Seventh International Conference on Computational Fluid Dynamics (ICCFD7), Big Island, Hawaii, July 9-13, 2012

ICCFD7-2503

Efficiently Modeling Viscous Flow Effects by means of Regularization Turbulence Modeling and Local Grid Refinement

P. van der Plas[†], H.J.L. van der Heiden[†], A.E.P. Veldman[†],
R. Luppes[†], R.W.C.P Verstappen[†]

Corresponding author: P.van.der.Plas@rug.nl

[†] Institute of Mathematics and Computer Science, University of Groningen,
P.O. Box 407, 9700 AK, The Netherlands.

Abstract: In this paper, a number of methods are described that enable a more accurate and more efficient computation of the viscous flow effects with the CFD tool ComFLOW. The intended application area is the offshore environment, where flows are characterized by Reynolds numbers of at least $\mathcal{O}(10^6)$. A symmetry-preserving regularization model will be used to model turbulence. In the near-wall region, an extension of the 2-D LSSTAG immersed boundary method is described and tested for Hagen-Poiseuille flow. To account for a turbulent boundary layer near solid boundaries, the effective wall-shear stress near boundaries is adjusted according to the Werner-Wengle model. In order to compute the flows efficiently, a semi-structured local grid refinement method is implemented, which is tested for a number of flow cases.

Keywords: Local grid refinement, LES, Regularization turbulence modeling.

1 Introduction

The CFD tool ComFLOW as described in [1, 2] is used to predict free-surface flows in offshore environments. In ComFLOW, the Navier–Stokes equations can be solved for one-phase incompressible flow and for two-phase flow (for details on the latter part, see [3]). In offshore applications, extreme events of wave impact on rigid and floating structures are of high interest. There are also important applications where details of viscous flow effects become relevant, e.g. when predicting drag forces on the pillars of oil drilling rigs, or the sloshing modes in drilling holes in floating production units.

Making progress in more accurately modeling and efficiently computing the details of viscous flow effects in high Reynolds number flows requires (i) an accurate turbulence model, (ii) an accurate *discretization of diffusion near solid walls*, (iii) *modeling of the wall-shear stress* accounting for an underresolved turbulent boundary layer, and (iv) *local grid refinement* to locally achieve a high grid resolution in a computationally efficient manner.

The structure of this paper is as follows. First, the symmetry preserving discretization of the Navier–Stokes will be described. After this, the symmetry-preserving turbulence modeling approach is explained in Section 3. The discretization of the wall-shear stress model that accounts for a turbulent boundary layer is the topic of Section 4. The discretization of diffusion in cut-cells is described in Section 5. Finally, the local grid refinment approach that is pursued in ComFLOW is presented in Section 6.

2 Symmetry-preserving discretization of the Navier–Stokes equations

An excellent model for incompressible fluid flow is provided by the Navier-Stokes equations. The set of equations consists of the continuity equation

$$\mathcal{M}\mathbf{u} = 0, \tag{1}$$

where $\mathcal{M} = \nabla\cdot$ is the divergence operator, and the momentum equation

$$\frac{\partial \mathbf{u}}{\partial t} + \mathcal{C}(\mathbf{u}, \mathbf{u}) + \mathcal{G}p - \frac{1}{\mathrm{Re}}\mathcal{D}\mathbf{u} = \mathbf{f} \tag{2}$$

based on the convection operator $\mathcal{C}(\mathbf{u}, \mathbf{v}) = \mathbf{u} \cdot \nabla\mathbf{v}$, the pressure gradient operator $\mathcal{G} = \nabla$, the diffusion operator $\mathcal{D}\mathbf{u} = \nabla \cdot \nabla\mathbf{u}$ and force term $\mathbf{f}$.

The governing equations are discretized using a second-order finite-volume discretization on a staggered Cartesian grid (the Arakawa C-grid) as illustrated in fig. 1.

For time integration one can choose between a second-order Adams-Bashforth time integration scheme or a first-order Forward-Euler method [1, 2]. For simplicity, the discrete system below will be formulated in terms of the Forward-Euler time integration scheme.

The continuity equation (1) is implicitly discretized at the new time level as

$$M\mathbf{u}_h^{n+1} = -M^{\Gamma}\mathbf{u}_h^{n+1}, \tag{3}$$

where M acts on the internal of the fluid domain and M^{Γ} acts on the boundaries of the domain.

Convection and diffusion are discretized explicitly in time. As for the divergence, the pressure gradient is discretized at the new time level. Denoting the momentum control volumes by the diagonal matrix Ω gives the discretized momentum equation as

$$\Omega\frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^{n}}{\Delta t} = -C(\mathbf{u}_h^{n})\mathbf{u}_h^{n} + D\mathbf{u}_h^{n} - G\mathbf{p}_h^{n+1}. \tag{4}$$

The discrete convection operator is skew-symmetric, i.e.

$$C(\mathbf{u}_h^{n}) + C(\mathbf{u}_h^{n})^* = 0. \tag{5}$$

Let us introduce an auxiliary variable $\mathbf{u}_h^*$ as

$$\Omega\mathbf{u}_h^* \equiv \Omega\mathbf{u}_h^n - \Delta t\, C(\mathbf{u}_h^n)\mathbf{u}_h^n + \Delta t\, D\mathbf{u}_h^n, \tag{6}$$

such that we can write

$$\Omega\frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^{*}}{\Delta t} = -G\mathbf{p}_h^{n+1} \tag{7}$$

After substitution of equation (7) in the continuity equation (3) the discrete pressure Poission equation is obtained:

$$\Delta t M\Omega^{-1}G\mathbf{p}_h^{n+1} = M\mathbf{u}_h^* + M^{\Gamma}\mathbf{u}_h^{n+1}. \tag{8}$$

This equation can be regarded as a discretization of the equation $\mathcal{M}\mathcal{G}p = \mathcal{M}\mathbf{u}$. Note that the composed operator $\mathcal{M} \circ \mathcal{G}$ is not directly discretized here, but only its separate parts. Hence of sole importance for the accuracy of (8) is the discretization of the divergence and gradient operators $\mathcal{M}$ and $\mathcal{G}$, respectively. This should be kept in mind when assessing the accuracy of the method.

2.1 Divergence and pressure gradient

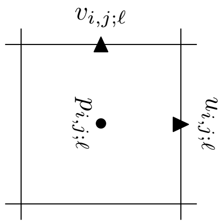


Figure 1: Staggered grid cell

The finite volume discretization of the divergence operator in the pressure cell in fig. 1 with indices i, j is given by

$$(M\mathbf{u}_h)_{i,j} = \Delta y_\ell (u_{i,j} - u_{i-1,j}) + \Delta x_\ell (v_{i,j} - v_{i,j-1}) \tag{9}$$

At the discrete level we require the divergence and gradient operators to satisfy the adjoint property, i.e.

$$G = -M^* \tag{10}$$

2.2 Note on energy conservation

Denoting the convection operator by $\mathcal{C}(\mathbf{u}, \mathbf{v}) = \mathbf{u} \cdot \nabla \mathbf{v}$, the skew-symmetry property of the operator is written as

$$(\mathcal{C}(\mathbf{u}, \mathbf{v}), \mathbf{w}) = -(\mathbf{v}, \mathcal{C}(\mathbf{u}, \mathbf{w})). \tag{11}$$

The skew-symmetry ensures the conservation of *energy* of the operator, $(\mathbf{u}, \mathcal{C}(\mathbf{u}, \mathbf{u}) = 0$, as well as the conservation of *helicity*, $(\boldsymbol{\omega}, \mathbf{u}))$, where $\boldsymbol{\omega} = \nabla \times \mathbf{u}$. In 2-D, additionally it holds that

$$(\mathcal{C}(\mathbf{u}, \mathbf{v}), \Delta \mathbf{v}) = -(\mathbf{u}, \mathcal{C}(\Delta \mathbf{v}, \mathbf{v})), \tag{12}$$

ensuring conservation of enstrophy in 2-D.

As indicated in equation (5) our discrete convection operator is also skew-symmetric, which implies that on the discrete level the same quantities are conserved by the convective term.

It should be noted that the diffusion operator is discretized such that its symmetric part is negative definite. The adjointness relation (10) and skew-symmetry of the convective term allows to write the time evolution of the kinetic energy as

$$\frac{\mathrm{d}}{\mathrm{d}t} \|\mathbf{u}_h^* \, \Omega \mathbf{u}_h\| = \mathbf{u}_h^* \left(D + D^*\right) \mathbf{u}_h \leq 0. \tag{13}$$

This shows that the time evolution of the discrete kinetic energy is only affected by the diffusion mechanism. No artificial/numerical diffusion is added to the discrete system, which guarantees that the discretization will not interfere with the subtle balance between convection and diffusion that is important for the simulation of turbulent flow.

3 Symmetry-preserving regularization modeling

In typical offshore environments, fluid flows are characterized by a high Reynolds number, $\mathcal{O}(10^6)$ or higher, and flows are highly turbulent, making it computationally impossible to resolve the full range of scales of motion. Therefore, an approximation of the solution to the original Navier–Stokes equations is sought that is dynamically less complex. Smoothing the convection term restrains the production of small scale structures, and stops the forward energy cascade that takes place at a cut-off length scale that is typically (much) larger than the smallest scale that is present in the flow: the Kolmogorov length scale.

Approaches in this direction can be traced back to the work of Leray [4], who proved that smoothing the advective velocity field in the convection term regularizes the 3-D Navier–Stokes equations. Leray's approach has inspired a number of approaches to modeling turbulent flow. A well-known example is the Navier–Stokes α-model [5]. Another class of models that has emerged in the last couple of years is the class of symmetry-preserving *regularization* turbulence models, first formulated in [6]. The importance of a symmetry-preserving discretization of the Navier–Stokes equations (the one that has been outlined above) has been assessed in [7]. In this paper, simulations of a turbulent channel flow were seen to give good results on coarse grids, without using an explicit turbulence model. The symmetry-preserving regularization approach combines these two ideas by replacing the original convection term by a skew-symmetric regularized convection term. The problem that results is well-posed from a mathematical perspective and does not interfere with the subtle balance between convection and diffusion in a turbulent flow on resolved scales.

3.1 Regularization models: analytical formulation

In order to smooth the convection operator a filter operation is introduced. The filter is a smoothing operation $\mathbf{u} \mapsto \overline{\mathbf{u}}$, and more precisely an integral kernel operation

$$\overline{\mathbf{u}}(\mathbf{x}, t) = \int_\Omega \phi_\varepsilon(\mathbf{x} - \mathbf{y})\mathbf{u}(\mathbf{y}, t)\ \mathrm{d}\Omega, \tag{14}$$

for $\mathbf{y} \in \Omega$. The time-dependency is denoted explicitly to stress that the kernel operator ϕ_ε is a purely spatial operator. The operator smooths the signal over a length scale ε. The operator ϕ_ε is self-adjoint and normalized. The residual is defined as the difference between the unfiltered and the filtered quantity: $\mathbf{u}' = \mathbf{u} - \overline{\mathbf{u}}$.

This definition of the filter allows for an expansion of the convection operator in terms of $\overline{\mathbf{u}}$ and $\mathbf{u}'$. The application of the self-adjoint filter to the convection term allows for the construction of a skew-symmetry preserving model. This family of regularization models has been proposed in [6]. The three resulting models are

$$\mathcal{C}_2(\mathbf{u}, \mathbf{v}) = \overline{\mathcal{C}(\overline{\mathbf{u}}, \overline{\mathbf{v}})}, \tag{15}$$

$$\mathcal{C}_4(\mathbf{u}, \mathbf{v}) = \mathcal{C}(\overline{\mathbf{u}}, \overline{\mathbf{v}}) + \overline{\mathcal{C}(\overline{\mathbf{u}}, \mathbf{v}')} + \overline{\mathcal{C}(\mathbf{u}', \overline{\mathbf{v}})}, \tag{16}$$

$$\mathcal{C}_6(\mathbf{u}, \mathbf{v}) = \mathcal{C}(\overline{\mathbf{u}}, \overline{\mathbf{v}}) + \mathcal{C}(\overline{\mathbf{u}}, \mathbf{v}') + \mathcal{C}(\mathbf{u}', \overline{\mathbf{v}}) + \overline{\mathcal{C}(\mathbf{u}', \mathbf{v}')}. \tag{17}$$

The subscript n indicates how accurate (namely, of order $\mathcal{O}(\varepsilon^n)$) the model approximates the original convection term.

In order for the $\mathcal{C}_n$ operator to satisfy the important skew-symmetry condition that holds for $\mathcal{C}$ in eq. (11), the filter operation should be self-adjoint. With this in mind, we turn to the discrete formulation of the regularization model.

3.2 Regularization models: numerical formulation

The discrete filter operation as it is applied to the discrete velocity field is defined as the linear operation $\overline{\mathbf{u}}_h = F\mathbf{u}_h$, and can be viewed as the discrete analogon of (14). The boundary conditions that apply to the Navier–Stokes equations, are applied to the filtered velocity $\overline{\mathbf{u}}_h$ as well. The filter F has to satisfy a number of properties:

I Symmetry: $(\Omega F)^* = \Omega F$, which guarantees the self-adjointness of the filter on the discrete level.

II Normalization: $F\mathbf{1}_h = \mathbf{1}_h$, i.e. a constant (discrete) velocity field ($\mathbf{1}_h$) should not be affected by the filter operation.

III Incompressibility: for an incompressible velocity field $\mathbf{u}_h$, the filtered velocity $\overline{\mathbf{u}}_h$ has to be divergence-free as well.

IV Low-pass filtering: only high-frequency components that occur in the discrete field should be removed by the filter.

The skew-symmetry of the discrete convection operator relies on the symmetry of the filter, but also on the velocity field being divergence-free. In a discrete setting, however, the filter cannot be guaranteed to satisfy Property III. Therefore a projection of the filtered velocity field $\overline{\mathbf{u}}$ on the space of solenoidal velocity fields is proposed. The solenoidal projection of the filtered velocity field is given by $\overline{\mathbf{u}}_h^P = \overline{\mathbf{u}}_h - M^*\mathbf{q}_h$, where the correction term $\mathbf{q}_h$ follows from the Poisson equation $MM^*\mathbf{q}_h = M\overline{\mathbf{u}}_h$. Note that solving an extra Poisson equation makes the computation roughly twice as expensive. The discrete implementation of the $\mathcal{C}_2$ model reads $\overline{C(\overline{\mathbf{u}}_h^P)\overline{\mathbf{u}}_h}$ and the discrete convection term is skew-symmetric as in the original discrete convection term in eq. (5). Similar expressions for $\mathcal{C}_4$ and $\mathcal{C}_6$ are readily derived.

Following [8], on an equidistant grid with grid spacing h, the discrete filter F is defined in polynomial form in terms of the operator $\tilde{D}_m = \Omega^{-1}\alpha_m^{1/m}h^2\, D$, where $D = M\Omega_C^{-1}M^*$ is the usual discrete diffusion operator (Ω_C denotes the diagonal matrix containing the volumes of the cell-centered variables.) The constant $\alpha_m^{1/m}$ represents the square of the ratio between the filter length ε and the local grid size h. The filter operation now reads

$$F = 1 + \sum_{m=1}^{M} \tilde{D}_m{}^m. \tag{18}$$

Note that this is the discrete analogon of an approximate deconvolution of the general class of elliptic differential filters; see [9] for an extensive review of this and other types of filters.

As will be shown later, depending on the physical properties of the flow field, the filter parameters α_m will be determined locally. Also, the filter will have to be generalized to non-uniform grids. Denote by ${A_m}^{1/m}H^2$ the diagonal matrix containing the local values of the filter length, i.e. the local values of ${\alpha_m}^{1/m}h^2$, which may now vary throughout the computational domain. The generalized form of the operator $\tilde{D}_m$ for staggered quantities reads

$$\tilde{D}_m = \Omega^{-1}M{A_m}^{1/m}H^2\Omega_C^{-1}M^*, \tag{19}$$

which resembles the form of a similar operator for cell-centered quantities that is defined in [10]. As before, M is the discrete divergence operator and $-M^*$ the discrete gradient operator.

The filter F that follows from (18) and (19) can be seen to satisfy Properties I, II, IV, all of which are properties that the filter inherits from the discrete diffusion operator.

3.3 Vortex-stretching and the dynamic determination of filter parameters

Vortex stretching is the mechanism that drives the forward energy cascade. It has been shown by Leray in [4] that a local intensification of vorticity can be prevented if the convection operator is regularized. The L_2 inner product of the vorticity vector $\boldsymbol{\omega} = \nabla \times \mathbf{u}$ with the curl of the Navier-Stokes momentum equations, gives the time evolution of the enstrophy:

$$\frac{\mathrm{d}}{\mathrm{d}t}|\boldsymbol{\omega}|^2 = (\boldsymbol{\omega}, \mathcal{C}_n(\boldsymbol{\omega}, \mathbf{u})) + \frac{1}{\mathrm{Re}}(\boldsymbol{\omega}, \Delta\boldsymbol{\omega}). \tag{20}$$

The evolution of enstrophy is seen to be determined by the vortex stretching and/or compression by the convection term and the (natural) dissipation rate due to diffusion.

3.3.1 Stopping the vortex-stretching mechanism

At the scale set by the grid, the intensification of the vorticity has to be prevented, i.e. the diffusion damping at the grid size should be enough to prevent the production of scales beyond the cut-off wave number $k_c = \pi/h$. In spectral space, this means that the following inequality (see [6]) has to be respected:

$$\frac{\boldsymbol{\omega}_{\mathbf{k}_c} \cdot \mathcal{C}_n(\boldsymbol{\omega}_{\mathbf{k}_c}, \mathbf{u})}{\omega_{\mathbf{k}_c} \cdot \boldsymbol{\omega}_{\mathbf{k}_c}} \leq \frac{1}{\mathrm{Re}}k_c^2. \tag{21}$$

If the transfer function $\widehat{G}_k$ is defined as $\widehat{\overline{\mathbf{u}}}_{\mathbf{k}} = \widehat{F\mathbf{u}}_{\mathbf{k}} = \widehat{G}_k \widehat{\mathbf{u}}_{\mathbf{k}}$, then the action of the turbulence model at cut-off wave number k_c can be written in terms of the damping function $f_n = f_n(\widehat{G}_{k_c}, \widehat{G}_p, \widehat{G}_q)$ such that

$$\frac{\boldsymbol{\omega}_{\mathbf{k}_c} \cdot \sum_{\mathbf{q}=\mathbf{p}-\mathbf{k}_c} f_n(\widehat{G}_{k_c}, \widehat{G}_p, \widehat{G}_q)\; \widehat{\mathbf{u}}_{\mathbf{p}}\, i\mathbf{q}\widehat{\mathbf{u}}_q}{\boldsymbol{\omega}_{\mathbf{k}_c} \cdot \boldsymbol{\omega}_{\mathbf{k}_c}} \leq \frac{1}{\mathrm{Re}} k_c^2. \tag{22}$$

Note that the damping function, describing the action of the regularization model in spectral space, is written as $f_n = f_n(\widehat{G}_{k_c}, \widehat{G}_p, \widehat{G}_q)$. It is clear from expression (22) that it is difficult to control the damping effect of the filter at cut-off scale k_c, as the function f_n cannot be taken out of the summation. Therefore another requirement has to be added to our list of filter properties:

V. The filter parameters should be chosen such that the damping function $f_n(\widehat{G}_{k_c}, \widehat{G}_p, \widehat{G}_q) \approx f_n(\widehat{G}_{k_c})$, i.e. the damping effect is almost independent of the interacting pair.

If the filter satisfies this property, then the damping function can be taken out of the summation, enabling the model to control the damping effect. As is shown in [11], condition (21) can now be transformed to physical space, giving

$$f_n(\widehat{G}_{k_c}) \frac{\boldsymbol{\omega} \cdot S(\mathbf{u})\, \boldsymbol{\omega}}{\boldsymbol{\omega} \cdot \boldsymbol{\omega}} \leq \frac{|\lambda_\Delta|}{\mathrm{Re}}, \tag{23}$$

where $S(\mathbf{u}) = \frac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^*\right)$ is the rate-of-strain tensor and $\lambda_\Delta < 0$ is the largest negative eigenvalue of the Laplacian operator Δ closest to zero.

As has been shown [12, 13], the left-hand side of (23) is related to invariants of the rate-of-strain tensor as $Q \equiv -\frac{1}{2}\mathrm{Tr}\ S(\mathbf{u})^2 = (\boldsymbol{\omega}, \boldsymbol{\omega})$, and $R \equiv -\det S(\mathbf{u}) = (\boldsymbol{\omega}, S(\mathbf{u})\, \boldsymbol{\omega})$. This combines to give

$$f_n(\widehat{G}_{k_c})\, \frac{R^+}{Q} \leq \frac{|\lambda_\Delta|}{\mathrm{Re}}, \tag{24}$$

As R might become negative, the quantity $R^+ = \max\{R, 0\}$ is used to determine the necessary damping.

In the discrete setting, the invariants of the discrete rate-of-strain tensor are used in criterion (24). As suggested by [13], the amount of natural damping by the diffusion operator at the cut-off scale set by the grid is the maximum amount of damping that is provided to the highest frequency that can be represented on the computational grid. Therefore, in the discrete setting λ_Δ is taken to equal the eigenvalue of the diffusion operator corresponding to the $\{+1, -1, +1, -1, \dots\}$-mode on the computational grid. This results in $-\lambda_\Delta = \frac{4}{\delta x^2} + \frac{4}{\delta y^2} + \frac{4}{\delta z^2}$.

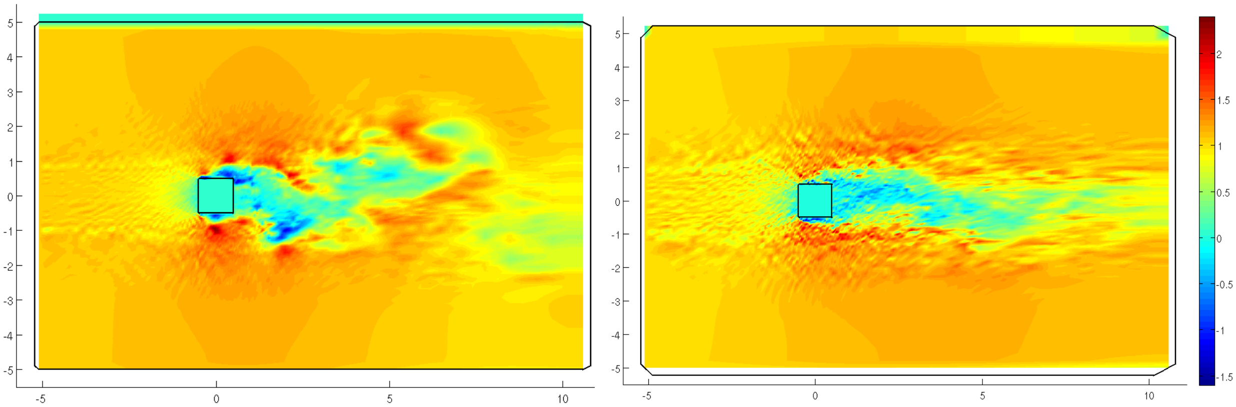
3.3.2 Determining filter parameters

The computation of the filter parameters is now straightforward. As suggested in [8], it is sufficient to choose $M = 2$ in (18) to satisfy Property V. Locally, the invariants of the rate-of-strain tensor $\{R^+, Q\}$ are computed, which determine the transfer function $\widehat{G}_{k_c}$ through inequality (24). For all the regularization models, the local parameters $\{\alpha_1, \alpha_2\}$ are expressed in terms of the transfer function $\widehat{G}_{k_c}$.

3.4 Test cases

The regularization model as described above has been tested for a number of test cases. In [6] the classical benchmark of turbulent channel flow has successfully been addressed. Furthermore, in [11] it has been shown that the regularization method is capable of predicting the turbulent flow in a differentially heated cavity accurately on a coarse computational grid. Currently, the aim is to test the model for another canonical test case: the flow around a square cylinder, at a Reynolds number of $22,000$.

To illustrate the regularization approach, the model $\mathcal{C}_2$ is considered. In fig. 2 the horizontal velocity profile at midplane height is plotted. The computational domain are as indicated in the figure, and the grid is stretched from the cylinder boundaries to the boundaries of the computational domain. The Reynolds number, based on the diameter of the cylinder is $10,000$. Although the boundary conditions that apply to the square cylinder have not been implemented in the filter operation yet, the influence of the filter in the



(a) $\mathcal{C}_2$ turbulence model

(b) no turbulence model applied

Figure 2: The instantaneous horizontal velocity profile at a midheight plane, illustrating the action of the regularization turbulence model $\mathcal{C}_2$.

interior domain is visible. The model smooths the solution for the velocity field in front of and in a band around the cylinder, and numerical wiggles in the solution are suppressed. In order to improve the solution and predict proper values for the lift and drag forces, however, the near-wall implementation of the filter should be improved.

4 The turbulent boundary layer

The turbulent flows under consideration will be characterized by a Reynolds number of $\mathcal{O}(10^6)$ or higher, making it computationally impossible to refine the grid to the level at which the boundary layer is resolved. Therefore, it is necessary to model the influence of the turbulent boundary layer. We will use the Werner-Wengle model as described in [14]. In this model, depending on the wall-tangential momentum flux, the effective wall shear stress τ_w is computed using a simple power-law approximation of the universal average velocity profile in the boundary-layer. The Werner–Wengle model reads, in terms of the dimensionless tangential velocity u^+, and the dimensionless wall-normal coordinate z^+,

$$\begin{array}{lll} u^+ = z^+ & \text{for} & z^+ \leq z_m, \\ u^+ = A\left(z^+\right)^B & \text{for} & z^+ > z_m, \end{array} \tag{25}$$

where the constants $A = 8.3$ and $B = 1/7$ result from fitting the curve to the universal velocity profile. In terms of the tangential momentum velocity component $\bar{U}$ in the first off-wall cell-face we obtain an expression for the 'effective' wall-shear stress:

$$\sqrt{\frac{\tau_w}{\rho}} = \left[\frac{1-B}{2}A^{\frac{B+1}{1-B}}\left(\frac{\nu}{\Delta z}\right)^{B+1} + \frac{B+1}{A}\left(\frac{\nu}{\Delta z}\right)^{B}\left|\bar{U}\right|\right]^{\frac{1}{B+1}}, \tag{26}$$

If the boundary layer is resolved, then the first cell spacing is well within the viscous sublayer, i.e. $z^+ < z_m$ in eq. (25), and the original discretization of the wall-shear stress is obtained.

5 Discretization of diffusion in cut-cells

ComFLOW solves the Navier–Stokes equations on a Cartesian staggered grid. In general, the boundaries are not grid-conforming and therefore it is a challenge to represent the immersed boundary and to provide proper boundary conditions on the computational grid. The discretization of the normal and shear stresses in the deformed control volumes need most attention, as currently they are approximated through a staircase representation of the geometrical object. The problem of the careful discretization of viscous stresses has been discussed in many proposals for immersed boundary methods, see [15].

For the computation of the diffusion term in the Navier–Stokes equations, the normal and shear stresses in the cut-cells are discretized in a similar fashion as the 2-D LSSTAG method that is described in [16]. The method has been extended to configurations that are aligned with one of the principal spatial directions (x, y, z) that define the Cartesian grid arrangement. An example of such an arrangement is given by fig. 3; it will be called quasi–3-D. The figure shows a geometrical shape cutting a computational cell, which is locally approximated by a plane cutting the computational cell. All the relevant geometrical information is contained in the local grid size, open cell face fractions, and fluid volume fractions.

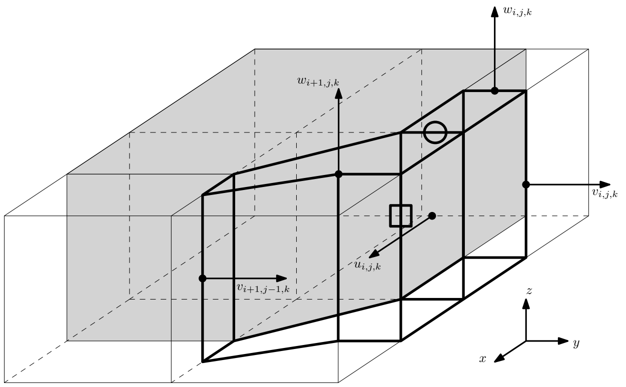


Figure 3: A possible quasi–3-D cut cell configuration.

5.1 Normal-stress discretization

Due to the staggered variable positioning that is used, the normal and shear stresses are discretized at different positions. The normal stresses $\frac{\partial u}{\partial x}, \frac{\partial v}{\partial y}, \frac{\partial w}{\partial z}$ are discretized as a cell-centered variable, like the pressure. Requiring that the Gaussian identity

$$\int_V \nabla \cdot \mathbf{u}\,\mathrm{d}V = \oint_{\partial V} \mathbf{n} \cdot \mathbf{u}\,\mathrm{d}S \tag{27}$$

holds in the discrete pressure cell, gives a natural discretization of the normal stress in terms of the geometrical information. Denoting the open cell-face fraction that is oriented normal to the u-component of the velocity by A^x, we obtain

$$\left.\frac{\partial u}{\partial x}\right|_{i,j,k} = (\Omega_C)^{-1}_{i,j,k}\left(u_{i,j,k}A^x_{i,j,k} - u_{i-1,j,k}A^x_{i-1,j,k}\right)\delta y_j \delta z_k. \tag{28}$$

5.2 Shear-stress discretization

In order to discretize the shear-stress contribution we should distinguish between different alignments of the momentum vector with the geometry. If the velocity vector is not parallel to the direction with which the geometry is aligned, e.g. u, v in fig. 3, the problem same problems occur as in 2-D. The cell-face fraction can be used straightforwardly to determine the wall shear stress at the position of the square as

$$\left.\frac{\partial u}{\partial y}\right|_{\mathrm{ib}} = \frac{u_{i,j,k} - u_{\mathrm{ib}}}{\frac{1}{2}A^x_{i,j,k}\delta y_j}.$$

Similarly, the shear stresses due to the other neighbours can be derived.

The situation is different if the discretization of the shear stresses for the velocity aligned with the principle direction of the geometry is considered. These configurations have no counterpart in 2-D. Consider the discretization of the shear stress $\frac{\partial w}{\partial x}$ on the position indicated by the square in fig. 4. If the velocities are assumed to be positioned on the barycenter of the open cell face, then the line connecting the w-velocity components is not normal to the cell face. A similar problem occurs for the discretization of the pressure gradient in a momentum control volume. The shear stress $\frac{\partial w}{\partial x}$ is therefore discretized consistent with the discretization of the pressure gradient. This gives

$$\left.\frac{\partial w}{\partial x}\right|_{i-1,j,k} = \frac{\alpha_{i-1,j,k}\left(w_{i,j,k} - w_{i-1,j,k}\right)}{\frac{1}{2}\left(A^z_{i-1,j,k}\delta x_{i-1} + A^z_{i-1,j,k}\delta x_i\right)}. \tag{29}$$

The performance of this scheme will be assessed below.

The wall-shear stress for the velocity w is obtained through computing the distance of the barycenter of the open cell-face to the wall-normal distance d_{ib}, i.e.

$$\left.\frac{\partial w}{\partial x}\right|_{\text{ib}} = \frac{w_{i,j,k} - w_{\text{ib}}}{d_{\text{ib}}}. \tag{30}$$

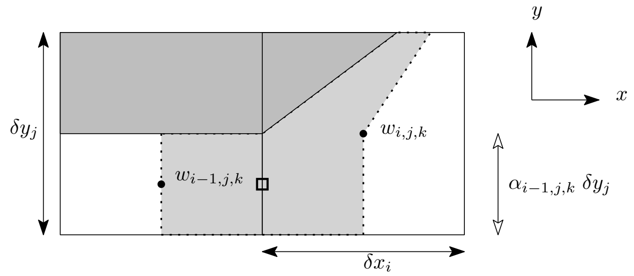
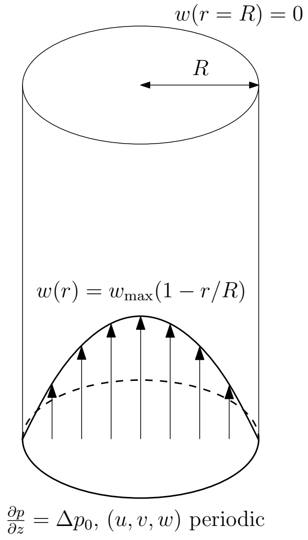


Figure 4: Two adjacent velocity vectors in cut-cell faces.

5.3 Test case: Hagen-Poiseuille flow

A suitable test case for the quasi–3-D discretization is the Hagen-Poiseuille flow, where a (constant) pressure difference drives the flow through a circular cylinder that has no-slip boundaries. The cylinder is aligned with the z-direction. In the numerical method, the circular cylinder is represented by cut cells, and the numerical solution can easily be compared with the analytical solution. The test case is set up as indicated in fig. 5. Periodic boundary conditions are applied in the z direction, in conjunction with a prescribed pressure difference between inflow and outflow. The computational domain has dimensions, $L_x \times L_y \times L_z = 1.0 \times 1.0 \times 2.0$. The diameter of the cylinder is $D = 2R = 1$, and the Reynolds number based on the cylinder diameter $\text{Re}_D = 2000$. The computational grid has a constant number of grid spacings in the z-direction and is refined in x- and y-direction. In fig. 6, the grid convergence results are presented for two cases: the original staircase discretization, and the immersed boundary method that has been outlined above. The improvement of the proposed quasi–3-D immersed boundary technique with respect to the original discretization of the diffusion in cut-cells is immediate, as the error in any norm is less. In the L_∞ norm both methods show first order behaviour. In the L_2 norm, however, the proposed discretization shows a significantly better convergence behaviour, with a convergence behaviour that is somewhere between first- and second-order, whereas the original staircase discretization shows a truly first-order convergence behaviour.



$w(r = R) = 0$
R
$w(r) = w_{\max}(1 - r/R)$
$\frac{\partial p}{\partial z} = \Delta p_0$, (u, v, w) periodic

Figure 5: The Hagen-Poiseuille problem with indicated boundary conditions

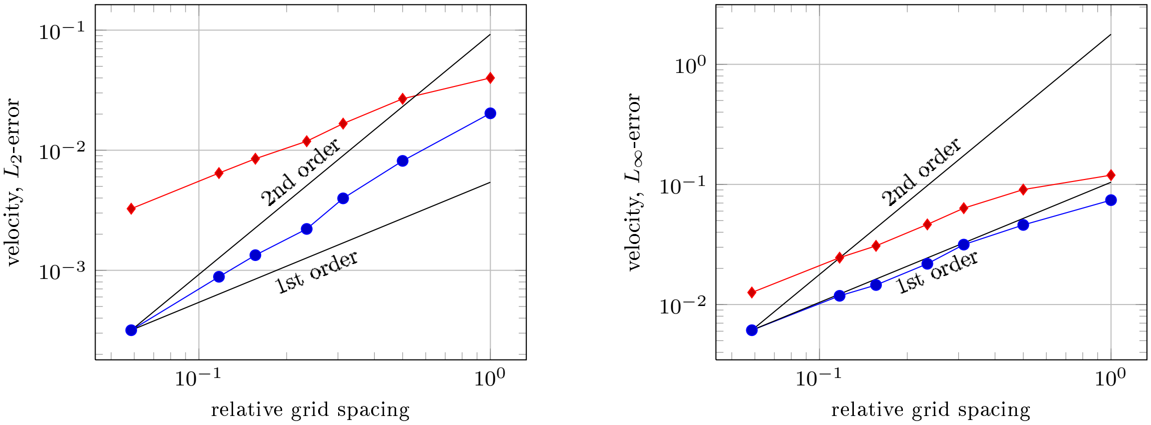


Figure 6: Convergence rate for Hagen-Poiseuille flow through a circular cylinder, *with* improved discretization of diffusion (blue dots) and *without* (red diamonds).

6 Local grid refinement

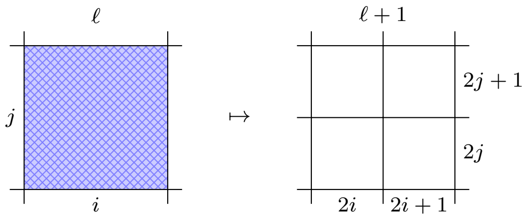


Figure 7: From coarse to fine indices

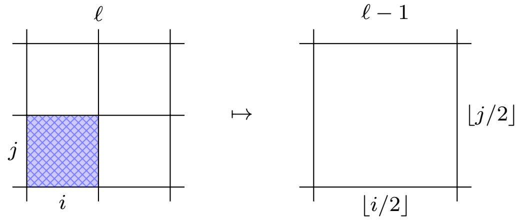


Figure 8: From fine to coarse indices

Remark: for the sake of presentation we only consider the two-dimensional case. Extension to three dimensions is straightforward.

A semi-structured approach is followed in which a cell (i, j) at refinement level ℓ is *replaced* by a set of 4 smaller cells at refinement level $\ell+1$ having indices $(2i+m, 2j+n)$ at offsets $m, n \in \{0, 1\}$. On block-shaped refinement regions the method is locally structured, hence the computational efficiency of the original array-based solution methods can be exploited as much as possible. The semi-structured indexing is illustrated in figs. 7 and 8. Only at the boundaries of the refinement regions, where the actual refinement takes place, a new treatment is required. For describing the grid layout an auxiliary array is introduced, storing only one integer for each potentially occuring cell $(i, j; \ell)$ pointing at the memory location of the subgrid in which it is contained (or null if the cell does not exist). The auxiliary array has a length of $N(1 - r_{ij})^{\ell_{\max}})/(1 - r_{ij})$, where $\ell_{\max}$ is the highest refinement level occuring on the grid, r_{ij} the product of the refinement ratios and N the number of cells on the unrefined grid. Altogether a data structure results that allows for fast and efficient look-up when compared with typical tree-based storage methods. A similar concept has been used in for example [17].

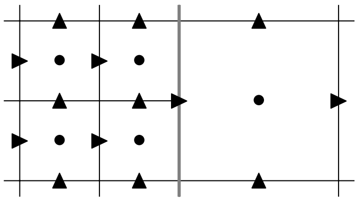


Figure 9: Arrangement of computational variables near a refinement interface (indicated with a gray line)

Typically, a large stencil is used for the approximation of missing pressure or velocity variables along the refinement interface. Interpolation of missing variables increases the number of non-zero coefficients in the pressure Poisson matrix and possibly makes the matrix non-symmetric, putting higher demands on the linear solver. Most authors use a non-overlapping and apply linear (or even higher-order) interpolation for missing variables on the other side of the interface [18]. Another approach is to apply linear interpolation inside an overlapping interface [19]. In all cases the discretization results in a non-symmetric system of equations.

In the present approach, we seek to maintain symmetry of the discretization scheme in order to exploit the use of efficient and robust symmetric iterative solvers. The idea of using a compact discretization scheme that maintains the symmetry of the discrete Poisson equation can also be found in for example [20, 21].

6.1 Divergence and pressure gradient at the interface

A compact discretization scheme is designed, which results in a small and symmetric scheme for the discrete composition $M \circ G$, making it possible to employ an efficient linear solver. Following ideas presented in [21], the pressure gradient is set constant along a refined cell face. Correspondingly, we use a uniform velocity across the entire refined cell face and only place coarse computational velocity variables at the interface (see fig. 9).

For the discretization of the divergence at the fine side of refinement interfaces, an approximation is needed for the missing fine velocities. As a first approach, missing velocities are simply approximated using constant extrapolation (see fig. 10). Assuming a uniform grid spacing for brevity, gives $\Delta x_\ell = \Delta y_\ell \equiv \Delta_\ell$, where it is remarked that the subscript ℓ is used to indicate the local refinement level. Then

$$\tilde{u}_{i,j;\ell} := u_{\lfloor i/2 \rfloor, \lfloor j/2 \rfloor; \ell-1} = u(\mathbf{x}^u_{i,j;\ell}) + O(\Delta), \tag{31a}$$

$$\tilde{u}_{i,j+1;\ell} := u_{\lfloor i/2 \rfloor, \lfloor j/2 \rfloor; \ell-1} = u(\mathbf{x}^u_{i,j+1;\ell}) + O(\Delta). \tag{31b}$$

By imposing the discrete adjoint property (10) we obtain the following approximation for missing coarse pressure variables, e.g. (see fig. 10)

$$\tilde{p}_{i-1,j,k;\ell} := \frac{p_{2i-1,2j;\ell+1} + p_{2i-1,2j+1;\ell+1}}{2} = p(\mathbf{x}^p_{i-1,j\ell}) + O(\Delta^2), \tag{32}$$

which is a simple average of the neighbouring fine pressure values. Writing out the discretization of the pressure gradient at the refinement interface we get

$$\frac{p_{i,j;\ell} - \tilde{p}_{i-1,j;\ell}}{\frac{1}{2}(\Delta x_\ell + \Delta x_{\ell+1})} = \frac{2p_{i,j;\ell} - p_{2i-1,2j;\ell+1} - p_{2i-1,2j+1;\ell+1}}{\Delta x_\ell + \Delta x_{\ell+1}} = \frac{\partial p}{\partial x}(\mathbf{x}^u_{i-1,j;\ell}) + O(\Delta). \tag{33}$$

Note that the approximation for the missing pressure variable is second-order accurate, but an order of accuracy is lost due to the loss of symmetry in the central scheme. Hence the approximation of the pressure gradient across the refinement interface is first order accurate.

Similarly, it can be seen that the scheme for the divergence operator is not consistent yet. A first order approximation of the missing velocity is used, which due to the loss of symmetry in the central difference scheme introduces a zero-th order error term,

$$\frac{\tilde{u}_{i,j;\ell} - u_{i-1,j;\ell}}{\Delta x_\ell} = \frac{u_{\lfloor i/2 \rfloor, \lfloor j/2 \rfloor; \ell-1} - u_{i-1,j;\ell}}{\Delta x_\ell} = \frac{\partial u}{\partial x}(\mathbf{x}^p_{i,j;\ell}) + O(1),$$

which will be corrected for below. More precisely, when writing out the discretization of the divergence operator it can be seen that

$$\frac{1}{\Delta x_\ell \Delta y_\ell}\left(M\mathbf{u}_h\right)_{i,j;\ell} = \frac{\partial u}{\partial x}(\mathbf{x}^p_{i,j;\ell}) + \frac{\partial v}{\partial y}(\mathbf{x}^p_{i,j;\ell}) + \frac{1}{2}\frac{\Delta y_\ell}{\Delta x_\ell}\frac{\partial u}{\partial y}(\mathbf{x}^u_{i,j;\ell}) + O(\Delta)$$

$$\frac{1}{\Delta x_\ell \Delta y_\ell}\left(M\mathbf{u}_h\right)_{i,j+1;\ell} = \frac{\partial u}{\partial x}(\mathbf{x}^p_{i,j+1;\ell}) + \frac{\partial v}{\partial y}(\mathbf{x}^p_{i,j+1;\ell}) - \frac{1}{2}\frac{\Delta y_\ell}{\Delta x_\ell}\frac{\partial u}{\partial y}(\mathbf{x}^u_{i,j+1;\ell}) + O(\Delta)$$

The inconsistency becomes particularly visible in the presence of large pressure gradients tangential to the refinement interface (e.g. hydrostatic pressure). As far as we know this issue is not addressed in the literature.

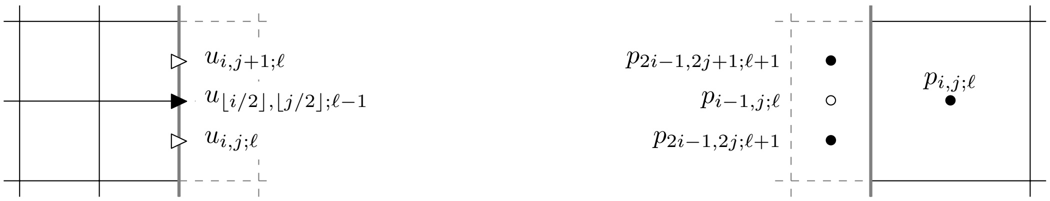


Figure 10: *Left*: missing variables (▷) for the divergence operator, together with the variable used for constant extrapolation (▶). *Right*: Missing variable (○) for the gradient operator, together with the variables used for linear interpolation (●)

In order to remove this inconsistency, a linear correction term is added to the divergence operator in cells behind refinement interfaces. For the situation depicted in fig. 10 we write

$$\left(M^{+}\mathbf{u}_h\right)_{i,j;\ell} = -\frac{1}{4}\Delta y_\ell \left(\delta_y \mathbf{u}_h\right)_{\lfloor i/2\rfloor,\lfloor j/2\rfloor;\ell-1} \tag{34}$$

$$\left(M^{+}\mathbf{u}_h\right)_{i,j+1;\ell} = \frac{1}{4}\Delta y_\ell \left(\delta_y \mathbf{u}_h\right)_{\lfloor i/2\rfloor,\lfloor j/2\rfloor;\ell-1} \tag{35}$$

where the derivative is approximated using a central differencing operator[1], which is applied along the refinement interface

$$\left(\delta_y \mathbf{u}_h\right)_{\lfloor i/2\rfloor,\lfloor j/2\rfloor;\ell-1} := \left(u_{\lfloor i/2\rfloor,\lfloor j/2\rfloor+1;\ell-1} - u_{\lfloor i/2\rfloor,\lfloor j/2\rfloor-1;\ell-1}\right)/2. \tag{36}$$

The differencing operator is similar for other interface orientations. Note that in the three-dimensional case the operator would also include a difference term in the secondary direction tangential to the interface. Through this linear correction term, the approximation of the divergence in fine cells becomes first order accurate again. In words, the amount of divergence of the auxiliary vector field $\mathbf{u}_h^*$ in fine cells behind the interface is approximated by $M + M^{+}$ instead of the original operator. Therefore, we need only to apply the correction M^{+} for the divergence operator to the right hand side of the pressure correction equation. Hence equation (8) is changed into:

$$\Delta t M \Omega^{-1} G \mathbf{p}_h^{n+1} = (M + M^{+})\mathbf{u}_h^* + M^{\Gamma}\mathbf{u}_h^{n+1}. \tag{37}$$

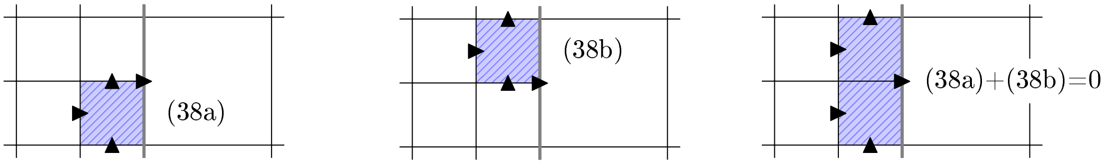
After solving the above equation and substituting $\mathbf{p}_h^{n+1}$ in equation (7), a velocity field $\mathbf{u}_h^{n+1}$ is obtained that satisfies the following equations (see figs. 11a to 11c):

$$\left(M\mathbf{u}_h^{n+1}\right)_{i,j+1;\ell} + (M^{+}\mathbf{u}_h^*)_{i,j+1;\ell} = 0 \tag{38a}$$

$$\left(M\mathbf{u}_h^{n+1}\right)_{i,j;\ell} - (M^{+}\mathbf{u}_h^*)_{i,j;\ell} = 0 \tag{38b}$$

$$\left(M\mathbf{u}_h^{n+1}\right)_{i,j+1;\ell} + \left(M\mathbf{u}_h^{n+1}\right)_{i,j;\ell} = 0 \tag{38c}$$

Note that by summing over the entire region behind the interface, the contribution to the divergence is exactly zero, as expressed in equation (38c). Hence a consistent discretization of the continuity equation is obtained for the coarse cell underlying the two fine cells.



(a) Lower cell

(b) Upper cell

(c) Total area

Figure 11: Divergence $M\mathbf{u}_h^{n+1}$ in cells along a refinement interface.

To conclude we remark that extension to the three-dimensional case is straightforward. In that case corrections have to be made in two directions tangential to the interface and the divergence sums up to zero again for each set of 4 fine cells.

6.2 Consequences for the pressure Poisson stencil

By placing coarse velocity variables $\mathbf{u}_h^{n+1}$ at the interface and by only applying linear interpolation procedures to the auxiliary variable $\mathbf{u}_h^*$ in the right hand side of the Poisson equation, we avoid the need of interpolating

[1] the interpolation is simplified by considering only the special case of uniform grids, hence $\Delta_\ell = 2\Delta_{\ell+1}$; reformulation into a linear interpolation for non-uniform grids is straightforward

$\mathbf{u}_h^{n+1}$ (and hence $\mathbf{p}_h^{n+1}$), which would have resulted in a larger pressure Poisson stencil (see e.g. [19, 18]). If we were to apply a correction M^+ to the operator on the left hand side of the pressure Poisson equation, we would have been interpolating the *velocities* $\mathbf{u}_h^{n+1}$. Now, instead, we are interpolating the auxiliary variable $\mathbf{u}_h^*$. The current approach can be interpreted as an interpolation of the "amount of necessary divergence correction". Using the operators M and G on the left hand side, as described above, leads to a compact numerical scheme for the discrete pressure Poisson scheme near refinement interfaces (as illustrated in fig. 12), avoiding the need for large interpolation stencils or overlapping grid regions. Furthermore, the number of stencil variations near corners of refinement grids or boundaries is reduced significantly.

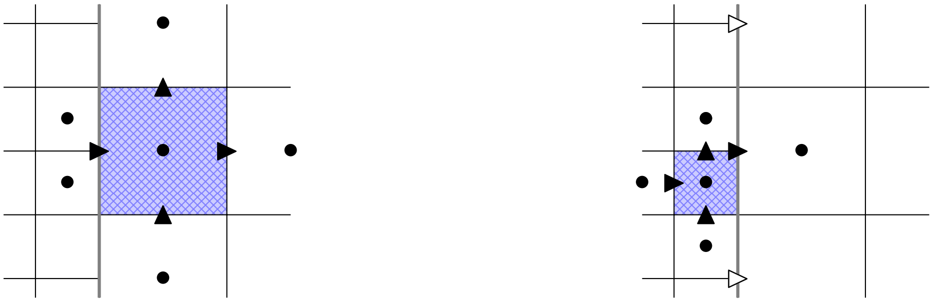


Figure 12: Visualization of the pressure Poisson stencil for the discretization given by (37). The velocity variables ▶ and ▲ are used in $M\mathbf{u}_h^{n+1}$. The actual Poisson stencil for $MG\mathbf{p}_h^{n+1}$ consists of the pressure variables ●. Note that the variables ▷ are only used on the right hand side in the term $M^+\mathbf{u}_h^*$.

To solve (37), we need the resulting Poisson matrix to be positive definitite in order to guarantee convergence of the SOR-solver as currently implemented in ComFLOW. It is not straightforward to find a theoretical proof of the (in)existence of this matrix property. A range of grid configurations of moderate size has been tested for this property and all cases resulted in positive definite matrices. The scheme is diagonally dominant in coarse interface cells and away from interfaces, but this is not the case for fine interface cells. Fortunately, in all practical test cases considered so far, this did not have a negative effect on the convergence speed.

6.3 Convection and diffusion

Missing velocities that are needed in the convection and diffusion scheme are approximated using bi-linear interpolation. Due to this interpolation, the standard second-order discretization schemes for convection and diffusion reduce to first-order accuracy at refinement interfaces. We note that in smooth regions of the solutions this does not pose a significant problem. Higher-order interpolation could be used for refinement interfaces in less smooth regions. However, the numerical error would then be dominated by the locally first-order behaviour of the divergence operator. Because it is difficult to make the divergence operator both mass conservative *and* higher-order accurate (as is also observed in [18]) this part puts limitations on the design of a higher-order scheme.

6.4 Convergence rate

To illustrate the spatial convergence behaviour of the numerical scheme, the method is applied to the following steady state vortex flow in two dimensions, at a Reynolds number of 20

$$\begin{aligned} u &= -\cos x \sin y, \\ v &= \sin x \cos y, \\ p &= \cos x + \cos y \end{aligned}$$

Appropriate forcing terms are added to the right-hand side of the momentum equations in order to make the solution exact (these are identical to the forcing terms in [19]). The solution is discretized on a periodic

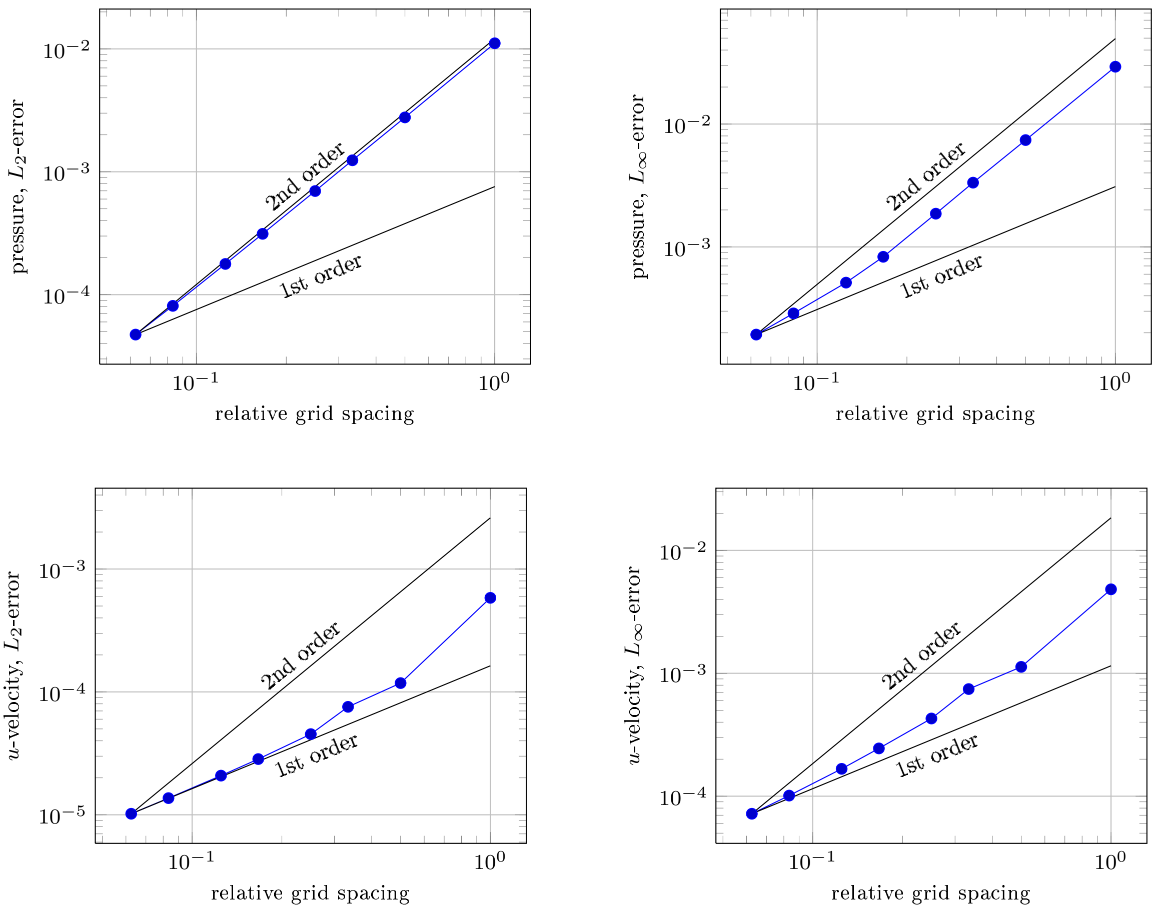


Figure 13: *Top*: convergence rate for the pressure, *bottom*: convergence rate for the x-component of the velocity. The results for the other velocity components are similar and are not presented here.

domain $0 \leq x, y < 2\pi$ for a sequence of uniform grid spacings, ranging from $\Delta = \pi/10$ to $\Delta = \pi/320$. The time step is fixed at $\Delta t = 0.0004$, ensuring a CFL number smaller than one for all considered grid configurations. The mesh interface scheme is *exact* for linear flow and the first-order error terms are proportional to the second-order derivatives of the local solution. Hence, refinement interfaces should be placed in regions of the domain where the solution has small second-order derivatives. As variations in the velocity or pressure gradient become larger, the convergence rate degrades to first order.

Since we are primarily interested in the error behaviour near the refinement interfaces, in the present test case, the refinement region is located in the region $1.35\pi \leq x, y \leq 1.65\pi$, which is a reasonably smooth part of the flow domain.

The numerical results show that convergence is initially second order in the L_∞ norm. Only upon extensive refinement the local convergence order of the pressure starts to degrade towards first order. The interface scheme for the pressure gradient introduces first-order errors that are proportional to the second derivatives of the pressure. Although these derivatives are small, they are not zero, hence the first-order error terms will eventually dominate the convergence behaviour. Note that, on the other hand, the global convergence rate of the pressure remains second order, even upon extensive refinement. This characteristic can be explained by the fact that the local error is only made at refinement interfaces, which is of lower dimension than the computational domain.

When looking at the approximation of the velocities, the convergence rate is roughly second order on coarse grids. Upon grid refinement, the local as well as the global convergence rate slowly degrade to first order, because on fine grained grids the first order error terms (which are propertional to the second

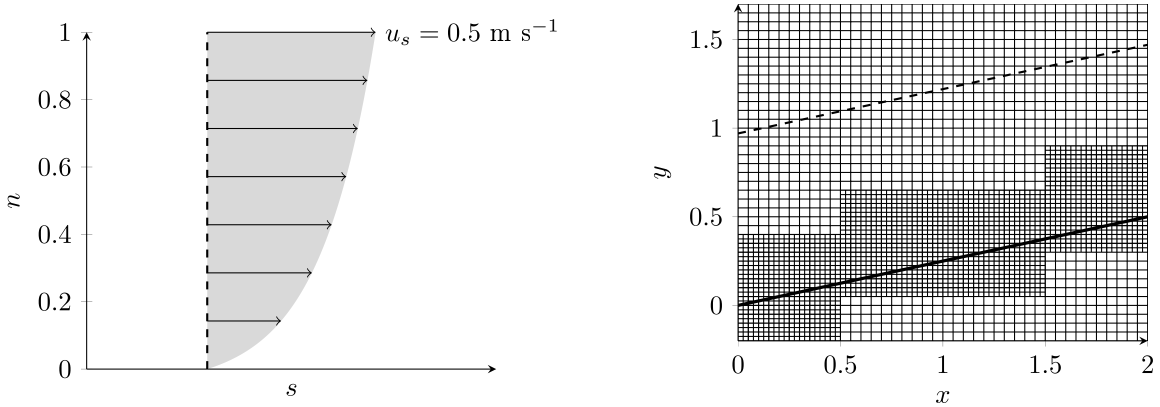


Figure 14: *Left*: Description of the manufactured solution (39) in the ns-plane. *Right*: Location of the refinement regions in the xy-plane. In the locally refined series, the grid is twice as coarse outside the refinement region whereas in the uniform series the same resolution is used outside the refinement regions.

derivatives of the solution) of the mesh interface scheme become more dominant. The results are comparable to those presented in [19], where an overlapping grid region has been used.

6.5 A more practical example

In order to test the local grid refinement approach for more practical situations, it is used to simulate a steady flow with logarithmic velocity profile. The flow is placed under an angle; we write $\mathbf{u}_h = (u_s, u_n)$, where u_s and u_n denote the streamwise and perpendicular velocity components respectively, which are given by

$$\begin{aligned} u_s &= \alpha \log(1 + \beta n) \\ u_n &= 0 \end{aligned} \quad , \quad 0 \leq n \leq 1 \tag{39}$$

The parameters α and β are chosen such that the velocity ranges from 0 at $n = 0$ (i.e. the location of the wall) to 0.5 m s^{-1} at $n = 1$. The slope of the flow is set to approximately 14 degrees. In order to reduce the effect of in- and outflow boundaries we consider a periodic domain. The flow is driven by a Dirichlet velocity boundary located at $n = 1$. At the left and right part of the domain periodic boundaries are used with an appropriate offset matching the slope of the wall.

A forcing term f_s is added to the momentum equation in streamwise direction to satisfy the Navier-Stokes equations exactly:

$$\nu \frac{\partial^2 u_s}{\partial n^2} - \frac{\partial p}{\partial s} = f_s(n)$$

Along the bottom of the channel, refinement regions are placed in order to obtain high resolution near the wall. The size and location of the refinement grids are chosen such that the interfaces are located in smooth parts of the flow (see e.g. fig. 14).

The numerical results presented in fig. 15 demonstrate the validity of the refinement method. If the refinements are correctly placed, the numerical errors introduced at the interfaces are negligible compared to the errors introduced in the rest of the domain (e.g. high gradients, irregular flow) and by the rest of the numerical scheme (e.g. boundary conditions, shear stresses, surface tension).

Remark It is important to observe that errors at refinement interfaces are not only introduced because of the modified discretization scheme. In coarse cells typically larger errors are introduced (or at least different), by which the solution can be affected. Note that also the incompressibility of the solution can play a role. Furthermore, errors can be diffused or advected with the flow. Refinement interfaces always

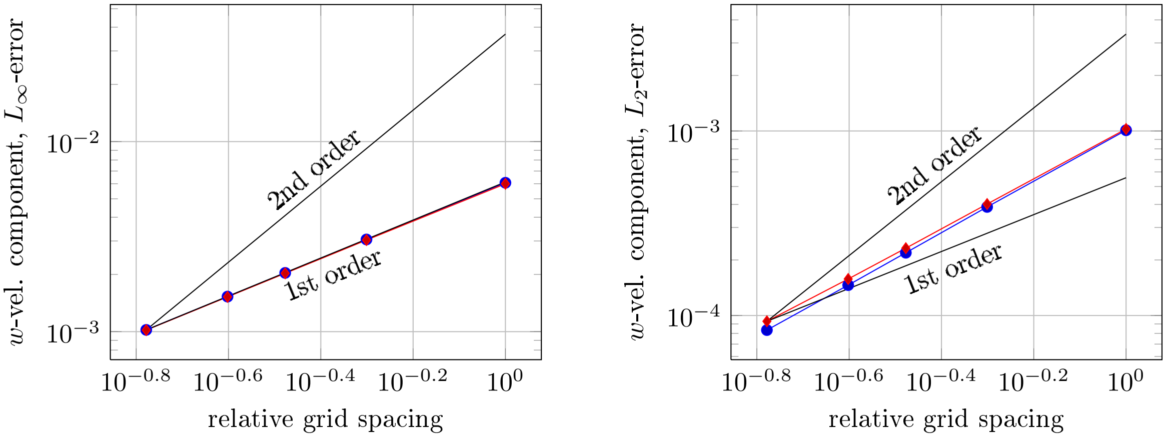


Figure 15: Numerical results for the manufactured solution (39) upon grid refinement. As starting point a uniform grid is used (red diamonds) and a locally coarsened grid (blue dots)

have one dimension less that the solution domain itself. This suggests that interface errors are best observed in the L_∞ norm.

7 Concluding remarks

In this paper, four ingredients have been presented to make efficient and accurate simulations of viscous flow effects possible. First, the regularization model as described in this paper is able to restrain the production of small-scale structures in the flow, making it possible to accurately simulate turbulent flow up to a grid resolution that can be afforded. The existing implementation has to be improved in order to enable a good prediction of wall-bounded flows. Secondly, in order to take the turbulent boundary layer into account, we use the Werner-Wengle model [14].

Thirdly, an immersed boundary technique has been proposed that extends the 2-D LSSTAG method from [16] to make quasi–3-D arrangements possible. The novel approach showed clear improvement over the existing staircase, in terms of actual accuracy as well as order of convergence.

It should be noted that these three subjects are intimately connected. An improved discretization of diffusion in cut cells gives important information on how to construct the regularization-filter operation near immersed boundaries. The immersed boundary technique might also allow us to extend the Werner-Wengle model to more general immersed boundaries than the grid-aligned boundaries for which it has originally been developed.

Finally, a local grid refinement approach for efficiently modeling viscous flow effects has been proposed, with special emphasis on the discretization across the refinement boundaries. A compact scheme is presented which reduces the overhead introduced by the refinement interfaces. Although the interface scheme is first-order accurate, the method still provides an effective approach to saving computational time while keeping comparable accuracy as in the globally refined case. By restricting the refinement interfaces to smooth areas of the flow, as is commonly done in practice, effectively the convergence rate of the original scheme is maintained. An academic test case has been presented, representing a typical practical scenario that requires local refinement that demonstrates the effectiveness of the method.

In the future these approaches will be combined to enable a complete as well as accurate description and an efficient computation of viscous flow effects.

References

[1] K.M.T. Kleefsman, G. Fekken, A.E.P. Veldman, B. Iwanowski, and B. Buchner. A volume-of-fluid based simulation method for wave impact problems. *J. Comp. Phys.*, 206(1):363–393, 2005.

[2] A.E.P. Veldman, R. Luppes, T. Bunnik, R.H.M. Huijsmans, B. Duz, B. Iwanowski, R. Wemmenhove, M.J.A. Borsboom, P.R. Wellens, H.J.L. van der Heiden, and P. van der Plas. Extreme wave impact on offshore platforms and coastal constructions. ASME, 2011.

[3] R. Wemmenhove. *Numerical simulation of two–phase flow in offshore environments.* PhD thesis, 2008.

[4] J. Leray. Sur le movement d'un liquide visqueaux emplissant l'espace. *Acta Mathematica*, 63:193–248, 1934.

[5] B.J. Geurts and D.D. Holm. Regularization modeling for large-eddy simulation. *Phys. of Fluids*, 15:L13–L16, 2003.

[6] R.W.C.P. Verstappen. On restraining the production of small scales of motion in a turbulent channel flow. *Computers & Fluids*, 37(7):887–897, 2008.

[7] R.W.C.P. Verstappen and A.E.P Veldman. Symmetry-preserving discretization of turbulent flow. *Computers & Fluids*, 187:343–368, 2003.

[8] F.X. Trias and R.W.C.P. Verstappen. On the construction of discrete filters for symmetry-preserving regularization models. *Computers & Fluids*, 40(1):139 – 148, 2011.

[9] P. Sagaut. *Large Eddy Simulation for Incompressible Flows.* Springer–Verlag, 2001.

[10] F.X. Trias, C.D. Pérez-Segarra, A. Gorobets, and A. Oliva. DNS and regularization modeling of a turbulent differentially heated cavity of aspect ratio 5. submitted to *Int. J. Heat and Mass Transfer.*

[11] F.X. Trias, R.W.C.P. Verstappen, A. Gorobets, M. Soria, and A. Oliva. Parameter-free symmetry-preserving regularization modeling of a turbulent differentially heated cavity. *Computers & Fluids*, 39(10):1815–1831, 2010.

[12] D. Chae. On the spectral dynamics of the deformation tensor and a new a priori estimates for the 3-D Euler equations. *Commun. Math. Phys.*, 263:789–801, 2005.

[13] R.W.C.P. Verstappen. When does eddy viscosity damp subfilter scales sufficiently? *J. Sci. Comp.*, 49(1):94–110, 2011.

[14] H. Werner and H. Wengle. Large-eddy simulation of turbulent flow over and around a cube in a plate channel. In *8th symposium on turbulent shear flows*, pages 155–168, 1991.

[15] R. Mittal and G. Iaccarino. Immersed boundary methods. *Ann. Rev. of Fluid Mech.*, 37(1):239–261, 2005.

[16] Y. Cheny and O. Botella. The LS-STAG method: A new immersed boundary/level-set method for the computation of incompressible viscous flows in complex moving geometries with good conservation properties. *J. Comp. Phys.*, 229:1043–1076, 2010.

[17] M.D. de Tullio, P. De Palma, G. Iaccarino, G. Pascazio, and M. Napolitano. An immersed boundary method for compressible flows using local grid refinement. *J. Comp. Phys.*, 225(2):2098–2117, 2007.

[18] M.L. Minion. A projection method for locally refined grids. *J. Comp. Phys.*, 127(1):158–178, 1996.

[19] Q. Liu. A stable and accurate projection method on a locally refined staggered mesh. *Int. J. Num. Meth. in Fluids*, 67(1):74–92, 2011.

[20] F. Losasso, F. Gibou, and R. Fedkiw. Simulating water and smoke with an octree data structure. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 457–462. ACM, 2004.

[21] E. Uzgoren, R. Singh, J. Sim, and W. Shyy. Computational modeling for multiphase flows with spacecraft application. *Progress in Aerospace Sciences*, 43(4):138–192, 2007.