https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

A conservative immersed boundary method for the multi-physics urban large-eddy simulation model uDALES v2.0

Sam O. Owens[1], Dipanjan Majumdar[1], Chris E. Wilson[1], Paul Bartholomew[2], and Maarten van Reeuwijk[1]

[1]Department of Civil and Environmental Engineering, Imperial College London, London SW7 2AZ, United Kingdom
[2]EPCC, University of Edinburgh, Edinburgh EH8 9BT, United Kingdom

Correspondence: Sam O. Owens (sam.owens18@imperial.ac.uk)

Abstract. uDALES is an open-source multi-physics microscale urban modelling framework, capable of performing large-eddy simulation (LES) of urban airflow, heat transfer, and pollutant dispersion. We present uDALES v2.0, which has two main new features: 1) an improved parallelisation that prepares the codebase for conducting exascale simulations; and 2) a conservative immersed boundary method (IBM) suitable for an urban surface that does not need to be aligned with the underlying Cartesian grid. The urban geometry and local topography are incorporated via a triangulated surface with a resolution that is independent of the fluid grid. The IBM developed here includes the use of wall functions to apply surface fluxes, and the exchange of heat and moisture between the surface and the air is conservative by construction. We perform a number of validation simulations, ranging from neutral, coupled internal-external flows and non-neutral cases. Good agreement is observed, both in cases in which the buildings are aligned with the Cartesian grid and when they are at an angle. We introduce a validation case specifically for urban applications, for which we show that supporting non grid-aligned geometries is crucial when solving surface energy balances, with errors of up to 20% associated with using a previous version of uDALES.

1 Introduction

If current trends continue, it is estimated that the global average temperature is expected to increase by at least 1.5°C by 2050 (Masson-Delmotte et al., 2022), and that the fraction of people living in cities will increase from around 50% today to 58% over the next 50 years (United Nations Human Settlements Programme (UN-Habitat), 2022). To illustrate, by 2050 the climate of London, UK is expected to resemble that of Barcelona, Spain (Bastin et al., 2019), and could be home to over one million more people (Greater London Authority, 2023). Excess heat, particularly when coupled with humidity, is uncomfortable at best and lethal at worst, and can damage infrastructure and impact wildlife. Wind is another environmental problem present in urban areas; although wind speeds tend to be slower overall compared to rural surroundings (Oke et al., 2017), tall buildings can create windy pockets at their base, which can be a safety concern (Xu et al., 2017). Also, cities tend to have poorer air quality, largely due to greater emission of pollutants within the city itself (Enenkel et al., 2020). Urban areas therefore face particular challenges in comparison to rural areas, and these issues will become ever more salient. It is therefore crucial to study

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

air flow, heat transfer, and pollutant dispersion in the built environment in order to provide insight into urban microclimate and air quality.

To study these processes, a combination of observations, experiments (physical modelling), and numerical modelling are used. Given the increasing capabilities of high performance computing (HPC), numerical models are being developed to take advantage of these resources. These models operate at a range of scales, from the mesoscale ($\mathcal{O}(1\ \text{km})$), at which the effect of urban areas are parametrised (Walters et al., 2019; Skamarock et al., 2019), to the microscale ($\mathcal{O}(1\ \text{m})$), at which the surface is resolved at high resolution. Airflow (and transport of heat, moisture, and pollutants) is commonly modelled using computational fluid dynamics (CFD). Pollutant dispersion also potentially involves modelling the chemical reactions of various aerosols. Incorporating heat transfer at the surface requires modelling the surface energy balance, *i.e.* the net radiation, convection, evaporation, and conduction at the surface-atmosphere interface.

There are standalone tools for modelling airflow, heat transfer, and pollutant dispersion individually, but as these processes fundamentally interact, multi-physics frameworks have been developed that have some degree of coupling between them. Of the available multi-physics tools for microscale modelling of urban processes, some are particularly focused on microclimate and pedestrian comfort in realistic urban environments. These tend to have advanced thermal models, typically employing ray-tracing for radiation. Examples include ENVI-met (Huttner, 2012), and frameworks that rely on open-source general-purpose CFD codes such as SOLENE-Microclimate (Musy et al., 2015, 2021), which uses Code_Saturne; urbanMicroclimateFoam (Kubilay et al., 2018) which uses OpenFoam; and the framework described in Wong et al. (2021) which also uses OpenFoam and employs EnergyPlus for heat transfer. However, these frameworks sacrifice accuracy when modelling airflow by using the Reynolds-averaged Navier-Stokes (RANS) approach. RANS only resolves the mean flow, and entirely relies on turbulence modelling to incorporate the effect of fluctuating components. As a result, RANS often fails to accurately reproduce transient flow features and quantities such as turbulent kinetic energy (van Hooff et al., 2017). On the contrary, large-eddy simulation (LES) resolves the mean flow as well as the large-scale turbulence with a length scale greater than the filter size (often equal to the grid size), with the small-scale turbulence being modelled using subgrid-scale models. As a result, LES is inherently more accurate than RANS (Murakami, 1993; Baker, 2007), but involves a significant increase in computational cost (Chen, 2009).

The complexity of LES can be reduced by using a Cartesian mesh. This increases computational performance because solving for the pressure can be carried out using the Fast Fourier transform (FFT) algorithm, which is the most efficient method (Ferziger et al., 2019). The immersed boundary method (IBM) can be used to model the effect of obstacles on a fluid defined on a Cartesian grid, though traditionally the IBM is discussed in the context of direct numerical simulation (DNS) approaches (Verzicco, 2023). IBMs for LES have different requirements because unlike DNS, the flow near the obstacle is not fully-resolved. LES frameworks opting for an IBM with a Cartesian mesh approach have typically evolved from atmospheric flow solvers and have been equipped with multi-physics surface schemes capturing urban processes, for example PALM-USM (Resler et al., 2017; Maronga et al., 2020), City-LES (Watanabe et al., 2021), and uDALES v1.0 (Suter et al., 2022). These frameworks can currently only model obstacles that align with the Cartesian grid, so urban geometries that do not actually meet this requirement are approximated as doing so using a voxel representation. For example, in Sect. 4.3 of Suter et al. (2022), the geometry was generated using a raster of a surface elevation map (obtained from GIS data), and the shape of buildings

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

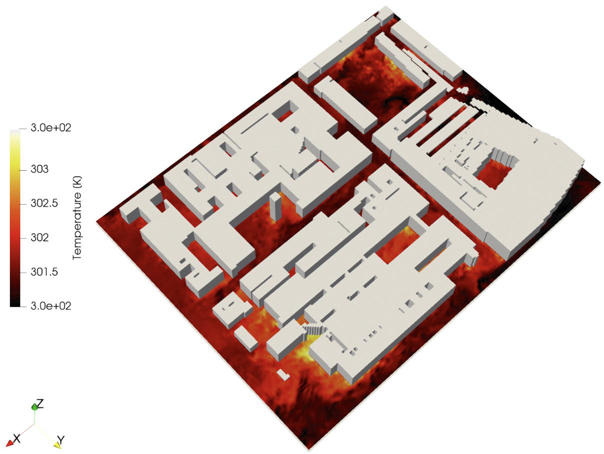

Figure 1. Temperature at street level in a realistic urban environment; simulated using uDALES v1.0 (Suter et al., 2022).

that are not aligned with the grid are clearly distorted, as shown in Fig. 1. This is potentially problematic for realistic, complex geometries, particularly with respect to radiation as this is strongly influenced by the surface area and orientation.

The aim of this paper is to present a new version of uDALES, as v1.0 is limited in several key ways. Firstly, as previously mentioned, it is only capable of modelling geometries that are aligned with the Cartesian computational grid, which hinders its ability to faithfully represent realistic urban geometries. Also, it has a one-dimensional domain decomposition implemented with bespoke routines using the message passing interface (MPI) library, which means it cannot effectively utilise the large number of processors available on HPC systems. The final limitation is that it is not possible to use the FFT algorithm to solve for the pressure with inflow-outflow boundary conditions, though it is possible to use the FFT in periodic situations.

uDALES v2.0 has been primarily developed to address these key issues. The geometry is specified as an unstructured triangulated surface that is given independently of the grid using the STL file format, and a novel method for applying surface momentum and heat fluxes given this non-grid-aligned approach has been implemented within the framework. It has a two-dimensional domain decomposition, implemented using the 2DECOMP&FFT library (Li and Laizet, 2010; Rolfo et al., 2023), thus increasing the limit of parallelisation of the code. Finally, it is capable of using an FFT-based method to solve for the pressure when using inflow-outflow boundary conditions, thereby enhancing the performance of simulations employing these conditions. In addition to these major changes, uDALES v2.0 also has an improved algorithm for calculating shortwave radiation, and a novel method to prevent the formation of artificial flow features in periodic simulations.

The paper is structured as follows. Section 2 gives an overview of the uDALES framework, Sect. 3 describes the improvements that together make up uDALES v2.0, and Sect. 4 presents an evaluation of the model by comparing with uDALES v1.0 and other physical and numerical models in canonical urban cases. Finally in Sect. 5, future development and applications of the framework are discussed.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

2 Model overview

uDALES is an LES-based framework designed to model air flow, heat transfer, and pollutant dispersion in urban environments. The governing equations for the LES are unchanged from Suter et al. (2022), and are discretised using the finite difference method on a Cartesian grid. The presence of obstacles are modelled using the IBM approach, and wall functions are used to parametrise processes near the immersed boundary. Solid surfaces are divided into facets, each with user-defined material properties. The surface energy balance of each facet can be modelled in a two-way coupled manner with the flow, enabling surface temperatures and moisture content (of vegetative surfaces) to be evolved. There is also the capability to model trees in terms of drag, deposition of pollutants, and heat transfer. uDALES has been used for studies concerning atmospheric boundary layer processes (Grylls et al., 2020), air quality (Grylls et al., 2019; Lim et al., 2022), the effect of trees (Grylls and van Reeuwijk, 2022), and parametrisations for larger-scale models (Sützl et al., 2021a, b). The framework has been described in detail in (Suter et al., 2022), so this work only discusses novel or improved aspects and relevant concepts.

2.1 Large-eddy simulation

Away from the boundaries, uDALES solves the (filtered) incompressible Navier-Stokes equations in the Boussinesq approximation.

$$\frac{\partial u_i}{\partial t} = -\frac{\partial p}{\partial x_i} - \frac{\partial u_j u_i}{\partial x_j} + \frac{\partial}{\partial x_j}\left(K_\mathrm{m}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right) + \frac{\theta_\mathrm{v} - \langle\theta_\mathrm{v}\rangle}{\langle\theta_\mathrm{v}\rangle} g\delta_{i3} + \mathcal{F}_i + S_{u_i}, \qquad (1)$$

where u_i is the filtered (Cartesian) velocity component in the x_i direction, p is the deviatoric kinematic pressure, $\mathcal{F}_i$ is a large-scale forcing, S_{u_i} is any local source or sink (*e.g.* due to wall shear stress), θ_v is the virtual potential temperature, K_m is the turbulent eddy viscosity calculated using the Vreman subgrid model (Vreman, 2004), and $\langle\cdot\rangle$ denotes the xy-plane average. The pressure is solved using a two-step correction method. Scalar quantities φ, such as the liquid potential temperature (θ_l), total specific humidity (q_t), and pollutant concentrations (c) evolve according to

$$\frac{\partial \varphi}{\partial t} = -\frac{\partial u_i \varphi}{\partial x_j} + \frac{\partial}{\partial x_j}\left(K_\mathrm{h}\frac{\partial \varphi}{\partial x_j}\right) + S_\varphi, \qquad (2)$$

where K_h is the turbulent eddy diffusivity and S_φ are local source or sink of the scalar (*e.g.* due to surface heat flux or chemical reactions).

The equations are discretised in space on an Arakawa-C grid (Arakawa and Lamb, 1977), which entails defining pressure and scalars at cell centres, and defining the three velocity components (u_x, u_y, u_z) at the lower cell face in each direction (x, y, z) in a staggered manner. Second-order central difference schemes are used for all prognostic fields except pollutant concentrations, for which a κ-advection scheme (Hundsdorfer et al., 1995) is employed to ensure monotonicity. Time integration is carried out by a third-order Runge-Kutta scheme. The y-grid spacing is always equidistant, whilst the z-grid spacing may be stretched. In uDALES v1.0, it is also possible to have a stretched grid spacing in the x-direction, though this is rarely used in practise due to considerations with the pressure solver that are discussed in Sect. 3.2. The IBM is used to set appropriate boundary conditions

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

for solid regions, and wall functions provide surface shear stresses and sensible and latent heat fluxes for fluid cells with solid neighbours.

When using inflow-outflow boundary conditions, the inlet can be defined by using either a fixed profile or a time-varying plane generated using a precursor simulation (this is only possible for the x-direction). The outflow condition is convective, meaning the variables are transported by the vertically-averaged velocity at the outlet.

2.2 Wall functions

Urban LES requires wall functions to model surface fluxes since the typical resolution is insufficient to resolve the wall boundary layers. Wall functions are parametrisations that use velocity and scalar values at the cells closest to the wall, as well as the conditions at the wall, to predict the wall shear stress τ_{w}, sensible heat flux H, and latent heat flux E (Louis, 1979; Uno et al., 1995; Jarvis, 1976; Stewart, 1988). The exact form of the wall functions used in uDALES is given in Suter et al. (2022); Suter (2018) - below a conceptual description is provided, which is illustrated in Fig. 2. For a given fluid cell adjacent to the boundary, the wall-tangential velocity, liquid potential temperature, surface temperature, and distance to the wall are denoted as u_{a}, θ_{a}, T_{s}, and d respectively. The flux of (tangential) momentum is given by:

$$\tau_{\mathrm{w}}/\rho \equiv u_\tau^2 = f_{\mathrm{m}}(d, u_{\mathrm{a}}, \theta_{\mathrm{a}}, T_{\mathrm{s}}; z_0, z_{0,\mathrm{h}}), \tag{3}$$

where f_m is the wall function for momentum and ρ is a reference air density. The momentum roughness length (z_0) and heat roughness length ($z_{0,\mathrm{h}}$) are parameters specified for each facet. Similarly, the flux for potential temperature is given by:

$$H/\rho c_p \equiv u_\tau \theta_\tau = f_{\mathrm{h}}(d, u_{\mathrm{a}}, \theta_{\mathrm{a}}, T_{\mathrm{s}}; z_0, z_{0,\mathrm{h}}, Pr_{\mathrm{t}}), \tag{4}$$

where f_{h} is the wall function for temperature, c_p is the air specific heat capacity, and Pr_{t} is the turbulent Prandtl number (a model parameter). Finally, denoting the atmospheric specific humidity, surface saturation humidity (itself dependent on T_{s}), and surface relative humidity by q_{a}, q_{sat}, and RH respectively, the corresponding specific humidity flux is given by:

$$E/\rho L_{\mathrm{v}} \equiv u_\tau q_\tau = f_{\mathrm{e}}(u_{\mathrm{a}}, q_{\mathrm{a}}, T_{\mathrm{s}}, q_{\mathrm{sat}}, \mathrm{RH}, C_{\mathrm{h}}; r_{\mathrm{c}}, r_{\mathrm{s}}), \tag{5}$$

where f_{e} is the wall function for specific humidity, L_{v} is the air latent heat of vaporisation, $C_{\mathrm{h}} \equiv u_\tau\theta_\tau/u_{\mathrm{a}}(T_{\mathrm{s}} - \theta_{\mathrm{a}})$ is the heat transfer coefficient, and r_{c} and r_{s} are the canopy and soil resistances respectively, which are model parameters.

The fluxes of momentum, heat, and humidity are converted to volumetric source or sink terms (S_{u_i}, $S_{\theta_{\mathrm{l}}}$ and $S_{q_{\mathrm{t}}}$ respectively) by multiplying by the surface area and dividing by the volume over which they act; the relevant area being the fraction of the adjacent facet that is within the fluid cell of interest, and the volume is that of the cell itself. For example in Fig. 2, the fluid cell depicted with a dotted line experiences fluxes from Facet 3. In this 2D representation half of Facet 3 is inside the cell, so this is the area used in determining the source/sink terms. Note this must be considered separately for each of the staggered grids.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

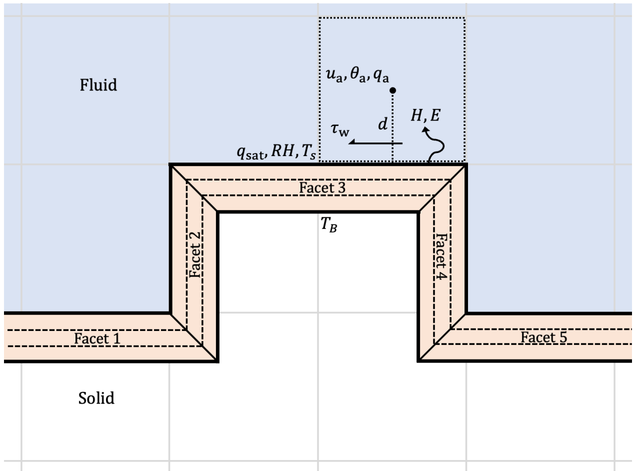

Figure 2. Two-way coupled wall functions and surface energy balance in uDALES. Fluid cells adjacent to the boundary experiences a wall shear stress (τ_w) and fluxes of sensible and latent heat (H and E), which are determined by wall functions. These fluxes act as volumetric sources or sinks in the fluid governing equations, and simultaneously contribute to the surface energy balance for the corresponding facet. Note that facets are not in fact trapezoidal - this representation is used to avoid them overlapping in this diagram.

2.3 Surface energy balance scheme

The surface energy balance is written in terms of the heat fluxes (in $\mathrm{Wm^{-2}}$) at the surface of each facet.

$$K^* + L^* = H + E + G_0 \tag{6}$$

Here K^* and L^* are the net shortwave and longwave radiation respectively, and G_0 is conduction. K^* and L^* are calculated using the radiosity approach, which assumes all reflection is diffuse, thereby allowing the use of view factors (Oke et al., 2017). An albedo (α), and emissivity (ϵ) are accordingly specified for each facet. The turbulent fluxes H and E are the average of the fluxes at neighbouring fluid cells as calculated via the wall functions (with opposite sign to that experienced by the fluid). Note that the surface energy balance is generally evolved using a larger timestep than the LES, therefore these fluxes are time-averaged. Since heat conduction is defined by $G_0 = -\lambda \frac{\partial T}{\partial \xi}$, where T is the internal facet temperature, ξ is the depth into the facet, and λ is the thermal conductivity; Eq. (6) acts as the exterior boundary condition for a 1D enthalpy equation for the temperature inside the facet:

$$C \frac{\partial T}{\partial t} = \frac{\partial}{\partial \xi}\left(\lambda \frac{\partial T}{\partial \xi}\right), \tag{7}$$

where C is the volumetric heat capacity. Equation 7 is solved numerically by discretising each facet into a number of layers along the depth (shown using dashed lines in Fig. 2), with an isothermal boundary condition at the interior (denoted T_B in Fig. 2). The solution is assumed to be a piecewise continuous quadratic polynomial. The properties of each layer (C, λ and thickness) can be specified freely for each facet.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

Table 1. Upgrades from uDALES v1.0 to v2.0.

Feature	uDALES v1.0	uDALES v2.0
Domain decomposition	One-dimensional	Two-dimensional
Pressure solver	Only entirely FFT-based for periodic cases	Entirely FFT-based for periodic and inflow-outflow cases
Geometry representation/IBM	Blocks (voxels) - grid-aligned	Triangulated surface - grid-independent
Shortwave radiation algorithm	Inaccurate shading	Accurate shading
Periodic (driver) simulations	No method	Method to avoid permanent streamwise super-structures

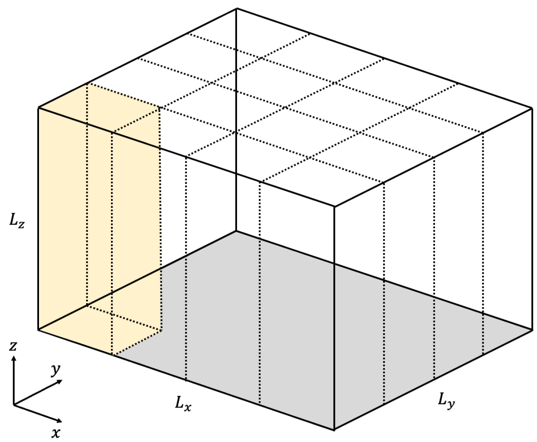

Figure 3. A two-dimensional domain decomposition. In this case, the x and y directions are decomposed across 4 processors, so there are 16 in total. The pencil operated on by a single processor is highlighted in yellow.

3 Improvements

Table 1 summarises the improvements to the uDALES framework that are discussed in this section.

3.1 2D domain decomposition

uDALES v1.0 has a one-dimensional domain decomposition, which involves splitting the computational domain into 'slabs' along a single direction - the y-direction in this case. Each processor (MPI rank) generally stores and manipulates data structures associated only within a single slab. This means the maximum number of processors that can be used is limited to the number of grid points in the y-direction (N_y). Given that typical use cases have N_y equal to 256-1024, and a single node on ARCHER2 (the UK National Supercomputing Service) has 128 processors available, simulations with uDALES v1.0 may only use 10 nodes at most of the thousands that are available (EPCC, 2022).

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

A two-dimensional decomposition means the domain is divided into 'pencils', which is achieved in uDALES v2.0 using the 2DECOMP&FFT library (Li and Laizet, 2010; Rolfo et al., 2023), which is also used by the Xcompact3d framework (Bartholomew et al., 2020). The default pencil orientation is in the z-direction, meaning the x- and y-directions are parallelised as shown in Fig. 3. The extra dimension of parallelisation compared to uDALES v1.0 means that it is possible to run simulations with many more processors, which practically increases the scaling capabilities of the code. The maximum number of processors that it is possible to use is equal to $N_x \times N_y$, where N_x is the number of points in the x-direction. In addition to the computational cells within the pencil, each rank also stores the cells just outside it. These are called halo cells, and are stored because they are part of the finite difference stencil for cells at the edge of the pencil. This naturally requires communication between adjacent ranks every timestep. The only time when data is transposed from z-pencil to x- and y-pencils is while solving for the pressure, because the full extent of the data in a particular direction is required for performing the Fourier transform for that direction.

Figure 4 shows the strong scaling for cases of various sizes and boundary conditions, which were run on ARCHER2 using 128 processors per node. To test the lower-level routines without any novel features, periodic boundary conditions were used in a $2048\ \mathrm{m}^3$ case with 1024^3 and 2048^3 grid points in the absence of geometry. These problem sizes are realistic for urban use cases that involve modelling a convective atmospheric boundary layer at $\mathcal{O}(1\ \mathrm{m})$ resolution. The speed-up for each case is shown in Figs 4a&b, respectively. The smaller case requires at least 8 nodes (in terms of memory usage), and a good parallel efficiency is observed up to 128 nodes. The larger case requires at least 64 nodes, and achieves reasonable efficiency, though less so than the smaller case. Note that all but one (the 1024^3 case run with 1024 processors) are impossible to run using uDALES v1.0 due to the number of processors exceeding the number of points in a single direction.

To test the key new features of uDALES v2.0, *i.e.* the IBM and fully FFT-based pressure solver for inflow-outflow boundary conditions, a case similar to the validation case shown in Sect. 4.3 (the original, non-rotated case) is simulated. For scaling purposes higher resolutions are used, but temperature is not solved for, and the domain width is smaller (the domain size is $190 \times 114 \times 114\ \mathrm{m}^3$). The smaller case has a grid size of $640 \times 384 \times 384$, and larger case has a grid size of $1280 \times 768 \times 768$. The speed-up for each case is shown in Figs 4c&d, respectively. It is possible to run the smaller case on a single node, and a good parallel efficiency is observed until 8,192 processors are reached, beyond which extra parallelisation in fact becomes detrimental. The larger case required 4 nodes minimum and demonstrated good scaling up to 16,384 cores.

3.2 Pressure solver

The pressure correction is defined as the difference in pressure between successive timesteps, and this quantity satisfies a Poisson equation, which is derived in Appendix A. The Poisson equation in uDALES is solved most efficiently using the FFT in the x- and y-directions and Gaussian elimination (GE) in the z-direction. The reason for not using the FFT in the z-direction as well is because GE supports a non-equidistant grid (unlike the FFT), and there is not likely to be a performance gain using the FFT rather than Gaussian elimination due to their algorithmic complexity ($\mathcal{O}(N \log N)$ versus $\mathcal{O}(N)$ respectively). The velocity boundary conditions in the x- and y-directions determine the specific type of transform performed. If velocity is periodic then the pressure is also periodic and the regular discrete Fourier transform (DFT) is used. If the velocity is inflow-

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

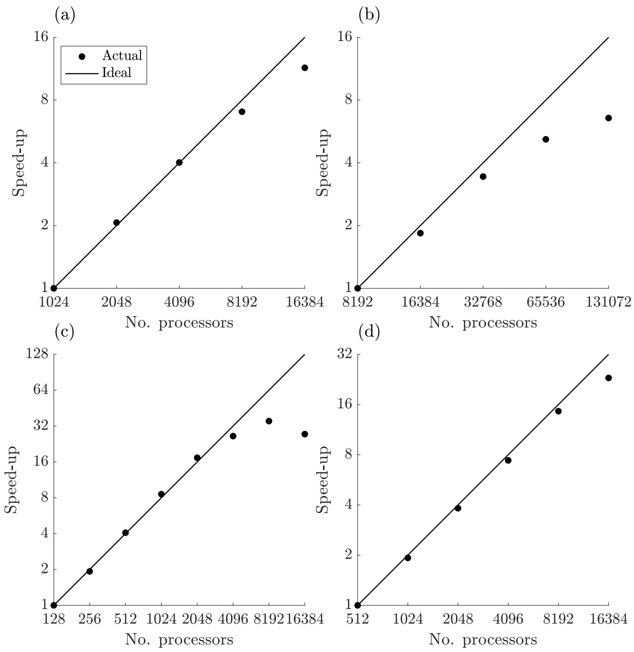

Figure 4. Strong scaling for cases: without IBM and periodic boundary conditions using (a) 1024^3 points, (b) 2048^3 points; with IBM and inflow-outflow boundary conditions using (c) $640 \times 384 \times 384$ points, (d) $1280 \times 768 \times 768$ points.

outflow then the pressure has a Neumann boundary condition and a discrete cosine (DCT) transform (for a staggered grid) is used. The details of these transforms can be found in Schumann and Sweet (1988); uDALES v2.0 uses the FFTW library to implement them (Frigo and Johnson, 1998). To solve for each direction requires transposing data between the three pencil orientations, which is achieved using routines from the 2DECOMP&FFT library (Li and Laizet, 2010; Rolfo et al., 2023).

Note that uDALES v1.0 does not use the DCT for inflow-outflow situations; instead the cyclic reduction (CR) algorithm is used to solve the x- and z-directions, while the FFT is used in the y-direction (meaning the y-direction is always periodic). A non-equidistant grid can be used with CR, hence the ability to use a stretched x-grid in uDALES v1.0. However, the CR-based solver is slower than the fully FFT-based solver and is only practical to use with a 1D domain decomposition because CR requires the full extent of the data in both the x- and z-directions. This is why the inflow-outflow cases are solved using the DCT in uDALES v2.0. When using inflow-outflow lateral boundary conditions, the boundary condition at the top of the domain must be treated with care. In periodic simulations, there is a no-penetration condition, but with inflow-outflow lateral boundaries, fluid must be allowed to pass through, which is equivalent to setting a non-zero vertical velocity (u_z). The physical reasoning for this is that the incoming flow responds to the geometry, which may result in net motion in the z-direction. This non-zero u_z is determined by the plane-averaged pressure gradient, and without it, the flow is not incompressible at the top, which is unacceptable both numerically and physically. Further detail on solving the Poisson equation and applying the top boundary condition can be found in Appendix A.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

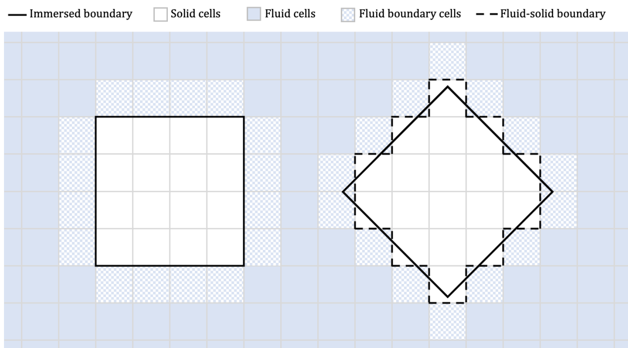

Figure 5. A 2D geometry depicted for grid-aligned (left) and non-grid-aligned (right) situations.

3.3 Geometry representation

The geometry in uDALES v1.0 is specified as a set of cuboid blocks with a given size in each Cartesian direction, and the faces of the blocks correspond to the facets. This can be classified as a voxel representation, with each block effectively composed of a set of voxels. The approach is fundamentally limited as it requires the facets to align with the edges of computational cells, which is problematic when the geometry of interest does not in fact do so, as is frequently the case for realistic urban environments. Also, the block format is bespoke, which makes it impossible to import geometries directly from other software packages. Instead, custom tools must be relied upon to perform the necessary voxelisation (from a 3D model) or rasterisation and extrusion (from a 2D surface elevation map).

The geometry in uDALES v2.0 is specified as a triangulated surface using an STL file, which consists of a list of triangles in terms of their three vertices in 3D Cartesian space, as well as their surface normals. This formatting is clearly independent of any grid specification, and is widely supported by other software packages. By treating each triangle as a facet in uDALES, the framework can use this representation to capture non-grid-aligned geometries more faithfully in comparison to a voxel representation, and to leverage the tools available to manipulate them.

3.3.1 Conservative immersed boundary method

Immersed boundary methods (IBMs) are used to model flow around complex geometries while benefitting from the advantages of using a Cartesian grid (Verzicco, 2023). In general, IBMs modify the governing equations of cells inside and/or close to the immersed boundary in order to set the desired flow conditions. uDALES assumes the geometry is non-porous and stationary, which are modelled using no-slip and no-penetration boundary conditions, with zero flux of momentum and scalars across boundaries (apart from those parametrised by wall functions). This last condition is particularly important for scalars - the IBM must be conservative, meaning there is no net transfer of heat, moisture, or pollutant concentration between the fluid and solid regions due to the advection and turbulent diffusion processes.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

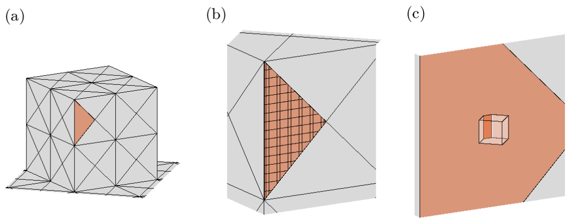

Figure 6. Surface treatment in uDALES v2.0: (a) triangular facets, with one coloured orange; (b) a facet divided into sections according to grid cells; (c) a facet section and fluid boundary cell.

In uDALES v2.0, the grid points are categorised as either fluid or solid for each of the staggered grids depending on whether they are respectively outside or inside the triangulated surface describing the geometry. This has been achieved using the ray-casting method (Borazjani et al., 2008). Boundary points for both types are identified as those with at least one neighbour of the other type. Note that in this text, 'point' is often used to refer to a cell centre (on a particular staggered grid), therefore to refer to a cell as being fluid or solid means that the cell centre is classified as a fluid or solid point respectively.

For each solid point on the velocity grids, the corresponding velocity component is set to zero. For fluid boundary points on all grids, the governing equations are modified such that any flux contribution from neighbouring solid points is negated. This means that when the facets are aligned with the cell edges, the desired boundary conditions are satisfied exactly. If not, they must be interpreted as follows; no-slip and no-penetration mean the velocity at solid points is zero, and the zero flux condition applies locally between each pair of fluid-solid neighbours. Therefore according to the present IBM implementation, the boundary is located literally at the interface of the fluid and solid cells, aligned with the cell edges, as shown in Fig. 5. This aspect of the IBM is conceptually unchanged from uDALES v1.0, though was implemented from scratch in uDALES v2.0 in order to accommodate the 2D domain decomposition and the triangulated surface geometry representation.

An alternative approach would be to enforce that the value of a variable on the boundary, as determined by interpolating the values at nearby points, satisfies the boundary condition exactly. For example, one method would be to set the values at solid boundary points such that when interpolated across the boundary in the facet normal direction to an image point inside the fluid region, the value or flux on the boundary itself satisfies the desired flow condition (*e.g* Majumdar et al., 2020). This approach is often used with DNS since the flow is fully-resolved, which is not always the case for LES. Another inherent challenge with this approach is that it is more complicated to ensure global conservation. In the current approach, this is guaranteed since local conservation is ensured by enforcing zero flux between each fluid-solid pair. In the alternative approach, the zero flux condition is at the boundary itself in an interpolated sense so local conservation no longer necessarily holds between fluid-solid pairs, but global conservation would still need to be enforced.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

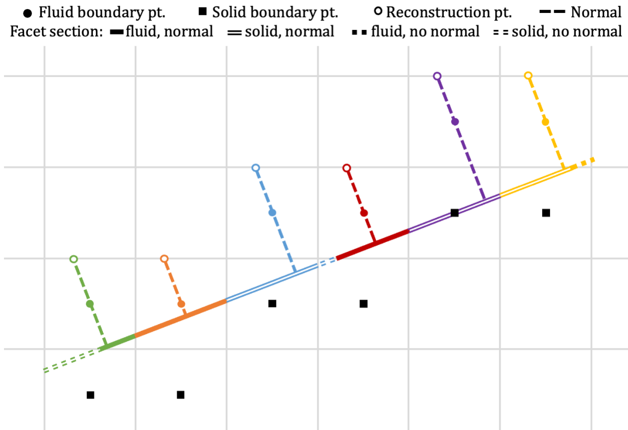

Figure 7. A 2D facet divided into sections. Fluid boundary points (and reconstruction points) are coloured uniquely. Facet sections are coloured by their corresponding fluid boundary point, and distinguished by whether they are in a fluid or solid cell, and whether a normal vector exists between the section and fluid boundary point.

3.3.2 Surface fluxes

The IBM in uDALES uses wall functions to determine surface fluxes of momentum, temperature, and humidity at the fluid boundary points. Given the IBM is conservative, these fluxes are the only mechanism by which the surface exerts friction drag and exchanges heat and moisture with the fluid. The novelty of uDALES v2.0 is in the treatment of these fluxes given the surface is not aligned with the grid. Another conservation principle that motivates the approach is that for the surface energy balance model, the total heat flux into the fluid is equal to the total heat flux out of the surface. Whilst this principle is not relevant for momentum flux, the same method is also applied for that calculation.

While many IBMs with parametrised fluxes are effectively from the perspective of the fluid boundary points (Ma and Liu, 2017; Bao et al., 2018), the approach described here is based on considering the effect of each and every part of the surface on the fluid. As depicted in Fig. 6, the surface is divided cell-wise (for each grid) into facet sections, such that each section lies in only one cell. Note that this cell can be fluid or solid - in either case, the flux it imparts to the fluid must be accounted for. In this approach, each section imparts flux to a single fluid boundary point, which means it is necessary to identify the fluid boundary point that is most appropriate to experience the flux. This process is illustrated in Figure 7. Note that the wall functions are defined in a wall-normal direction, and in this approach, the relevant 'wall' is the facet section. Therefore, in many cases, the chosen fluid boundary point for each facet section is the one where there exists a vector from the facet section to the fluid boundary point in the direction of the facet normal. However, there also may be some facet sections that do not have a fluid boundary point lying on a vector in the direction of the facet normal. For such facet sections, the appropriate fluid boundary point (to give flux) is chosen to be the point which maximises $\cos\theta/d$, where θ is the angle between the facet normal and the surface-point vector, and d is the minimum distance between the facet section and the point. This minimises the angle

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

between the surface normal and the surface-point vector as well as the distance to the point, because a closer point feels the effect of the surface to a greater extent.

For each facet section, the flux to the appropriate fluid boundary point is calculated according to the wall functions using the properties of the facet that the section makes up, as well as the velocity and scalar fields, as discussed in Sect. 2.2. The exact location that fields are evaluated at can optionally either be at the fluid boundary point itself, or at a reconstruction point further from the surface, located at the boundary with the next adjacent cell in the normal direction. The reconstruction approach is similar to that of Ma and Liu (2017); further detail is provided in Appendix B. Whether using reconstruction or not, a local orthonormal coordinate system is defined using the facet normal $\hat{\boldsymbol{n}}$, the spanwise direction $\hat{\boldsymbol{p}} = (\hat{\boldsymbol{n}} \times \boldsymbol{u})/||\hat{\boldsymbol{n}} \times \boldsymbol{u}||$, where $\boldsymbol{u}$ is the velocity, and the streamwise direction $\hat{\boldsymbol{t}} = \hat{\boldsymbol{p}} \times \hat{\boldsymbol{n}}$. The streamwise velocity component $\boldsymbol{u} \cdot \hat{\boldsymbol{t}}$ corresponds to the velocity u_a used in the wall functions.

The surface stress (calculated using Eq. (3)) is a tensor that must be transformed back to the corresponding Cartesian direction. This transformation is simplified because in the local basis, the stress is entirely parallel to $\hat{\boldsymbol{t}}$ and perpendicular to $\hat{\boldsymbol{n}}$. The global basis is denoted by $\boldsymbol{e}_i$, where $i = 1,2,3$ corresponds to $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, $\hat{\boldsymbol{z}}$ respectively, and the local basis by $\boldsymbol{e}'_i$, where $i = 1,2,3$ corresponds to $\hat{\boldsymbol{t}}$, $\hat{\boldsymbol{p}}$, and $\hat{\boldsymbol{n}}$ respectively. The local stress tensor $\boldsymbol{\tau}'$ is therefore defined by $\tau'_{13} = \tau_\mathrm{w}$ and $\tau'_{ij} = 0$ otherwise. The transformation to the global basis is given by:

$$\tau_{ij} = \sum_l \sum_m (\boldsymbol{e}_i \cdot \boldsymbol{e}'_l)(\boldsymbol{e}_j \cdot \boldsymbol{e}'_m)\tau'_{lm}. \tag{8}$$

The total stress in direction i is equal to $||\boldsymbol{\tau} \cdot \boldsymbol{e_i}||$, and the volumetric sink of momentum at the fluid boundary point $S_{u_i} = ||\boldsymbol{\tau} \cdot \boldsymbol{e_i}||a/(\rho V)$, where a is the facet section area and V is the cell volume.

The potential temperature and specific humidity fluxes are calculated using Eq. (4) and Eq. (5) respectively, and correspond to source/sink terms $S_{\theta_1} = u_\tau \theta_\tau a/V$ and $S_{q_\mathrm{t}} = u_\tau q_\tau a/V$. If the surface energy balance model is in use, the sensible and latent heat flux for facet m is given by:

$$H_m = \frac{\rho c_p}{A_m} \sum_{n \in C_m} a_n (u_\tau \theta_\tau)_n, \quad E_m = \frac{\rho L_v}{A_m} \sum_{n \in C_m} a_n (u_\tau q_\tau)_n. \tag{9}$$

Here, C_m is the set of facet sections comprising facet m, and $A_m = \sum_{n \in C_m} a_n$ is the facet area. Given this formulation, global conservation with the surface energy balance is guaranteed. This is because each section is accounted for exactly once (and the sum of their area equals the total surface area), and for each section the fluid cell and facet experience an equivalent flux determined by the section area.

3.4 Shortwave radiation algorithm

In uDALES, the net shortwave radiation is the sum of a direct solar, diffuse sky, and diffuse reflected contributions (Suter et al., 2022). The position of the sun is given in terms of the solar azimuth (Ω) and zenith (Θ). For a coordinate system defined such that the x-direction is at an angle Ω_x clockwise from north, letting $\Omega' = \Omega - \Omega_x$, the vector to the sun $\hat{s}$ (referred to as the sun

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

vector) is defined by:

$$\hat{\boldsymbol{s}} = \begin{bmatrix} \sin\Theta\cos\Omega' \\ -\sin\Theta\sin\Omega' \\ \cos\Theta \end{bmatrix} \tag{10}$$

305 For each facet in uDALES v1.0, the direct solar radiation S is calculated according to $S = I\hat{\boldsymbol{n}}\cdot\hat{\boldsymbol{s}}$, where I is the direct normal irradiance (a model parameter) and $\hat{\boldsymbol{n}}$ is the facet normal. Rays are projected from the facet corners and centre in the direction of the sun vector to check for intersection with any other facet. If so, then shading occurs, and S is reduced by a fraction for each of these points for which there is an intersection. This is a crude approximation for quantifying the radiation on shaded facets, and scales poorly ($\mathcal{O}(N^2)$, where N is the number of facets).

310 The method used to calculate direct solar radiation in uDALES v2.0 is designed to address these limitations. It involves projecting the facets onto a plane whose normal is parallel to the sun vector. To define a point on this plane p_0, take the centre of the geometry $\boldsymbol{p}_g = [L_x/2 \quad L_y/2 \quad 0]^\top$ and project some reasonable distance L in the direction of the sun vector: $\boldsymbol{p}_0 = \boldsymbol{p}_g + L\hat{\boldsymbol{s}}$. To define an orthonormal pair of vectors spanning the plane, let $\hat{\boldsymbol{q}} = [-\hat{s}_2 \quad \hat{s}_1 \quad 0]^\top$ and $\hat{\boldsymbol{r}} = \hat{\boldsymbol{q}} \times \hat{\boldsymbol{s}}$. Given a point $\boldsymbol{p}$, its projected position onto the plane is equal to $\boldsymbol{p} - \mu\hat{\boldsymbol{s}}$ for some unknown μ, which is the distance between the point
315 and the plane. The projected position can also be written as $\boldsymbol{p}_0 + \xi\hat{\boldsymbol{q}} + \eta\hat{\boldsymbol{r}}$ for some unknown ξ and η, which can be thought of local coordinates with respect to the origin $\boldsymbol{p}_0$. Equating these yields the system of equations for the unknowns:

$$\boldsymbol{p} - \boldsymbol{p}_0 = \begin{bmatrix} \hat{\boldsymbol{q}} & \hat{\boldsymbol{r}} & \hat{\boldsymbol{s}} \end{bmatrix} \begin{bmatrix} \xi \\ \eta \\ \mu \end{bmatrix} \tag{11}$$

Solving this system for all the vertices making up the triangulated surface yields the projected position of all the facets in terms of ξ and η. The projected facets potentially overlap, which correspond to one blocking the path of radiation to the
320 other, *i.e.* shading. The cell centres are therefore also projected so that μ can be used to sort the facets by distance from the plane. The plane is discretised into pixels, which are classified according to whether they are inside or outside a projected facet using a modified version of the MATLAB function 'poly2mask' (that has also been implemented in Fortran for improved performance). This classification is carried out sequentially for all the facets in order of decreasing distance to the plane so that in the case of overlaps, the pixels are recorded as belonging to the closer facet. The result is effectively an image of the facets
325 from the sun's point of view, with pixels labelled according to which facet they correspond to. The unshaded projected area (A') of each facet is calculated by summing the pixels labelled by it (and scaling the result by the discretisation resolution). Finally, denoting the actual facet area by A, then $S = IA'/A$, which is equal to $I\hat{\boldsymbol{n}}\cdot\hat{\boldsymbol{s}}$ for unshaded facets. This formulation accurately captures the radiation on shaded facets, and scales faster than $\mathcal{O}(N^2)$ because there are no pairwise facet calculations.

The net shortwave radiation, including reflections, is obtained via the same iterative procedure as described in Suter et al.
330 (2022). View factors are calculated using the open source program View3D (DeGraw and Walton, 2018), which is written in the C programming language and supports triangular facets. Note that in uDALES v1.0, view factors are calculated using

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

bespoke routines for rectangular facets implemented in MATLAB (Suter, 2018); this is slower to run than View3D in practice, and would be challenging to modify to support triangular facets.

3.5 Periodic driver simulations

Driver simulations can be used to provide the time-dependent boundary conditions required to perform inflow-outflow simulations with LES (Suter et al., 2022; Grylls et al., 2019; Lim et al., 2022). They typically use periodic lateral boundary conditions, and their advantage is that the turbulence generated is real, as opposed to the more popular method of generating synthetic turbulence (*e.g.* Xie and Castro, 2008). However, the streamwise and spanwise extent of the driver simulation is typically limited, which can lead to the development of streamwise structures that span the entire domain and become permanent features of the average flow-field. These so-called superstructures are observed in reality (Hutchins and Marusic, 2007) but are exacerbated by the limited domain size used in driver simulations. This is undesirable because it results in spanwise variations of Reynolds-averaged quantities.

uDALES v1.0 simulations suffer from artificial superstructures as there is no functionality for counteracting their formation. In constrast, uDALES v2.0 has a technique designed to avoid them by applying an advection forcing term $\mathcal{F}_i = V(x,z)\frac{\partial u_i}{\partial y}$. Assuming mean flow in the x-direction only, the effect of the velocity $V(x,z)$ is to shift the flow by a spanwise displacement Δ_y over a streamwise distance Δ_x, *i.e.*

$$\Delta_y = \frac{1}{\langle u \rangle}\int_0^{\Delta_x} V(x,z)dx \tag{12}$$

where $\langle u \rangle(z)$ is the plane-averaged streamwise velocity. The velocity V can be chosen arbitrarily, but in order to avoid discontinuities in the forcing field, we require that $V = 0$ at $x = 0$ and $x = \Delta_x$. Assuming a sinusoidal x-dependence that satisfies these conditions, i.e $V(x,z) = \hat{v}(z)\sin(\frac{\pi x}{\Delta_x})$ for some $\hat{v}(z)$ to be determined, the integral above can be evaluated with result

$$\Delta_y = \frac{2\Delta_x}{\pi}\frac{\hat{v}}{\langle u \rangle}. \tag{13}$$

Therefore, $\hat{v} = \frac{\pi\Delta_y}{2\Delta_x}\langle u \rangle$ which implies that the momentum forcing is given by

$$\mathcal{F}_i = \frac{\pi\Delta_y}{2\Delta_x}\langle u \rangle \sin\left(\frac{\pi x}{\Delta_x}\right)\frac{\partial u_i}{\partial y}, \tag{14}$$

where Δ_x and Δ_y are parameters that can be chosen freely. For the simulations in this text using the method, $\Delta_x = L_x/2$, $\Delta_y = L_y/4$, and the output plane is taken at $x = 0$.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

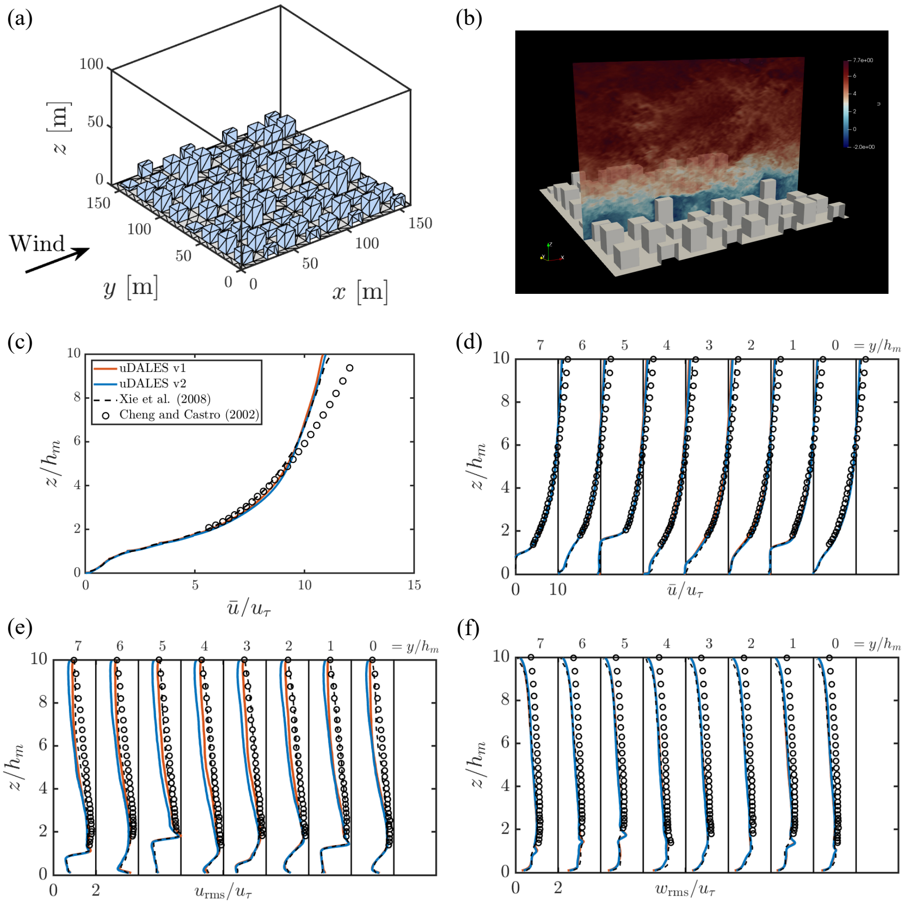

Figure 8. Staggered cubes validation: (a) computational domain; (b) instantaneous streamwise velocity contour at an arbitrary time instant; (c) Spatially averaged mean streamwise velocity profile; (d) mean and (e) r.m.s. streamwise velocity, and (f) r.m.s vertical velocity profiles at different locations behind the third row of buildings. Legend shown in (c) also true for all the panels.

4 Validation

4.1 Neutral flow over staggered array of cubes

In this case, uDALES v1.0 and v2.0 are set up following a numerical simulation by Xie et al. (2008), which compares against an experiment by Cheng and Castro (2002). The set-up in uDALES is full-scale, *i.e.* the dimensions are 1000 times larger than the experiment and numerical simulations presented in the mentioned literature.

The case consists of neutral flow around a staggered array of cubes with width and mean height equal to $h_m = 10$ m (Figs. 8a&b). Periodic lateral boundary conditions are used, with free-slip at the top. The flow is forced by a constant kinematic pressure gradient $\mathcal{F}_u = 4.1912 \times 10^{-3}$ ms^{-2}. The domain size is $L_x \times L_y \times L_z = 16h_m \times 16h_m \times 10h_m$ (see Figs. 8a&b), and the number of grid points is $N_x \times N_y \times N_z = 256 \times 256 \times 256$. The initial velocity profile is uniform; the value is equal

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

to the vertically averaged profile of Xie et al. (2008), which was estimated to be $5.15\ \mathrm{ms}^{-1}$. Note that since the geometry is aligned with the grid, and the boundaries are periodic, this case does not use any new features of uDALES v2.0, so the flow is expected to match that of v1.0.

The spatially-averaged mean streamwise velocity computed by uDALES v1.0 and v2.0 are in good agreement with each other and with the literature; see Fig. 8c. Additionally, the streamwise mean and r.m.s. velocity profiles at different spanwise locations (at $y/h_m = 0,\ 1,\ 2,\ ...,\ 7$) along the centre line between the third and the fourth row of the buildings (*i.e.* at $x/h_m = 6$) are presented in Figs. 8d-f. Note that each profile here is computed as an average of the four available profiles within the computational domain following Xie et al. (2008). There is close agreement with the LES data of Xie et al. (2008), but the agreement with the experimental measurements is less good, particularly higher up in the flow domain. These differences are most likely caused by differences in the inflow conditions above $z/h_m = 5$ (Fig. 8c).

4.2 Neutral cross-ventilation flow for isolated building

This case involves using uDALES v2.0 to simulate an internal-external flow: cross-ventilation of an isolated enclosure (building) with rectangular openings (windows) in both windward and leeward walls (van Hooff et al., 2017). The results are compared with both the RANS and LES simulations of van Hooff et al. (2017) and the wind tunnel measurements of Tominaga and Blocken (2015). Note that uDALES v1.0 cannot handle a geometry of this kind, as it is not possible to have blocks overhanging fluid.

The interior volume of the enclosure has dimensions $D \times W \times H = 0.2 \times 0.2 \times 0.16\ \mathrm{m}^3$; see Figs. 9a&b. In contrast to the simulations in van Hooff et al. (2017) that were carried out using an unstructured mesh, the wall thickness cannot be specified independently of the grid size when using the present IBM. At least 3 grid points must be present inside the solid domain, which translates to a wall thickness of $0.02\ \mathrm{m}$. The size of the windows are $W_0 \times H_0 = 0.092 \times 0.036\ \mathrm{m}^2$, and are placed $0.062\ \mathrm{m}$ from the bottom and $0.054\ \mathrm{m}$ from the inner lateral walls. The computational domain is such that the inlet and outlet boundaries are at distances of $3H$ and $15H$ from the windward and leeward walls of the building respectively. The lateral sides and top boundary of the computational domain are at a distance of $5H$ from the building walls/roof. The number of grid points is $N_x \times N_y \times N_z = 1024 \times 512 \times 128$, and the z-grid is stretched, with the finest grid resolution near the building being approximately $\Delta x = \Delta y = \Delta z = 0.0035\ \mathrm{m}$. Inflow-outflow boundary conditions are used in the x-direction, while the y-direction is periodic and the top is free-slip. A precursor (driver) simulation is used for the inflow to match the fully-developed neutral boundary layer inlet used by van Hooff et al. (2017) such that the streamwise velocity at a height of $H = 0.16$ m equals to the reference velocity $U_H = 4.3\ \mathrm{ms}^{-1}$.

Figure 9c shows a comparison of the streamwise velocity profile computed by uDALES v2.0 with standard k-ϵ RANS (RANS|SKE) and LES results of van Hooff et al. (2017) and the experimental measurements of Tominaga and Blocken (2015), along three different vertical lines at $x/D = 0.25$, 0.5 and 0.75 in the vertical centre plane ($y/W = 0$) of the enclosure shown using dashed lines in Fig. 9b. A similar comparison for the TKE profile is also shown in Fig. 9d. Velocity and TKE along the horizontal centre line ($y/W = 0$, $z/H = 0.5$) are shown in Fig. 9e. All these figures demonstrate good agreement between uDALES v2.0 and those reported in the literature. Furthermore, uDALES v2.0 captures the shape and width of the jet entering

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

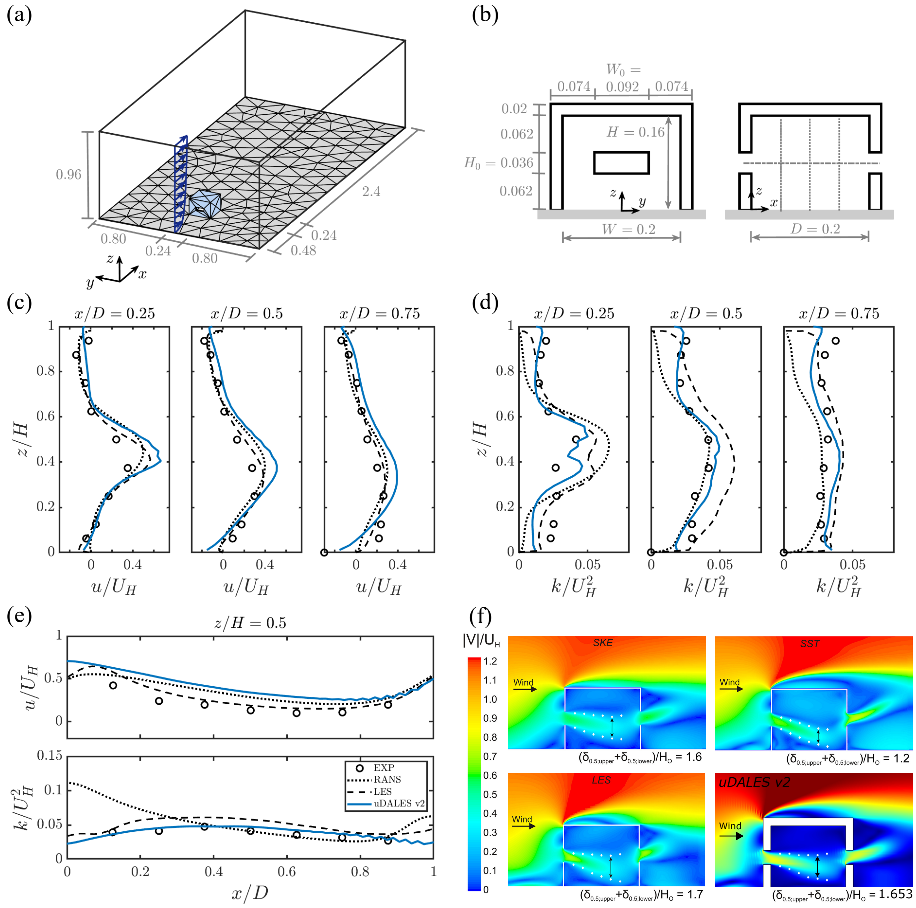

Figure 9. Cross-ventilation case validation (van Hooff et al., 2017): (a) computational domain; (b) front and side view of the building, including windows (all dimensions are in m). (c) Mean streamwise velocity, and (d) mean TKE profiles at streamwise locations inside the building at the vertical centre plane ($y/W = 0$). (e) Mean streamwise velocity and TKE profiles along a horizontal line through the middle of the windows ($y/W = 0$, $z/H = 0.5$). Legend shown in bottom panel of (e) holds for all the panels in all frames. (f) Contours of dimensionless mean velocity magnitude ($|V|/U_H$); SKE, SST and LES panels are reproduced here from van Hooff et al. (2017) with appropriate permission from the publisher.

through the window accurately as shown in terms of the dimensionless mean velocity magnitude ($|V|/U_H$) contour in Fig. 9f. At $x/D = 0.625$, uDALES v2.0 predicts the jet width $(\delta_{0.5;\text{upper}} + \delta_{0.5;\text{lower}})/H_0 = 1.653$ to fall between the jet width prediction by RANS|SKE ($= 1.6$) and LES ($= 1.7$) simulations of van Hooff et al. (2017). Here, $\delta_{0.5}$ indicates the jet half-width - the vertical distance between where the jet velocity magnitude is a local maximum and where it is equal to half the maximum at a particular x-location.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

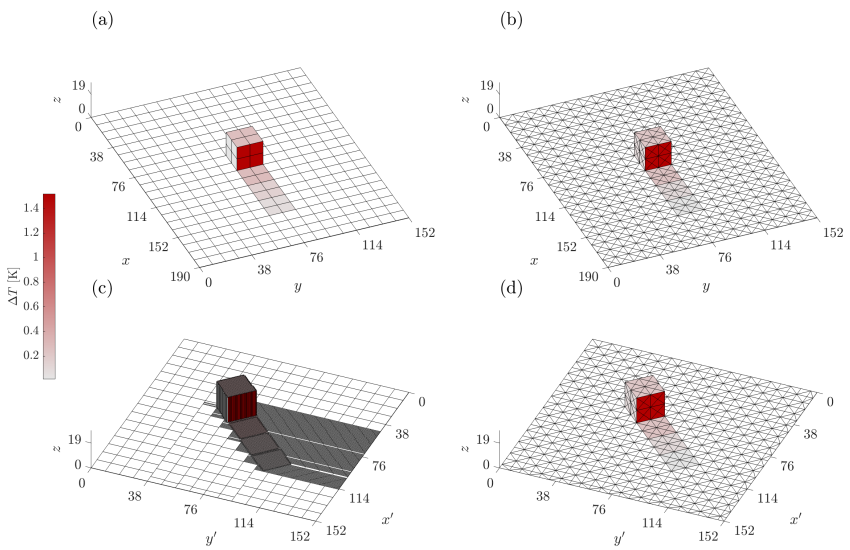

Figure 10. Facets coloured according to temperature difference ΔT: (a) v1.0, original; (b) v2.0, original; (c) v1.0, rotated; (d) v2.0, rotated.

Table 2. Temperature differences applied to facets.

Facet	ΔT [K]
Leeward	1.52
Roof	0.26
Side wall	0.14
Floor 1	0.28
Floor 2	0.14
Floor 3	0.07

4.3 Non-neutral flow over single cube with surfaces at constant temperature

This case is based on an LES study by Boppana et al. (2013), in which the flow is compared against an experiment conducted by Richards et al. (2006). Here both uDALES v1.0 and v2.0 are used, and the simulations are run at full scale, *i.e.* 100 times larger than in the mentioned literature. The geometry consists of a single cube, and a subset of the facets are heated to a constant temperature. In the original case (Figs. 10a&b) the cube is grid-aligned, but in order to test the ability of the IBM to handle non-aligned geometries, the situation in which the cube is rotated with respect to the grid is also simulated (Figs. 10c&d). In these cases, the intention is to maintain the same physical flow over the cube. Fig. 10c shows the effect of using a voxel representation for the geometry - the facets for the uDALES v1.0 rotated case need to be much smaller (and more numerous) in places where the actual geometry does not align with the grid.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

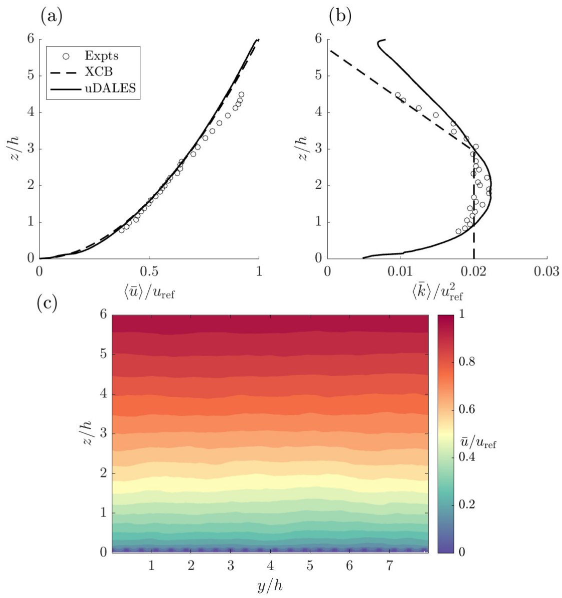

Figure 11. Spanwise-averaged inflow profiles: (a) wind speed, (b) turbulent kinetic energy; (c) spanwise dependence of wind speed. In legend, 'Expts' refers to Richards et al. (2006) and 'XCB' refers to Boppana et al. (2013).

The cube has a side length of $h = 19$ m, and in the original (non-rotated) case the domain size is $L_x \times L_y \times L_z = 10h \times 8h \times 6h$. Note the domain width has been increased from that used by Boppana et al. (2013) so that it matches that of the rotated case, therefore the two cases can use the same precursor simulation. The centre of the cube is located at $x_c = 4.5h$, $y_c = 4h$. For the rotated case, the flow is rotated by an angle of 45°, and the domain size is $L'_x = L'_y = 8h$. The cube is positioned such that the distance from the origin to the cube centre is $4.5h$, *i.e.* the centre is at $x'_c = 4.5h/\sqrt{2}$, $y'_c = 4.5h/\sqrt{2}$. This means that the distance from the cube centre to the outlet is at least $5.5h$, thus making it possible to compare the wake with the original case. The grid sizes of the original and rotated cases are $N_x \times N_y \times N_z = 320 \times 256 \times 192$ and $N'_x \times N'_y \times N'_z = 256 \times 256 \times 192$ respectively, meaning the resolution is 0.59375 m in each direction.

The temperature of the facets are set to match the Richardson number (Ri) of Boppana et al. (2013), namely the temperature differences are 100 times smaller, and are shown in Table 2. The cases corresponding to $Ri = 0$ (neutral) and $Ri = -0.66$ (non-neutral) were run, and the non-neutral results are presented here. The momentum and heat roughness lengths are unknown, as a different wall model is used in Boppana et al. (2013). Here $z_0 = 0.01$ m is used, with $z_{0,\mathrm{h}} = z_0$ as in Louis (1979).

Regarding boundary conditions, the y-direction is periodic and the vertical velocity at the top is allowed to vary as discussed in Sect. 3.2. The x-direction is inflow-outflow, with the inflow provided by a precursor simulation, whereas Boppana et al. (2013) used synthetic turbulence generation. The geometry consists of staggered cubes of side length $h/8$ and plan area fraction $\lambda_p = 0.25$. The domain length is $12h$, with the same width, height, and resolution as the main simulation. The aim of the precursor is to generate mean streamwise velocity and turbulent kinetic energy profiles that match that of Richards et al.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

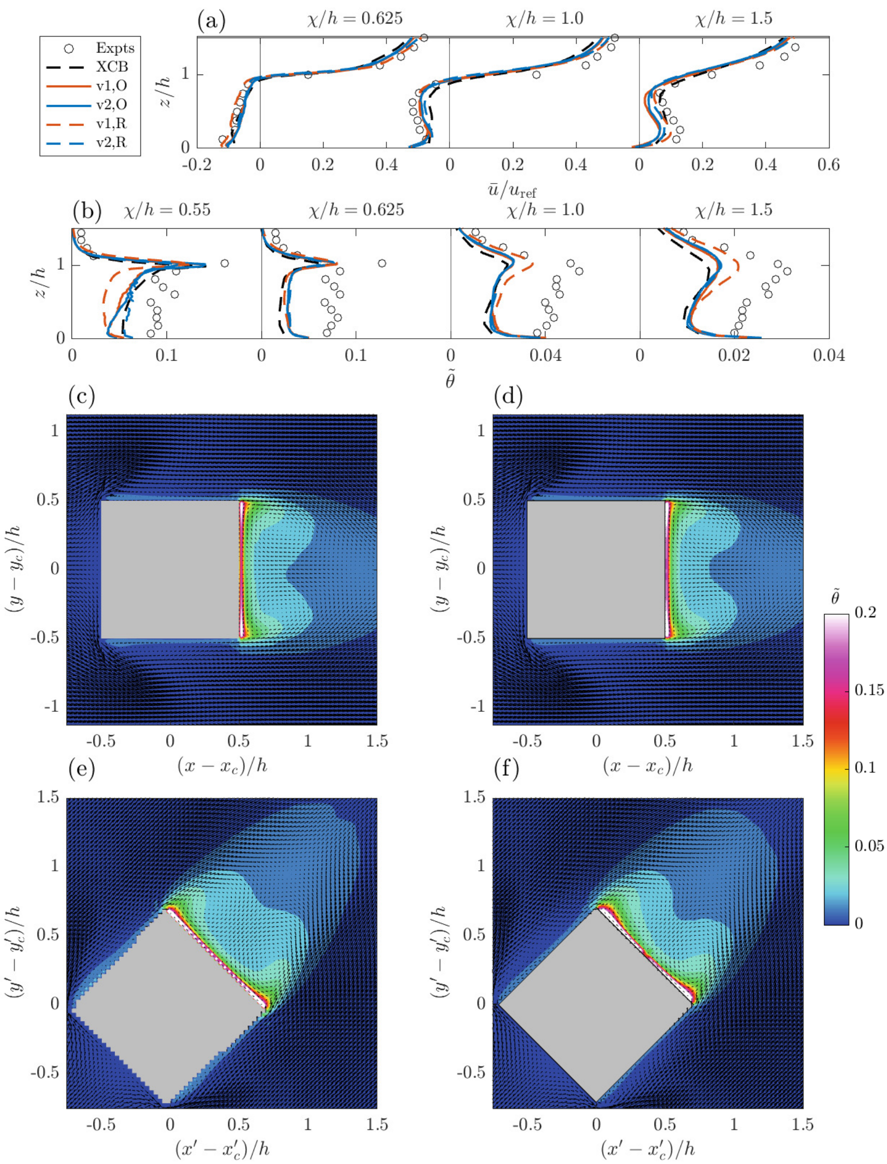

Figure 12. Normalized mean profiles of (a) streamwise velocity $\bar{u}/u_{\mathrm{ref}}$ and (b) potential temperature $\tilde{\theta} = (\bar{\theta} - \theta_{\mathrm{ref}})/(\theta_{\mathrm{wall}} - \theta_{\mathrm{ref}})$ at downstream locations in the cube wake, with streamwise distance from cube centre denoted by χ. In legend, In legend, 'Expts' refers to Richards et al. (2006), 'XCB' refers to Boppana et al. (2013), and 'O' denotes original and 'R' denotes rotated cases for uDALES v1.0 and v2.0. Below: contours of $\tilde{\theta}$ with mean velocity vectors at $z/h = 0.5$ for (c) v1.0, original; (d) v2.0, original; (e) v1.0, rotated; (f) v2.0, rotated.

(2006) and Boppana et al. (2013). In the experiment, the flow is characterised by a power-law profile with exponent $\alpha = 0.52$:

$$u_{\mathrm{exp}}(z) = u_{\mathrm{ref}} \left(\frac{z}{z_{\mathrm{ref}}} \right)^{\alpha}. \tag{15}$$

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

The reference height z_{ref} was taken to be $10\ \mathrm{m}$ and the reference velocity u_{ref} was estimated to be $0.33\ \mathrm{ms}^{-1}$. The initial wind speed is set to this profile, and the plane-averaged velocity is forced towards it by using a nudging term:

$$\mathcal{F}_u(z) = \frac{u_{\mathrm{exp}} - \langle u \rangle}{t_{\mathrm{nudge}}}, \tag{16}$$

with $t_{\mathrm{nudge}} = 60\ \mathrm{s}$. In order to avoid the formation of artificial superstructures, the shifting method described in Sect. 3.5 is used. Figure 11c shows the time-averaged inflow plane velocity; it is reasonably uniform in the y-direction, indicating that the shifting method is successful. Averaging this plane in the y-direction results in the velocity profile shown in Fig. 11a, which matches the power-law profile almost exactly. The turbulent kinetic energy (TKE) is shown in Fig. 11b. The profile broadly agrees with the experimental and numerical values, in that the peak is at around 0.02 between $z/h = 1$ and $z/h = 3$ and decreases above this height.

For the main simulation, Figs. 12c-f shows slices of mean scaled potential temperature halfway up the cube. From this it appears that the v2.0 rotated case agrees better with the original v2.0 case, and indeed to the original v1.0 case, than the v1.0 rotated case does. The slight asymmetry in the wake in the rotated cases is most likely due to the boundary conditions - since the y-direction is periodic, the effective distance to the inflow of the two lateral sides of the cube is not the same. Figures 12a and 12b respectively show profiles of mean normalised streamwise velocity and potential temperature downstream of the cube on the wake centre line. The velocity profiles are generally in agreement with the XCB simulations, with greater discrepancy below cube height as the distance from the wall increases. This shows that the models capture the wake similarly in the near field. The temperature profile at $x/h = 0.55$ is very close to the leeward face so the temperature gradient in the streamwise direction is large, making the comparison at this point less precise. Other than at this location, uDALES generally agrees well with the simulations of Boppana et al. (2013), with v2.0 matching more closely than v1.0 in the rotated case. Note that there is not good agreement with the experiments; this is discussed in detail in Boppana et al. (2013).

4.4 Non-neutral flow over single cube with surface temperatures modelled using surface energy balance scheme

Similarly to Sect. 4.3, this study involves simulating the flow around the original (non-rotated) and rotated geometries using uDALES v1.0 and v2.0, this time allowing the facet temperatures to evolve according to the surface energy balance model. This is in order to both validate the implementation in uDALES v2.0 against v1.0 in the case where the two models should agree, and to evaluate how the rotated cases compare to the original cases.

The boundary conditions, including the precursor-defined inflow, are the same as in Sect. 4.3. The initial conditions are also unchanged apart from the facet temperatures. The simulation parameters pertaining to the surface energy balance are given in Table 3. Note the relatively low facet volumetric heat capacity used, which allows the simulations to reach a steady state faster. The steady state reached does not depend on the volumetric heat capacity, and it is only the steady state (and not transient behaviour) that is of interest here. The solar position is chosen such that of the cube faces, only the leeward and roof (should) receive direct solar radiation. This means these faces should be heated more than the others, so the flow is expected to be qualitatively similar to Sect. 4.3, meaning any considerations discussed there also apply here.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

Table 3. Surface energy balance parameters.

Property	Value
Solar azimuth (Ω)	90°
Solar zenith (Θ)	45°
Direct normal solar irradiance (I)	800 $\mathrm{Wm^{-2}}$
Diffuse sky shortwave irradiance (D_{sky})	80 $\mathrm{Wm^{-2}}$
Diffuse sky longwave irradiance (L_{sky}	336 $\mathrm{Wm^{-2}}$
Initial exterior facet temperature (T_{s})	293 K
Interior facet temperature (T_B)	293 K
Facet albedo (α)	0.5
Facet emissivity (ϵ)	0.85
Facet volumetric heat capacity (C)	0.1 $\mathrm{MJm^{-3}K^{-1}}$
Facet conductivity (λ)	1 $\mathrm{Wm^{-1}K^{-1}}$
Facet thickness (D)	0.4 m
Number of facet layers	5

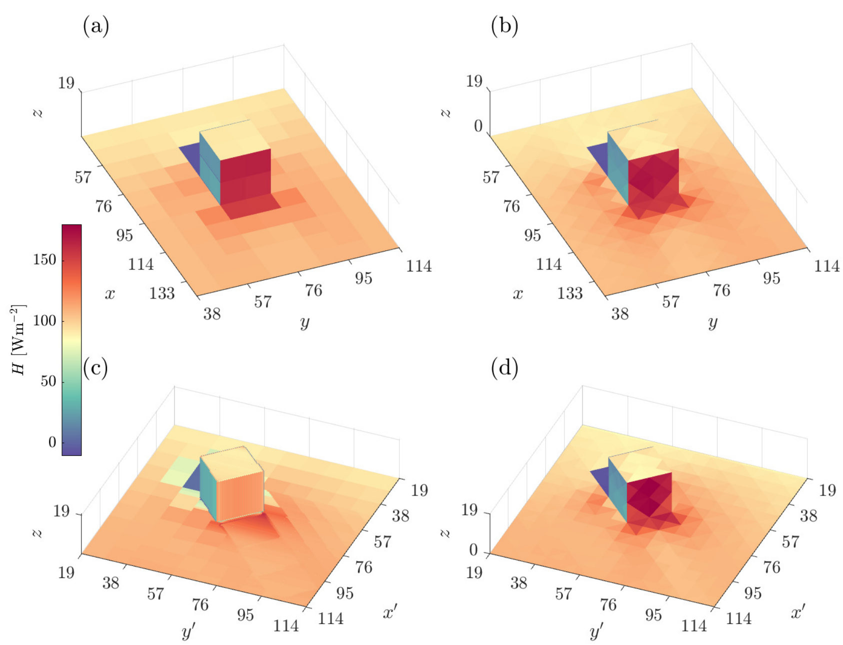

Figure 13. Facets coloured according to sensible heat flux H for (a) v1.0, original; (b) v2.0, original; (c) v1.0, rotated; (d) v2.0, rotated.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

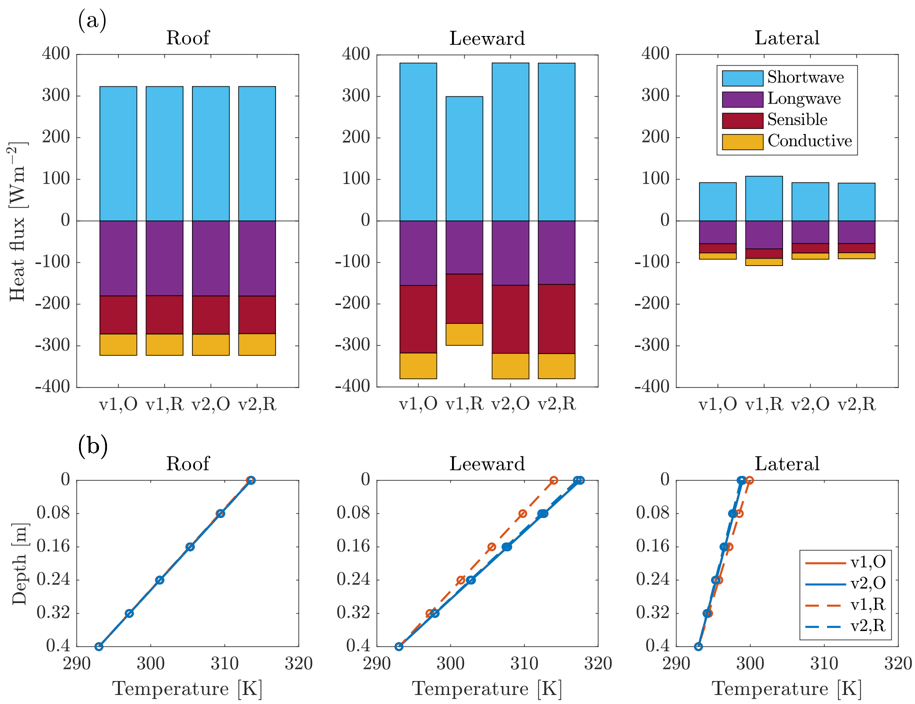

Figure 14. Steady state (a) surface energy balance terms (heat fluxes), (b) internal facet temperature for roof, leeward, and lateral facets.

Figure 13 shows the steady state sensible heat flux (H) on the facets. It is clear that the v1.0 rotated case differs both qualitatively and quantitively. For example on the leeward face, H does not reach as high a value, and the shaded region (with low H) is not captured accurately. This latter observation relates to the fact that how exactly the geometry is discretised into facets is significant. In the shaded region, the facets are large in both dimensions, so the result would likely be improved by using much smaller facets. The facets making up the ground just downstream of the leeward face are very thin in the x'-direction, but large in the y'-direction, meaning the structure (namely H decreasing away from the cube) is not symmetric along the cube centre line like the other cases. This discussion indicates that the geometry specification for v1.0 is important, and this is ultimately because it is dependent on the underlying computational grid. Although the v2.0 rotated case is better, there is still some discrepancy, chiefly that the structure is not as symmetric along the cube centre line when compared to the original case. As discussed in Sect. 4.3, it is most likely that this discrepancy is due to the boundary conditions - the x'-direction is inflow-outflow and the y'-direction is periodic, which allows the turbulent inflow to reach both sides of the cube but having effectively travelled different distances to each. Another factor in this study is that the incoming flow (with uniform temperature) will be affected by the warmer ground facets, causing thermal inlet effects and variation in the x/x'-direction. This is unavoidable when using the precursor simulation, but it seems from Fig. 13 that this effect has diminished sufficiently before reaching the cube so that the four cases are comparable.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

Figure 14 show the steady state surface heat fluxes and internal temperatures for the roof, leeward, and lateral faces of the cube. For the roof, all cases agree, as the effect of rotation is negligible. For the leeward and lateral sides, the v1.0 rotated case differs from the non-rotated cases to a much greater extent than the v2.0 rotated case; for the leeward side, the shortwave flux is around 21% less. These differences are primarily caused by errors in the radiation, which is directly affected by the rotation. Note that the average of the direct shortwave radiation over the leeward face is in fact correct in the v1.0 rotated case. This is because although each facet receives less than the other cases as their normal is not aligned with the vector to the sun, the wall's surface area is larger by the same factor (equal to $\sqrt{2}$ for this choice of solar position and rotation angle of $45°$). This result would also hold for other angles as long as there is no internal shading on the face, which clearly introduces errors in the direct contribution. The error is therefore due to the other contributions to the net shortwave radiation. The reflected contribution caused by neighbouring facets facing each other is incorrect because it obviously should be zero for a flat surface. However, the net shortwave on the leeward face is in fact less than expected, so this error must be outweighed by the error in the diffuse sky radiation. If neighbouring facets can 'see' each other, their sky view factor is reduced, hence reducing this contribution. This discussion concerning contributions due to neighbour reflections and from the sky also holds for the lateral sides, though unlike the leeward face, the net shortwave on these sides is larger than expected. This is because the sides should not have a direct contribution as they are not visible to the sun, but this is not in fact true for the v1.0 rotated case. Around half of the facets are oriented such that they are visible to the sun, and these are not completely shaded by the neighbour that they face.

Regarding the other terms in the surface energy balance, the leeward and side faces are expected to have reduced incoming longwave due to the same reasons given for the shortwave. The outgoing longwave, as well as the the other fluxes, are largely determined by the surface temperature. The leeward face has less available energy due to the reduced shortwave (as well as longwave), meaning the temperature is lower, which also acts to reduce the outgoing longwave, sensible heat, and conductive fluxes. On the other hand, the side faces have more available energy due to the increased shortwave (despite the reduced longwave), and so the temperature is higher, which acts to increase the other fluxes, though the effects are small. These results demonstrate that uDALES v2.0 is more accurate than v1.0 in capturing the radiative terms in the surface energy balance when the geometry not aligned with the grid, which in turn means the other terms are more accurate too.

5 Conclusion

In this work, an upgrade to the uDALES framework has been presented that addresses several key limitations with the previous version. The limit of parallelisation of the code has been increased by implementing a two-dimensional domain decomposition using the external library 2DECOMP&FFT, which was shown in Sect. 3.1. The pressure solver now makes use of the FFT algorithm for inflow-outflow boundary conditions, as was described in Sect. 3.2. Finally, the geometry representation has been improved in that it can be specified independently of the computational grid (using the widely-used STL file format), and the IBM has been developed significantly to accommodate this, as described in Sect. 3.3. In particular, the novel approach taken to apply parametrised surface fluxes ensures that heat is conserved with the surface energy balance scheme. uDALES v2.0 has been compared against uDALES v1.0 and other physical and numerical simulations of neutral and non-neutral cases in order

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

to validate the development. The neutral cases confirm that v2.0 produces very similar results to v1.0 in canonical urban cases (Sect. 4.1), and demonstrate its utility for coupled internal-external flows that are not possible to study using v1.0 (Sect. 4.2). The non-neutral cases (Sect. 4.3 and Sect. 4.4) were designed to evaluate both versions in situations where the geometry is not aligned with the grid, and the results demonstrate that v2.0 consistently exhibits better accuracy than v1.0.

This upgrade makes uDALES v2.0 a far more practical tool than v1.0. The code scales well for realistic problem sizes up to a large number of processors on ARCHER2, meaning simulations can be run far more quickly. This enables the study of larger urban areas, and makes longer simulations (*e.g.* a diurnal cycle) more feasible. The changes to the pressure solver mean that simulations can be performed just as quickly for inflow-outflow situations as for periodic, which was not the case for uDALES v1.0. The fact that non-grid-aligned obstacles can be modelled more accurately means that uDALES v2.0 can be used reliably in situations involving realistic urban geometries, which is important for wind loading and microclimate studies for example. As a result, the capabilities of the uDALES framework, and so of Cartesian LES codes more generally, are now positioned at the cutting-edge of modelling techniques for urban environments, and provide an attractive alternative to using LES codes that require body-fitted mesh generation.

The work presented here provides a firm foundation for future development. The new IBM approach is flexible to modification, such as incorporating interpolation in the way described in Sect. 3.3.1, and altering the exact form of the wall functions. Given the code now uses low-level routines from the 2DECOMP&FFT library, which is used by many other frameworks, any development of these other frameworks can potentially be adopted in uDALES. For example, the library supports parallel I/O via either MPI-IO or ADIOS2 backends (Rolfo et al., 2023). The ADIOS2 backend provides access to a streaming interface which can be used to implement in-situ analysis, reducing the amount of data written to disk (Bartholomew et al., 2023). Also, support for NVIDIA GPUs has recently been added to 2DECOMP&FFT with support for the other vendors planned (Rolfo et al., 2023), which will allow future efforts to port uDALES to GPUs to focus on the uDALES-specific components only. The use of GPUs also suits a radiative transfer scheme that is based on ray tracing, which is being investigated in part as a way of integrating trees into the surface energy balance scheme. In general, the improved low-level functionality of the codebase prepares uDALES for exascale HPC systems, which means that it will be increasingly feasible to perform ever higher resolution simulations of larger, more realistic urban areas, which is crucial to furthering understanding of urban climate processes.

Code and data availability. The uDALES codebase and user manual is available on GitHub at https://github.com/uDALES/u-dales, and on Zenodo at https://zenodo.org/records/10671714. The user manual includes guidance on the 2D domain decomposition, and on running the pre-processing routines required to generate the necessary inputs for the immersed boundary method and the surface energy balance model.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

Appendix A: Pressure solver

Following Ferziger et al. (2019), the governing equations for velocity can be written (using an Euler time integration scheme for simplicity):

$$\frac{\boldsymbol{u}^{n+1}-\boldsymbol{u}^{n}}{\Delta t} = -\nabla p^{n+1} + \boldsymbol{r}^{n}. \tag{A1}$$

Here $\boldsymbol{r}^n$ includes all processes other than pressure. The pressure p^{n+1} is determined using a two-step scheme which involves introducing the pressure correction $\pi^n = p^{n+1} - p^n$, and a predicted velocity $\mathbf{u}^*$ defined according to:

$$\frac{\boldsymbol{u}^{*}-\boldsymbol{u}^{n}}{\Delta t} = -\nabla p^{n} + \boldsymbol{r}^{n}. \tag{A2}$$

Substitution into Eq. (A1) yields:

$$\frac{\boldsymbol{u}^{n+1}-\boldsymbol{u}^{*}}{\Delta t} = -\nabla \pi^{n} \tag{A3}$$

Requiring incompressibility ($\nabla \cdot \boldsymbol{u}^{n+1} = 0$) results in a Poisson equation for the pressure correction:

$$\nabla^2 \pi^n = \frac{\nabla \cdot \boldsymbol{u}^*}{\Delta t} \tag{A4}$$

Discretising in space and denoting the r.h.s of Eq. (A4) as f:

$$\frac{\pi_{i+1,j,k} - 2\pi_{i,j,k} + \pi_{i-1,j,k}}{\Delta x^2} + \frac{\pi_{i,j+1,k} - 2\pi_{i,j,k} + \pi_{i,j-1,k}}{\Delta y^2} + \frac{\frac{\pi_{i,j,k+1}-\pi_{i,j,k}}{\Delta z_{k+1/2}} - \frac{\pi_{i,j,k}-\pi_{i,j,k-1}}{\Delta z_{k-1/2}}}{\Delta z_k} = f_{i,j,k} \tag{A5}$$

Next, take the DFT/DCT transform of both sides with respect to x then y, denote the modes using n and m respectively, and use hat notation to denote a transformed quantity.

$$a_k \hat{\pi}_{n,m,k-1} + b_{n,m,k}\hat{\pi}_{n,m,k} + c_k \hat{\pi}_{n,m,k+1} = \hat{f}_{n,m,k} \tag{A6}$$

Here $a_k = \frac{1}{\Delta z_k \Delta z_{k-1/2}}$, $c_k = \frac{1}{\Delta z_k \Delta z_{k+1/2}}$, and $b_{n,m,k} = -(a_k + c_k) + \lambda_n + \lambda_m$, where λ_n and λ_m are as follows:

$$\lambda_n = \begin{cases} \frac{2}{\Delta x^2}(\cos\frac{2\pi n}{N_x} - 1) & \text{DFT} \\ \frac{2}{\Delta x^2}(\cos\frac{\pi n}{N_x} - 1) & \text{DCT} \end{cases} \qquad n = 0, ..., N_x - 1 \tag{A7}$$

$$\lambda_m = \begin{cases} \frac{2}{\Delta y^2}(\cos\frac{2\pi m}{N_y} - 1) & \text{DFT} \\ \frac{2}{\Delta y^2}(\cos\frac{\pi m}{N_y} - 1) & \text{DCT} \end{cases} \qquad m = 0, ..., N_y - 1 \tag{A8}$$

For each mode, this is a tridiagonal system with $k = 0, .., N_z + 1$ which is solved using Gaussian elimination. In uDALES, boundary conditions are not inserted into $\hat{f}_0$ and $\hat{f}_{N_z+1}$ in order to explicitly solve for the ghost points $\hat{\pi}_0$ and $\hat{\pi}_{N_z+1}$. Instead,

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

the equations for $k = 1$ and $k = N_z$ are modified in light of the top and bottom boundary conditions, which is numerically equivalent. Specifically, the bottom boundary condition is Neumann for all modes, *i.e.* $\hat{\pi}_{n,m,0} = \hat{\pi}_{n,m,1} \;\forall\; n, m$. This means that the $k = 1$ equation becomes:

$$(a_1 + b_{n,m,1})\hat{\pi}_{n,m,1} + c_1\hat{\pi}_{n,m,2} = \hat{f}_{n,m,1}. \tag{A9}$$

The top is Neumann for all modes other than $n = m = 0$ (the 'zero mode') in which case it is Dirichlet, *i.e.* $\hat{\pi}_{N_z+1} = -\hat{\pi}_{N_z}$. Applying the Neumann condition results in:

$$a_{N_z-1}\hat{\pi}_{n,m,N_z-1} + (b_{n,m,N_z} + c_{N_z})\hat{\pi}_{n,m,N_z} = \hat{f}_{n,m,N_z}, \tag{A10}$$

and the Dirichlet condition for the zero mode:

$$a_{N_z-1}\hat{\pi}_{0,0,N_z-1} + (b_{0,0,N_z} - c_{N_z})\hat{\pi}_{0,0,N_z} = \hat{f}_{0,0,N_z}. \tag{A11}$$

Numerically, this avoids the problem being ill-posed: since $b_{0,0,k} = 0$, the matrix describing the system of equations (A6) is singular if both the first and last rows are the same, which would be the case if both correspond to a Neumann condition. Setting a Dirichlet condition instead at the top is also a physically reasonable choice because the zero mode is equal to the plane-averaged pressure, which can reasonably expected to be zero at the top of the domain. For inflow-outflow simulations, the pressure gradient at the top is accounted for in the boundary condition for vertical velocity $w \equiv u_z$. Given that the Dirichlet boundary condition for plane-averaged pressure is $\langle\pi\rangle_{N_z+1} = -\langle\pi\rangle_{N_z}$, the vertical velocity satisfies:

$$\frac{w^*_{i,j,N_z+1} - w^n_{i,j,N_z+1}}{\Delta t} = \frac{2\langle p^n\rangle_{N_z}}{\Delta z_{N_z+1}} \quad \forall i,j \tag{A12}$$

$$\frac{w^{n+1}_{i,j,N_z+1} - w^*_{i,j,N_z+1}}{\Delta t} = \frac{2\langle \pi^n\rangle_{N_z}}{\Delta z_{N_z+1}} \quad \forall i,j \tag{A13}$$

Appendix B: Reconstruction

When using reconstruction, the flux is calculated at a point further from the facet in the facet normal direction, where the stencil should contain only fluid points. Once the flux has been calculated at this reconstruction point it can be applied to the fluid boundary point because by definition the region in which the wall functions are valid is a constant flux layer. The location is found by projecting a ray in the normal direction from the fluid boundary point point until it hits one of the edges of the cell. A common approach is to project a fixed distance into the adjacent cell, but this would mean it is possible that the surrounding points are at least two indices away, which is problematic if the flux point is on the edge of a decomposition pencil. This approach guarantees that all the points used in the stencil are known to the current rank.

Trilinear interpolation is used to determine the value of flow variables at the reconstruction point. To calculate a variable φ at a point (x, y, z) somewhere in cell i, j, k of a given grid, *i.e.* $x_i \leq x \leq x_{i+1}$, $y_j \leq y \leq y_{j+1}$, and $z_k \leq z \leq z_{k+1}$, let $x_d =$

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

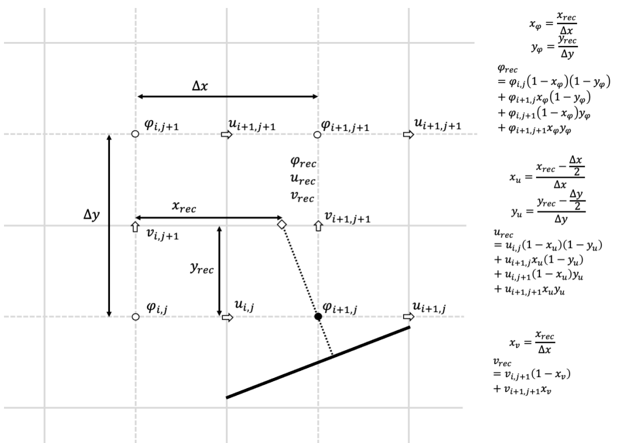

Figure B1. Reconstruction of flow variables in 2D at the point $\varphi_{i+1,j}$ using a bilinear interpolation scheme. A facet section is shown in solid black, the c-grid cell edges are solid grey, and the u- and v-grid cell edges are dotted grey (where they are staggered with respect to the c-grid). Note that the bilinear interpolation reduces to a linear one for the v-velocity.

$(x - x_i)/(x_{i+1} - x_i) = (x - x_i)/\Delta x$, and similar for y_d and z_d. Then the value is given by:

$$
\begin{aligned}
\varphi(x,y,z) \quad = \quad & \varphi(x_i,y_j,z_k)(1-x_d)(1-y_d)(1-z_d) \\
+ \quad & \varphi(x_{i+1},y_j,z_k)x_d(1-y_d)(1-z_d) \\
+ \quad & \varphi(x_i,y_{j+1},z_k)(1-x_d)y_d(1-z_d) \\
+ \quad & \varphi(x_{i+1},y_{j+1},z_k)x_d y_d(1-z_d) \\
+ \quad & \varphi(x_i,y_j,z_{k+1})(1-x_d)(1-y_d)z_d \\
+ \quad & \varphi(x_{i+1},y_j,z_{k+1})x_d(1-y_d)z_d \\
+ \quad & \varphi(x_i,y_{j+1},z_{k+1})(1-x_d)y_d z_d \\
+ \quad & \varphi(x_{i+1},y_{j+1},z_{k+1})x_d y_d z_d
\end{aligned}
$$

Note that in this case since the reconstruction point is at one of the cell edges, the scheme for variables defined on this edge reduces to a bilinear interpolation. A 2D example for a scalar variable is shown in Fig. B1.

Author contributions. SOO carried out the bulk of the model development including the 2D domain decomposition (Sect. 3.1), the pressure solver (Sect. 3.2), and the geometry representation (3.3); and was involved in the periodic neutral validation (4.1) and the non-neutral validations (Sects. 4.3 and 4.4). DM contributed to model development, particularly with respect to pre-processing, and was involved in the periodic neutral validation (Sect. 4.1) and the internal-external validation (Sect. 4.2). CEW informed the model development and was involved with the surface energy balance validation (Sect. 4.4). PB supervised and contributed to the model development with respect to the use of the 2DECOMP&FFT library, including performing scaling tests (Sect. 3.1). MvR acquired funding and supervised the project as

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

a whole, and conceived of the shortwave radiation algorithm (Sect. 3.4) and the shifting method for periodic simulations (Sect. 3.5). SOO prepared the manuscript with contributions from all co-authors.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. This work was funded under the embedded CSE programme of the ARCHER2 UK National Supercomputing Service (http://www.archer2.ac.uk) (project ARCHER2-eCSE05-3) and the Turbulence at the Exascale project (EP/W026686/1) which is part of the ExCALIBUR HPC programme funded by EPSRC. For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising from this submission.

https://doi.org/10.5194/egusphere-2024-96
Preprint. Discussion started: 20 February 2024

References

Arakawa, A. and Lamb, V. R.: Computational design of the basic dynamical processes of the UCLA general circulation model, General circulation models of the atmosphere, 17, 173–265, 1977.

Baker, C.: Wind engineering—Past, present and future, Journal of Wind Engineering and Industrial Aerodynamics, 95, 843–870, 2007.

Bao, J., Chow, F. K., and Lundquist, K. A.: Large-eddy simulation over complex terrain using an improved immersed boundary method in the Weather Research and Forecasting Model, Monthly Weather Review, 146, 2781–2797, 2018.

Bartholomew, P., Deskos, G., Frantz, R. A., Schuch, F. N., Lamballais, E., and Laizet, S.: Xcompact3D: An open-source framework for solving turbulence problems on a Cartesian mesh, SoftwareX, 12, 100 550, 2020.

Bartholomew, P., Moulinec, C., Weiland, M., and Laizet, S.: Developing in-situ analysis capabilities for pre-Exascale simulations with Xcompact3D, ARCHER2-eCSE03-02, https://www.archer2.ac.uk/ecse/reports/eCSE03-02/, 2023.

Bastin, J.-F., Clark, E., Elliott, T., Hart, S., van den Hoogen, J., Hordijk, I., Ma, H., Majumder, S., Manoli, G., Maschler, J., et al.: Understanding climate change from a global analysis of city analogues, PloS one, 14, e0217 592, 2019.

Boppana, V., Xie, Z.-T., and Castro, I. P.: Large-eddy simulation of heat transfer from a single cube mounted on a very rough wall, Boundary-layer meteorology, 147, 347–368, 2013.

Borazjani, I., Ge, L., and Sotiropoulos, F.: Curvilinear immersed boundary method for simulating fluid structure interaction with complex 3D rigid bodies, Journal of Computational physics, 227, 7587–7620, 2008.

Chen, Q.: Ventilation performance prediction for buildings: A method overview and recent applications, Building and environment, 44, 848–858, 2009.

Cheng, H. and Castro, I. P.: Near wall flow over urban-like roughness, Boundary-Layer Meteorology, 104, 229–259, 2002.

DeGraw, J. and Walton, G.: View3D, Tech. rep., National Renewable Energy Lab.(NREL), Golden, CO (United States), 2018.

Enenkel, K., Quinio, V., and Swinney, P.: Holding our breath — How poor air quality blights cities, https://www.centreforcities.org/reader/cities-outlook-2020/air-quality-cities/, 2020.

EPCC: ARCHER2 User Documentation, https://docs.archer2.ac.uk, 2022.

Ferziger, J. H., Perić, M., and Street, R. L.: Computational methods for fluid dynamics, springer, 2019.

Frigo, M. and Johnson, S. G.: FFTW: An adaptive software architecture for the FFT, in: Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181), vol. 3, pp. 1381–1384, IEEE, 1998.

Greater London Authority: Interim 2021-based Population Projection Results, https://data.london.gov.uk/dataset/trend-based-population-projections, 2023.

Grylls, T. and van Reeuwijk, M.: How trees affect urban air quality: It depends on the source, Atmospheric Environment, 290, 119 275, 2022.

Grylls, T., Le Cornec, C. M., Salizzoni, P., Soulhac, L., Stettler, M. E., and van Reeuwijk, M.: Evaluation of an operational air quality model using large-eddy simulation, Atmospheric Environment: X, 3, 100 041, 2019.

Grylls, T., Suter, I., and van Reeuwijk, M.: Steady-state large-eddy simulations of convective and stable urban boundary layers, Boundary-Layer Meteorology, 175, 309–341, 2020.

Hundsdorfer, W., Koren, B., Verwer, J., et al.: A positive finite-difference advection scheme, Journal of computational physics, 117, 35–46, 1995.

Hutchins, N. and Marusic, I.: Evidence of very long meandering features in the logarithmic region of turbulent boundary layers, Journal of Fluid Mechanics, 579, 1–28, 2007.

Huttner, S.: Further development and application of the 3D microclimate simulation ENVI-met, Ph.D. thesis, Mainz, Univ., Diss., 2012, 2012.

Jarvis, P.: The interpretation of the variations in leaf water potential and stomatal conductance found in canopies in the field, Philosophical Transactions of the Royal Society of London. B, Biological Sciences, 273, 593–610, 1976.

Kubilay, A., Derome, D., and Carmeliet, J.: Coupling of physical phenomena in urban microclimate: A model integrating air flow, wind-driven rain, radiation and transport in building materials, Urban climate, 24, 398–418, 2018.

Li, N. and Laizet, S.: 2decomp & fft-a highly scalable 2d decomposition library and fft interface, in: Cray user group 2010 conference, pp. 1–13, 2010.

Lim, H., Hertwig, D., Grylls, T., Gough, H., Reeuwijk, M. v., Grimmond, S., and Vanderwel, C.: Pollutant dispersion by tall buildings: Laboratory experiments and Large-Eddy Simulation, Experiments in Fluids, 63, 92, 2022.

Louis, J.-F.: A parametric model of vertical eddy fluxes in the atmosphere, Boundary-Layer Meteorology, 17, 187–202, 1979.

Ma, Y. and Liu, H.: Large-eddy simulations of atmospheric flows over complex terrain using the immersed-boundary method in the Weather Research and Forecasting Model, Boundary-layer meteorology, 165, 421–445, 2017.

Majumdar, D., Bose, C., and Sarkar, S.: Capturing the dynamical transitions in the flow-field of a flapping foil using immersed boundary method, Journal of Fluids and Structures, 95, 102 999, 2020.

Maronga, B., Banzhaf, S., Burmeister, C., Esch, T., Forkel, R., Fröhlich, D., Fuka, V., Gehrke, K. F., Geletič, J., Giersch, S., et al.: Overview of the PALM model system 6.0, Geoscientific Model Development, 13, 1335–1372, 2020.

Masson-Delmotte, V., Zhai, P., Pörtner, H.-O., Roberts, D., Skea, J., Shukla, P. R., et al.: Global Warming of 1.5 C: IPCC special report on impacts of global warming of 1.5 C above pre-industrial levels in context of strengthening response to climate change, sustainable development, and efforts to eradicate poverty, Cambridge University Press, 2022.

Murakami, S.: Comparison of various turbulence models applied to a bluff body, in: Computational Wind Engineering 1, pp. 21–36, Elsevier, 1993.

Musy, M., Malys, L., Morille, B., and Inard, C.: The use of SOLENE-microclimat model to assess adaptation strategies at the district scale, Urban Climate, 14, 213–223, 2015.

Musy, M., Azam, M.-H., Guernouti, S., Morille, B., and Rodler, A.: The SOLENE-Microclimat model: potentiality for comfort and energy studies, Urban Microclimate Modelling for Comfort and Energy Studies, pp. 265–291, 2021.

Oke, T. R., Mills, G., Christen, A., and Voogt, J. A.: Urban climates, Cambridge University Press, 2017.

Resler, J., Krč, P., Belda, M., Juruš, P., Benešová, N., Lopata, J., Vlček, O., Damašková, D., Eben, K., Derbek, P., et al.: PALM-USM v1. 0: A new urban surface model integrated into the PALM large-eddy simulation model, Geoscientific Model Development, 10, 3635–3659, 2017.

Richards, K., Schatzmann, M., and Leitl, B.: Wind tunnel experiments modelling the thermal effects within the vicinity of a single block building with leeward wall heating, Journal of Wind Engineering and Industrial Aerodynamics, 94, 621–636, 2006.

Rolfo, S., Flageul, C., Bartholomew, P., Spiga, F., and Laizet, S.: The 2DECOMP&FFT library: an update with new CPU/GPU capabilities, Journal of Open Source Software, 8, 5813, 2023.

Schumann, U. and Sweet, R. A.: Fast Fourier transforms for direct solution of Poisson's equation with staggered boundary conditions, Journal of Computational Physics, 75, 123–137, 1988.

Skamarock, W. C., Klemp, J. B., Dudhia, J., Gill, D. O., Liu, Z., Berner, J., Wang, W., Powers, J. G., Duda, M. G., Barker, D. M., et al.: A description of the advanced research WRF version 4, NCAR tech. note ncar/tn-556+ str, 145, 2019.

Stewart, J.: Modelling surface conductance of pine forest, Agricultural and Forest meteorology, 43, 19–35, 1988.

Suter, I.: Simulating the impact of blue-green infrastructure on the microclimate of urban areas, 2018.

Suter, I., Grylls, T., Sützl, B. S., Owens, S. O., Wilson, C. E., and van Reeuwijk, M.: uDALES 1.0: a large-eddy simulation model for urban environments, Geoscientific Model Development, 15, 5309–5335, 2022.

Sützl, B. S., Rooney, G. G., Finnenkoetter, A., Bohnenstengel, S. I., Grimmond, S., and van Reeuwijk, M.: Distributed urban drag parametrization for sub-kilometre scale numerical weather prediction, Quarterly Journal of the Royal Meteorological Society, 147, 3940–3956, 2021a.

Sützl, B. S., Rooney, G. G., and van Reeuwijk, M.: Drag distribution in idealized heterogeneous urban environments, Boundary-Layer Meteorology, 178, 225–248, 2021b.

Tominaga, Y. and Blocken, B.: Wind tunnel experiments on cross-ventilation flow of a generic building with contaminant dispersion in unsheltered and sheltered conditions, Building and Environment, 92, 452–461, 2015.

United Nations Human Settlements Programme (UN-Habitat): Envisioning Future Cities: World Cities Report 2022., https://unhabitat.org/wcr/., 2022.

Uno, I., Cai, X. M., Steyn, D., and Emori, S.: A simple extension of the Louis method for rough surface layer modelling, Boundary-Layer Meteorology, 76, 395–409, 1995.

van Hooff, T., Blocken, B., and Tominaga, Y.: On the accuracy of CFD simulations of cross-ventilation flows for a generic isolated building: Comparison of RANS, LES and experiments, Building and Environment, 114, 148–165, 2017.

Verzicco, R.: Immersed boundary methods: Historical perspective and future outlook, Annual Review of Fluid Mechanics, 55, 129–155, 2023.

Vreman, A.: An eddy-viscosity subgrid-scale model for turbulent shear flow: Algebraic theory and applications, Physics of fluids, 16, 3670–3681, 2004.

Walters, D., Baran, A. J., Boutle, I., Brooks, M., Earnshaw, P., Edwards, J., Furtado, K., Hill, P., Lock, A., Manners, J., et al.: The Met Office Unified Model global atmosphere 7.0/7.1 and JULES global land 7.0 configurations, Geoscientific Model Development, 12, 1909–1963, 2019.

Watanabe, K., Kikuchi, K., Boku, T., Sato, T., and Kusaka, H.: High Resolution of City-Level Climate Simulation by GPU with Multi-physical Phenomena, in: IFIP International Conference on Network and Parallel Computing, pp. 3–15, Springer, 2021.

Wong, N. H., He, Y., Nguyen, N. S., Raghavan, S. V., Martin, M., Hii, D. J. C., Yu, Z., and Deng, J.: An integrated multiscale urban microclimate model for the urban thermal environment, Urban Climate, 35, 100 730, 2021.

Xie, Z.-T. and Castro, I. P.: Efficient generation of inflow conditions for large eddy simulation of street-scale flows, Flow, turbulence and combustion, 81, 449–470, 2008.

Xie, Z.-T., Coceal, O., and Castro, I. P.: Large-eddy simulation of flows over random urban-like obstacles, Boundary-layer meteorology, 129, 1–23, 2008.

Xu, X., Yang, Q., Yoshida, A., and Tamura, Y.: Characteristics of pedestrian-level wind around super-tall buildings with various configurations, Journal of Wind Engineering and Industrial Aerodynamics, 166, 61–73, 2017.