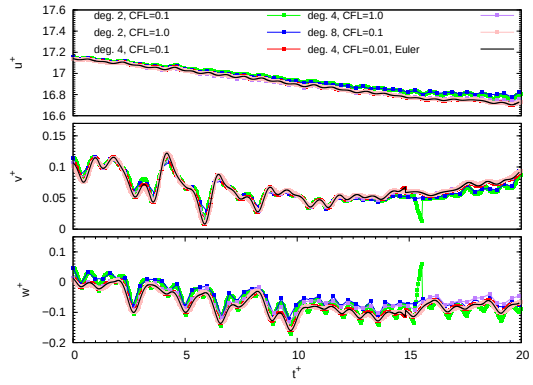


Figure 9. Particle velocity calculated in the boundary layer simulation from Hermite splines of varying order over a series of different time-steps.

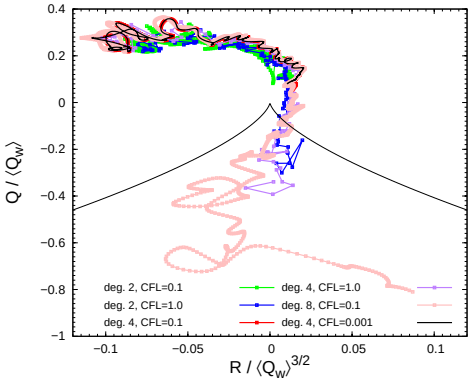


Figure 10. Lagrangian evolution of Q_A, R_A invariants of the velocity gradient tensor in the boundary layer simulation calculated for a single particle using Hermite splines of varying order over a series of different time-steps.

Re_τ	$(L_x, L_y, L_z)/\delta$	$\Delta x^+, \Delta y^+, \Delta z^+$	N_x, N_y, N_z	*Particles*
2000	$2\pi \times 2 \times \pi$	$10 \times 9 \times 5$	$1536 \times 633 \times 1536$	7.296×10^5

Table 2. Parameters for channel flow tests. Nomenclature as given in table 1.

Lagrangian trajectory of the invariants of an individual particle should look like, results shows a similar pattern of oscillation to the particle velocity. This oscillation is again reduced by using a higher-order interpolation although, unlike in the case of the particle velocity, these oscillations continue to reduce up to an interpolation degree of at least 8. This decreased oscillation for higher-degree splines suggests that the oscillations are at least partly due to the interpolation method.

The channel flow has been tested most extensively in the configuration given in Table 2. Figure 11 illustrates oscillations in the time series of quantities interpolated using both cubic B-splines and four-point Hermite splines. As in the case of the boundary layer, these oscillations do not correspond to non-monotonic behaviour of the unsteady flow in the surrounding grid points. As shown by Figure 12, oscillations in the velocity gradients are more noticeable. It should be noted that the magnitude of the oscillations is typically small but, as shown in Figure 10, they can have a significant effect on the observed Lagrangian evolution of the invariants, and will have consequences when it comes to the calculation of Lagrangian accelerations.

Despite significant differences between the resolution and the computation of velocity gradients in both the boundary layer and channel flow simulation, the similarity in the computed Lagrangian velocity and velocity gradient tracers suggests that the accuracy of the Lagrangian statistics is highly sensitive to the interpolation scheme. Given the shape of the transfer function of the interpolation method (see Figure 5) and the spacing of these oscillation with respect to the grid points, it is likely that these oscillations are purely the result of the sub-grid velocity interpolations. It may therefore be possible to low-pass filter these oscillations without effecting the Lagrangian statistics, assuming that the time-step of the simulation is small enough that these oscillations do not significantly influence the calculated particle trajectory, and given that there should be no appreciable energy in the fluid phase for parametric wavenumbers $k\Delta > 2\pi/3$ under the 2/3 de-aliasing of the DNS. Technically, this filtering should be a function

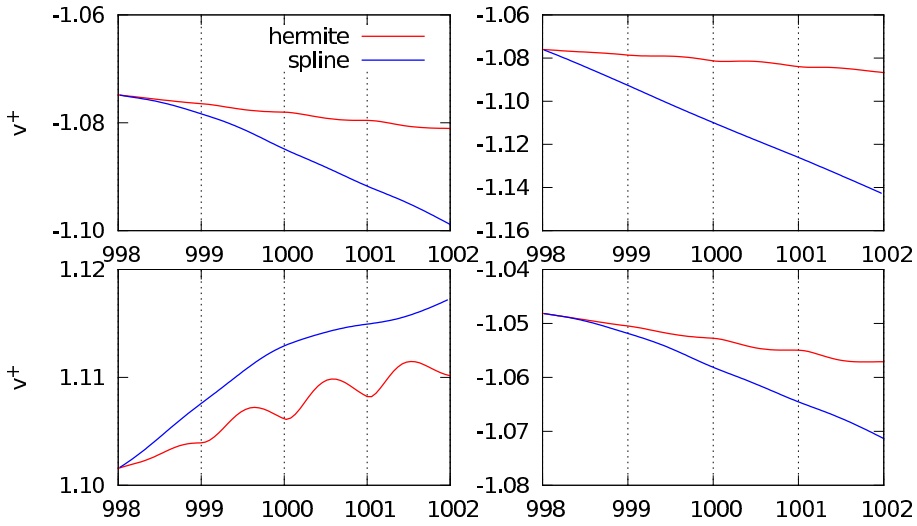


Figure 11. Time series of wall-normal velocity v for representative particles initialized at grid points on a cross-stream plane of the channel at $Re_\tau = 2000$. CFL = 0.3

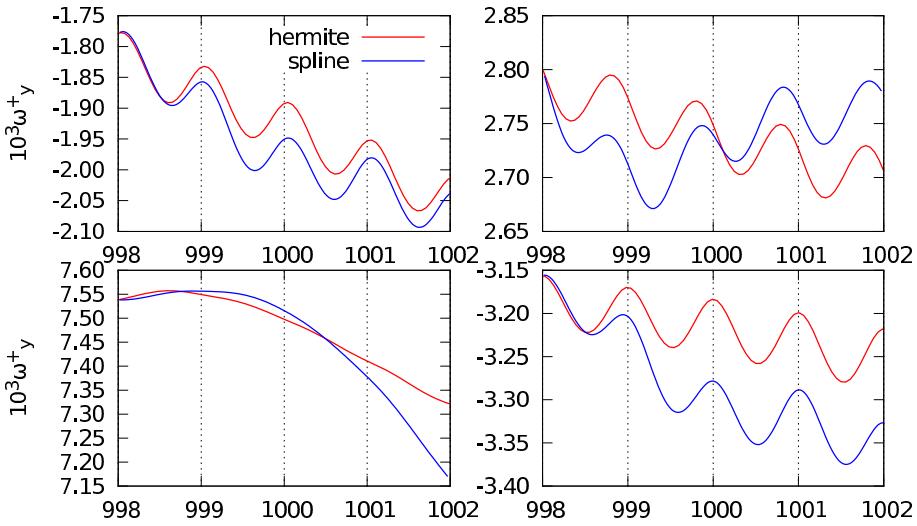


Figure 12. Time series of wall-normal vorticity ω_y for representative particles initialized at grid points on a cross-stream plane of the channel at $Re_\tau = 2000$. CFL=0.3

of positional variation in all three-directions and not a function of time. Given that the size of the computational cell varies with the wall-normal position, and that oscillations occur as the particle moves with respect to each face of the computational cell, the implementation of this filtering is non-trivial.

6. Conclusions

In order to enable the calculation of Lagrangian statistics and time-scale in wall-bounded flows, the Lagrangian tracking of fluid particles has been implemented in direct numerical simulation codes for both a boundary layer and for a fully developed channel flow. Sub-grid interpolation was implemented using both cubic B-spline, four-point Hermite and higher-order Hermite interpolation schemes. Despite significant differences in cell size, the computation of velocity gradients, and the implementation of the interpolation, the particle velocities and gradients in

both the boundary layer and the channel flow showed similar oscillations in their Lagrangian tracers. These oscillations are highly correlated with the movement of particles with respect to the computation grid and, given the similar pattern in the phase error of the interpolation method, appear to be due to the sub-grid interpolation method. While these oscillations in the particle velocity are relatively small and have negligible effect on the particle trajectories for time-steps on the order of CFL = 0.1, they appear to be the cause of more significant oscillations in the evolution of the invariants of the velocity gradient tensor. The extent to which these interpolation errors may influence the Lagrangian statistics is not yet clear.

Acknowledgments

This research was instigated as part of, and partially funded by, the Multiflow program of the European Research Council, and was partially supported by a grant from the Australian Research Council, both of which are gratefully acknowledged.

References

[1] Chong MS, Perry AE and Cantwell BJ 1990 A general classification of three-dimensional flow fields *Phys. Fluids* **2** 765–77
[2] Ooi A, Martín J, Soria J and Chong MS 1999 A study of the evolution and characteristics of the invariants of the velocity-gradient tensor in isotropic turbulence *J. Fluid Mech.* **381** 141–74
[3] Cantwell BJ 1992 Exact solution of a restricted Euler equation for the velocity gradient tensor *Phys. Fluids A* **4** 782–93
[4] Martín J, Ooi A, Chong MS and Soria J 1998 Dynamics of the velocity gradient tensor invariants in isotropic turbulence *Phys. Fluids* **10** 2336–46
[5] Atkinson C, Chumakov S, Bermejo-Moreno I and Soria J 2012 Lagrangian evolution of the invariants of the velocity gradient tensor in a turbulent boundary layer *Phys. Fluids* **24** 105104
[6] Elsinga GE and Marusic I 2010 Evolution and lifetimes of flow topology in a turbulent boundary layer *Phys. Fluids* **22** 015102
[7] Soria J, Sondergaard R, Cantwell BJ, Chong MS and Perry AE 1994 A study of the fine-scale motions of incompressible time-developing mixing layers *Phys. Fluids* **6** 871–84
[8] Jiménez J, Hoyas S, Simens MP and Mizuno Y 2010 Turbulent boundary layers and channels at moderate Reynolds numbers *J. Fluid Mech.* **657** 335–60
[9] Bisset DK, Hunt JCR and Rogers MM 2002 The turbulent/non-turbulent interface bounding a far wake *J. Fluid Mech.* **451** 383–410
[10] Westerweel J, Hofmann T, Fukushima C and Hunt JCR 2002 The turbulent/non-turbulent interface at the outer boundary of a self-similar turbulent jet *Exp. Fluids* **33** 873–78
[11] Mizuno Y, Atkinson C and Soria J 2011 Effect of the outer laminar flow on zero-pressure-gradient turbulent boundary layers *Bull. Am. Phys. Soc.* **56** G74
[12] Da Silva CB and Dos Reis RJN 2011 The role of coherent vortices near the turbulent/non-turbulent interface in a planar jet *Phil. Trans. Roy. Soc. A* **369** 738–53
[13] Da Silva CB, Dos Reis RJN and Pereira JC 2011 The intense vorticity structures near the turbulent/non-turbulent interface in a jet *J. Fluid Mech.* **685** 165–90
[14] Yeung P and Pope S 1989 Lagrangian statistics from direct numerical simulations of isotropic turbulence *J. Fluid Mech.* **207** 531–86
[15] Squires KD and Eaton JK 1991 Lagrangian and Eulerian statistics obtained from direct numerical simulations of homogeneous turbulence *Phys. Fluids A* **3** 130–43
[16] Choi JI, Yeo K and Lee C 2004 Lagrangian statistics in turbulent channel flow *Phys. Fluids* **16** 779–93
[17] Simens MP, Jiménez J, Hoyas S and Mizuno Y 2009 A high-resolution code for turbulent boundary layers *J. Comput. Phys.* **228** 4218–31
[18] Sillero J, Borrell G, Jiménez J and Moser RD 2011 *Recent Advances in the Message Passing Interface* (Berlin: Springer) pp 218–27
[19] Kim J, Moin P and Moser R 1987 Turbulence statistics in fully developed channel flow at low Reynolds number *J. Fluid Mech.* **177** 133–66
[20] Hoyas S and Jiménez J 2006 Scaling of the velocity fluctuations in turbulent channels up to $Re_\tau = 2003$ *Phys. Fluids* **18** 011702
[21] Denga X and Maekawab H 1997 Compact high-order accurate nonlinear schemes *J. Comput. Phys.* **130** 77–91

[22] Nagarajan S, Lele SK and Ferziger JH 2003 A robust high-order compact method for large-eddy simulation *J. Comput. Phys.* **191** 392–419
[23] Stegeman PC, Soria J and Ooi A 2012 Comparison of the dynamics of particles in a flow field with Reynolds and Favre filtered flow velocities *Australasian Fluid Mech. Conf.* eds. P.A. Brandner and B.W. Pearce (http://www.afms.org.au) **320**
[24] Hildebrand F 1974 Introduction to numerical analysis *New York: McGraw-Hill, 2nd ed.*
[25] Fornberg B 1988 Generation of finite difference formulas on arbitrarily spaced grids *Math. of Comput.* **51** 699–706
[26] Uhlmann M 2004 *Simulation of particulate flows on multi-processor machines with distributed memory* (Madrid: Ciemat)