

Velocity-correction schemes for the incompressible Navier–Stokes equations in general coordinate systems



D. Serson ^{a,b,*}, J.R. Meneghini ^b, S.J. Sherwin ^a

^a Department of Aeronautics, South Kensington Campus, Imperial College London, SW7 2AZ, UK

^b NDF, Escola Politécnica, Universidade de São Paulo, Av. Prof. Mello Moraes, 2231, São Paulo, 05508-030, Brazil

ARTICLE INFO

Article history:

Received 30 July 2015

Received in revised form 11 December 2015

Accepted 11 April 2016

Available online 14 April 2016

Keywords:

Incompressible flow

Velocity-correction methods

ABSTRACT

This paper presents methods of including coordinate transformations into the solution of the incompressible Navier–Stokes equations using the velocity-correction scheme, which is commonly used in the numerical solution of unsteady incompressible flows. This is important when the transformation leads to symmetries that allow the use of more efficient numerical techniques, like employing a Fourier expansion to discretize a homogeneous direction. Two different approaches are presented: in the first approach all the influence of the mapping is treated explicitly, while in the second the mapping terms related to convection are treated explicitly, with the pressure and viscous terms treated implicitly. Through numerical results, we demonstrate how these methods maintain the accuracy of the underlying high-order method, and further apply the discretisation strategy to problems where mixed Fourier-spectral/hp element discretisations can be applied, thereby extending the usefulness of this discretisation technique.

© 2016 The Authors. Published by Elsevier Inc. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The velocity-correction scheme [6,4] is a widely used time-integration method for the unsteady incompressible Navier–Stokes equations, having being applied in conjunction with finite volume [12], finite element [4], spectral-Legendre [12] and spectral/hp [6] discretisations. This method has the advantage of allowing the pressure and the velocity to be solved separately, leading to an efficient solution.

Although some of these spatial discretisations allow for simulations to be performed in arbitrary geometries, in some cases it can be useful to solve the flow in a coordinate system other than the typical Cartesian frame of reference, in order to create symmetries which might reduce the computational cost. For example, in the context of spectral/hp methods, if the coordinate transformation makes the third direction of the domain homogeneous, we can employ what is called the Fourier-spectral/hp element method proposed by Karniadakis [5]. In this approach, the homogeneous direction is discretized by a Fourier expansion, leading to a more efficient solution that can compensate for the extra computational costs of solving the Navier–Stokes in the general coordinates. This idea is illustrated in Fig. 1, which shows how we can obtain a simpler representation of a complex geometry by changing the coordinate system.

Although being able to employ the velocity-correction scheme in general coordinates would be desirable, this method has not been extended to account to general coordinate transformations. As far as the authors are aware, only specialised

* Corresponding author at: Department of Aeronautics, South Kensington Campus, Imperial College London, SW7 2AZ, UK.

E-mail addresses: d.serson14@imperial.ac.uk (D. Serson), jmeneg@usp.br (J.R. Meneghini), s.sherwin@imperial.ac.uk (S.J. Sherwin).

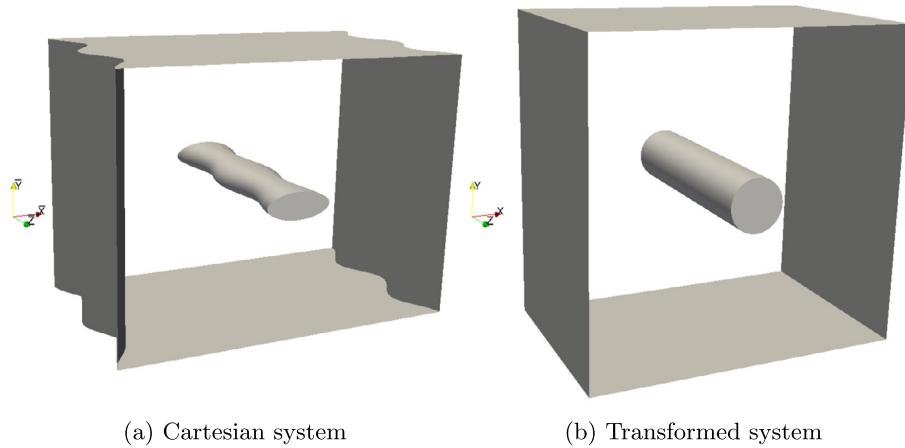


Fig. 1. Example of how a complex geometry in the physical Cartesian coordinate system can be mapped into a simpler geometry in a different coordinate system.

situations have been considered, like the constant-Jacobian time-dependent transformation of Newman and Karniadakis [11], and the constant-Jacobian time-independent mappings of Darekar and Sherwin [3]. However, no extensions have been proposed for cases where the Jacobian of the transformation is not constant.

Considering other approximations of the Navier–Stokes equations, Carlson et al. [2] proposed a method for accounting for general coordinate transformations in the context of pseudo-spectral methods, using iterative procedures to solve for the pressure and velocity fields. Although this method leads to the appropriate equations that can be used with the velocity-correction scheme, it does not provide the required high-order pressure boundary conditions that are essential to the accuracy of this time-integration scheme.

The paper proposes two methods for including coordinate transformations in the velocity-correction scheme. The first one is a generalization of the approach of Darekar and Sherwin [3] and Newman and Karniadakis [11], with the mapping being treated explicitly. On the other hand, the second method is a modified version of the iterative procedure employed by Carlson et al. [2], with the pressure and viscous terms of the mapping being treated implicitly. Neither of these methods is restricted to the constant-Jacobian case, and both of them can deal with time-dependent transformations. Therefore, it is our belief that they represent a more general and systematic approach to include coordinate transformations to the velocity-correction scheme, when compared to the works available in the literature.

The paper is organized as follows. Section 2 describes the numerical methods proposed here. Then, section 3 presents results from some simulations employing them. Finally, 4 briefly discusses some aspects of these methods and 5 presents the conclusions of the paper.

2. Numerical methods

In this section the numerical methods that allow the velocity-correction scheme to be applied to general coordinate systems are presented. First, section 2.1 briefly discusses the original velocity-correction scheme, and then section 2.2 proposes modifications to introduce coordinate transformations to this method.

2.1. The standard velocity-correction scheme

We are interested in solving the unsteady Navier–Stokes equations for an incompressible fluid, which assuming a unity density can be written as

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} &= -(\mathbf{u} \cdot \nabla) \mathbf{u} - \nabla p + \nu \nabla^2 \mathbf{u}, \\ \nabla \cdot \mathbf{u} &= 0, \end{aligned} \quad (1)$$

where $\mathbf{u}$ is the velocity, p is the pressure, and ν is the kinematic viscosity. Given a reference length L and a reference velocity U , the Reynolds number is defined as $Re = \frac{LU}{\nu}$. Equation (1) needs to be solved in the domain Ω subject to a initial condition $\mathbf{u}^0$ and to the boundary conditions

$$\begin{aligned} \mathbf{u} &= \mathbf{u}_{\mathcal{D}} & \text{on } \Gamma_{\mathcal{D}}, \\ \frac{\partial \mathbf{u}}{\partial n} &= \mathbf{u}_{\mathcal{N}} & \text{on } \Gamma_{\mathcal{N}}, \end{aligned} \quad (2)$$

where $\Gamma_{\mathcal{D}}$ is the part of the boundary where Dirichlet boundary conditions are imposed on the velocity, while in $\Gamma_{\mathcal{N}}$ Neumann boundary conditions are imposed. It is also convenient to denote the convective terms as

$$\mathbf{N}(\mathbf{u}) = -(\mathbf{u} \cdot \nabla)\mathbf{u} \quad (3)$$

and the viscous terms as

$$\nu \mathbf{L}(\mathbf{u}) = \nu \nabla^2 \mathbf{u}, \quad (4)$$

so that the momentum equation now becomes

$$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{N}(\mathbf{u}) - \nabla p + \nu \mathbf{L}(\mathbf{u}). \quad (5)$$

Following the time-integration scheme of [6], the first step to solving this problem is to discretize the momentum equation in time by approximating the time derivative by a backward differentiation formula (BDF) and by representing the convective term explicitly using a polynomial extrapolation from previous time-steps. Using these approximations, the momentum equation in time-step $n+1$ is reduced to

$$\frac{\gamma_0 \mathbf{u}^{n+1} - \sum_{q=0}^{J_i-1} \alpha_q \mathbf{u}^{n-q}}{\Delta t} = \sum_{q=0}^{J_e-1} \beta_q \mathbf{N}(\mathbf{u}^{n-q}) - \nabla p^{n+1} + \nu \mathbf{L}(\mathbf{u}^{n+1}), \quad (6)$$

where J_e and J_i are the integration orders of the explicit and implicit terms, respectively, and α , β and γ represent the coefficients of this time-integration scheme.

In order to simplify the notation of equation (6), we introduce the following definition of the summations:

$$\mathbf{u}^+ = \sum_{q=0}^{J_i-1} \alpha_q \mathbf{u}^{n-q}, \quad (7)$$

$$\mathbf{N}^* = \sum_{q=0}^{J_e-1} \beta_q \mathbf{N}(\mathbf{u}^{n-q}). \quad (8)$$

We note that the definition of the operator $*$ from equation (8) applies not only for $\mathbf{N}$, but to the extrapolation of any term. Employing these definitions, equation (6) takes the following form:

$$\frac{\gamma_0 \mathbf{u}^{n+1} - \mathbf{u}^+}{\Delta t} = \mathbf{N}^* - \nabla p^{n+1} + \nu \mathbf{L}(\mathbf{u}^{n+1}). \quad (9)$$

From this point, instead of following the formulation of Karniadakis et al. [6], we present the scheme using the framework of Guermond and Shen [4]. Although they lead to the same scheme, the later approach leads naturally to the appropriate boundary conditions, while the former requires returning to the momentum equation in order to obtain them. This distinction makes the approach of Guermond and Shen [4] easier to be generalised to include coordinate transformations, as will be considered in the next section. The idea behind this method is to first use a modified form of equation (9) with the viscous terms treated explicitly, in order to calculate the pressure and obtain an approximation of $\mathbf{u}$ satisfying the incompressibility condition. The velocity is then corrected by treating the viscous terms implicitly, hence the name velocity-correction scheme. Based on this concept, Guermond and Shen [4] show that the advance in time can be represented by the following two steps:

$$\begin{cases} \frac{\gamma_0 \bar{\mathbf{u}}^{n+1} - \mathbf{u}^+}{\Delta t} + \nabla p^{n+1} + \nu(\nabla \times \nabla \times \mathbf{u})^* - \mathbf{N}^* = 0 & \text{in } \Omega, \\ \nabla \cdot \bar{\mathbf{u}}^{n+1} = 0 & \text{in } \Omega, \\ \bar{\mathbf{u}}^{n+1} \cdot \mathbf{n} = \mathbf{u}_{\mathcal{D}} \cdot \mathbf{n} & \text{on } \Gamma_{\mathcal{D}}, \end{cases} \quad (10)$$

and

$$\begin{cases} \frac{\gamma_0 (\mathbf{u}^{n+1} - \bar{\mathbf{u}}^{n+1})}{\Delta t} - \nu \mathbf{L}(\mathbf{u}^{n+1}) - \nu(\nabla \times \nabla \times \mathbf{u})^* = 0 & \text{in } \Omega, \\ \mathbf{u}^{n+1} = \mathbf{u}_{\mathcal{D}} & \text{on } \Gamma_{\mathcal{D}}, \\ \frac{\partial \mathbf{u}^{n+1}}{\partial \mathbf{n}} = \mathbf{u}_{\mathcal{N}} & \text{on } \Gamma_{\mathcal{N}}, \end{cases} \quad (11)$$

where we have applied the $*$ operator in a analogous manner to that defined in equation (8). The term $\nabla \times \nabla \times \mathbf{u}$ comes from the decomposition $\nabla^2 \mathbf{u} = \nabla(\nabla \cdot \mathbf{u}) - \nabla \times \nabla \times \mathbf{u}$, and therefore is a valid form for the viscous terms in the continuous case, since $\nabla \cdot \mathbf{u} = 0$. The motivation to use this term instead of the Laplacian is to avoid imposing an artificial pressure boundary condition, as was noted by both Karniadakis et al. [6] and Guermond and Shen [4]. Also, $\bar{\mathbf{u}}^{n+1}$ is an intermediate velocity field satisfying the incompressibility condition, and the solution procedure does not require an explicit evaluation of this term.

In the first step (equation (10)), we want to solve for the pressure field. This can be achieved by dotting the equation with gradients of the test functions $\nabla\phi$ and integrating over the domain Ω , resulting in

$$\int_{\Omega} \nabla p^{n+1} \cdot \nabla \phi \, d\Omega = \int_{\Omega} \left[\frac{\mathbf{u}^+ - \gamma_0 \bar{\mathbf{u}}^{n+1}}{\Delta t} + \mathbf{N}^* - \nu(\nabla \times \nabla \times \mathbf{u})^* \right] \cdot \nabla \phi \, d\Omega. \quad (12)$$

Now, we can apply the divergence theorem to the right-hand-side of equation (12). Then, the equation can be simplified by imposing $\nabla \cdot \bar{\mathbf{u}}^{n+1} = 0$ and because the divergence of the curl is zero. Introducing the definition $\hat{\mathbf{u}} = \mathbf{u}^+ + \Delta t \mathbf{N}^*$, this leads to:

$$\int_{\Omega} \nabla p^{n+1} \cdot \nabla \phi \, d\Omega = \int_{\Omega} \phi \nabla \cdot \left(-\frac{\hat{\mathbf{u}}}{\Delta t} \right) d\Omega + \int_{\Gamma} \phi \left[\frac{\hat{\mathbf{u}} - \gamma_0 \bar{\mathbf{u}}^{n+1}}{\Delta t} - \nu(\nabla \times \nabla \times \mathbf{u})^* \right] \cdot \mathbf{n} \, dS, \quad (13)$$

which is the weak form of a Poisson equation, with the first integral on the right-hand-side representing a forcing term and the second integral a high-order Neumann pressure boundary condition.

The velocity-correction in the second step (equation (11)) can be performed by substituting $\bar{\mathbf{u}}^{n+1}$ from equation (10) and defining $\hat{\mathbf{u}} = \hat{\mathbf{u}} - \nabla p^{n+1} \Delta t$, resulting in

$$\frac{\gamma_0 \mathbf{u}^{n+1} - \hat{\mathbf{u}}}{\Delta t} = \nu \mathbf{L}(\mathbf{u}^{n+1}), \quad (14)$$

with the term $\nu(\nabla \times \nabla \times \mathbf{u})^*$ cancelled out in the substitution. Solving this Helmholtz equation we obtain the velocity field, completing the solution for time-step $n + 1$.

2.2. The velocity-correction scheme in general coordinates

We now move to the problem of solving the Navier–Stokes equations in a general coordinate system. We will denote the usual Cartesian system by $(\bar{x}, \bar{y}, \bar{z})$ and the transformed system by (x, y, z) , so that a general time-dependent transformation can be represented by:

$$\begin{aligned} \bar{x} &= \bar{x}(x, y, z, t), \\ \bar{y} &= \bar{y}(x, y, z, t), \\ \bar{z} &= \bar{z}(x, y, z, t). \end{aligned} \quad (15)$$

In order to study the flow using the transformed domain represented by the (x, y, z) coordinate system, it is necessary to obtain the appropriate form of the transformed Navier–Stokes equations. This can be done using tensor calculus, with the result for a general time-dependent transformation being [10]:

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} &= \bar{\mathbf{N}}(\mathbf{u}) - \bar{\mathbf{G}}(p) + \nu \bar{\mathbf{L}}(\mathbf{u}), \\ D(\mathbf{u}) &= 0, \end{aligned} \quad (16)$$

where:

$$\begin{aligned} \bar{\mathbf{N}}(\mathbf{u}) &= -u^j u_{,j}^i + V^j u_{,j}^i - u^j V_{,j}^i, \\ \bar{\mathbf{G}}(p) &= g^{ij} p_{,j}, \\ \nu \bar{\mathbf{L}}(\mathbf{u}) &= \nu g^{jk} u_{,jk}^i, \\ D(\mathbf{u}) &= \frac{1}{J} \nabla \cdot (J u^i), \end{aligned} \quad (17)$$

with g^{ij} representing the inverse of the metric tensor, u^i the components of the vector $\mathbf{u}$, J the Jacobian of the transformation to the Cartesian system, and a subscript after a comma denotes the covariant derivative. The term $V^j = -\frac{\partial x^j}{\partial t}$ represents the velocity of the coordinate system, and therefore is only relevant for time-dependent transformations. For the fundamentals of tensor calculus leading to the derivation of equation (16), the reader is referred to Aris [1]. To simplify the notation, the ∇ operator will be assumed to correspond to the usual Cartesian operation representing the partial derivatives.

After obtaining the appropriate form of the transformed Navier–Stokes equations, we can consider the domain represented by the (x, y, z) coordinates and solve equation (16). Ignoring any numerical approximation errors at this point, the result of the transformed equation should be equivalent to solving the Navier–Stokes equations in the original system. However, in order to solve equation (16), the time-integration scheme of section 2.1 needs to be properly modified to this equation. Sections 2.2.1 and 2.2.2 propose two different approaches to achieve this.

2.2.1. Explicit treatment of the mapping

In this section, a method to solve equation (16) treating all the mapping terms explicitly is proposed. First, equation (16) is restated as

$$\begin{aligned}\frac{\partial \mathbf{u}}{\partial t} &= \mathbf{N}(\mathbf{u}) - \frac{\nabla p}{J} + \nu \mathbf{L}(\mathbf{u}) + \mathbf{A}(\mathbf{u}, p), \\ D(\mathbf{u}) &= 0,\end{aligned}\quad (18)$$

where $\mathbf{N}(\mathbf{u})$ and $\mathbf{L}(\mathbf{u})$ are the usual convective and viscous operators from equations (3) and (4), and

$$\mathbf{A}(\mathbf{u}, p) = [\tilde{\mathbf{N}}(\mathbf{u}) - \mathbf{N}(\mathbf{u})] + \left[-\tilde{\mathbf{G}}(p) + \frac{\nabla p}{J} \right] + \nu [\tilde{\mathbf{L}}(\mathbf{u}) - \mathbf{L}(\mathbf{u})] \quad (19)$$

is a forcing term that imposes the coordinate transformation and can clearly be interpreted as the difference between the Cartesian and transformed expressions. Although it is not explicit in the equations, $\mathbf{A}$ also depends on the particular form of the mapping employed. We note that at this step, the momentum equation has been restated in a form that is almost the same as the original Navier–Stokes equations with a forcing term, with the difference being the Jacobian dividing the pressure gradient. The reason for including the Jacobian here will become evident later, when it will allow the equations to be simplified using the incompressibility condition $\nabla \cdot (J\mathbf{u}) = 0$.

The next step is to modify equation (9) accordingly, with the forcing term being treated explicitly using an extrapolation from previous time-steps, in the same manner as the convective term, leading to:

$$\frac{\gamma_0 \mathbf{u}^{n+1} - \mathbf{u}^+}{\Delta t} = \mathbf{N}^* + \mathbf{A}^* - \frac{\nabla p^{n+1}}{J} + \nu \mathbf{L}(\mathbf{u}^{n+1}), \quad (20)$$

where the meaning of $\mathbf{u}^+$, $\mathbf{N}^*$ and $\mathbf{A}^*$ follows from the definitions of equations (7) and (8).

Another aspect that must be modified is the decomposition of the viscous term employed in the velocity-correction scheme. The relation $\nabla^2 \mathbf{u} = \nabla(\nabla \cdot \mathbf{u}) - \nabla \times \nabla \times \mathbf{u}$ that was used in the Cartesian case is not helpful in the transformed system, since the incompressibility condition is no longer contained in the condition $\nabla \cdot \mathbf{u} = 0$. Therefore, it is more convenient to use the identity $\nabla^2 \mathbf{u} = \nabla(D(\mathbf{u})) - \nabla\left(\frac{\mathbf{u}}{J} \cdot \nabla J\right) - \nabla \times \nabla \times \mathbf{u}$, since the first term on the right-hand-side vanishes, leading to a behaviour similar to the use of the divergence condition in the original Cartesian problem. Therefore, we propose the time-integration scheme of equations (10) and (11) can be replaced by:

$$\begin{cases} \frac{\gamma_0 \tilde{\mathbf{u}}^{n+1} - \mathbf{u}^+}{\Delta t} + \frac{\nabla p^{n+1}}{J} + \nu(\nabla\left(\frac{\mathbf{u}}{J} \cdot \nabla J\right))^* + \nu(\nabla \times \nabla \times \mathbf{u})^* - \mathbf{N}^* - \mathbf{A}^* = 0 & \text{in } \Omega, \\ D(\tilde{\mathbf{u}}^{n+1}) = 0 & \text{in } \Omega, \\ \tilde{\mathbf{u}}^{n+1} \cdot \mathbf{n} = \mathbf{u}_{\mathcal{D}} \cdot \mathbf{n} & \text{on } \Gamma_{\mathcal{D}}, \end{cases} \quad (21)$$

and

$$\begin{cases} \frac{\gamma_0(\mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1})}{\Delta t} - \nu \mathbf{L}(\mathbf{u}^{n+1}) - \nu(\nabla\left(\frac{\mathbf{u}}{J} \cdot \nabla J\right))^* - \nu(\nabla \times \nabla \times \mathbf{u})^* = 0 & \text{in } \Omega, \\ \mathbf{u}^{n+1} = \mathbf{u}_{\mathcal{D}} & \text{on } \Gamma_{\mathcal{D}}, \\ \frac{\partial \mathbf{u}^{n+1}}{\partial \mathbf{n}} = \mathbf{u}_{\mathcal{N}} & \text{on } \Gamma_{\mathcal{N}}. \end{cases} \quad (22)$$

To solve the first step, we multiply the equation by J , dot with the gradient of the test functions $\nabla \phi$ and integrate in the transformed domain. Defining $\hat{\mathbf{u}} = \mathbf{u}^+ + \Delta t(\mathbf{N}^* + \mathbf{A}^*)$, results in

$$\int_{\Omega} \nabla p^{n+1} \cdot \nabla \phi \, d\Omega = \int_{\Omega} J \left[\frac{\hat{\mathbf{u}} - \gamma_0 \tilde{\mathbf{u}}^{n+1}}{\Delta t} - \nu \left(\nabla \left(\frac{\mathbf{u}}{J} \cdot \nabla J \right) \right)^* - \nu(\nabla \times \nabla \times \mathbf{u})^* \right] \cdot \nabla \phi \, d\Omega. \quad (23)$$

Applying the divergence theorem to the right-hand-side of the equation, and noting that the divergence of the curl is zero and that from the incompressibility condition $\nabla \cdot (J\tilde{\mathbf{u}}^{n+1}) = 0$, we obtain:

$$\begin{aligned} \int_{\Omega} \nabla p^{n+1} \cdot \nabla \phi \, d\Omega &= \int_{\Omega} \phi \nabla \cdot \left[-\frac{J\hat{\mathbf{u}}}{\Delta t} + \nu \left(\nabla \left(\frac{\mathbf{u}}{J} \cdot \nabla J \right) \right)^* \right] + \nu \nabla J \cdot (\nabla \times \nabla \times \mathbf{u})^* \, d\Omega \\ &\quad + \int_{\Gamma} \phi J \left[\frac{\hat{\mathbf{u}} - \gamma_0 \tilde{\mathbf{u}}^{n+1}}{\Delta t} - \nu \left(\nabla \left(\frac{\mathbf{u}}{J} \cdot \nabla J \right) \right)^* - \nu(\nabla \times \nabla \times \mathbf{u})^* \right] \cdot \mathbf{n} \, dS. \end{aligned} \quad (24)$$

This is still a Poisson equation in the weak form for the pressure, and can be solved using the same method as in the original velocity-correction scheme, with the only changes appearing in the forcing term and in the high-order boundary condition term in the right-hand-side of the equation.

In the second step of the solution, both $\nu(\nabla(\frac{\mathbf{u}}{J} \cdot \nabla J))^*$ and $\nu(\nabla \times \nabla \times \mathbf{u})^*$ cancel out upon substituting $\bar{\mathbf{u}}^{n+1}$ from equation (21) in equation (22), and by changing the definition of $\hat{\mathbf{u}}$ to $\hat{\mathbf{u}} = \hat{\mathbf{u}} - \frac{\nabla p^{n+1}}{J} \Delta t$ we once again obtain equation (14). Therefore the definition of $\hat{\mathbf{u}}$ is the only change that needs to be performed in this step when comparing it to the original velocity-correction scheme. It is clear that this formulation requires only small modifications to the original velocity-correction scheme procedure:

- a forcing term needs to be added to the convective term,
- the forcing terms for the pressure and viscous systems need to be modified,
- the high-order pressure boundary conditions are slightly different.

This characteristic is one of the advantages of the method, since an existing solver can be easily adapted to include the coordinate transformation.

An interesting particular case of this method occurs when the Jacobian of the transformation is constant, usually $J = 1$. In this case, we have a divergence-free mapping, and the only modification that is required is adding the forcing term $\mathbf{A}(\mathbf{u}, p)$ to the convective term. The resulting method is exactly the same as the one used by Newman and Karniadakis [11] and Darekar and Sherwin [3], and therefore we can consider their approach a particular case of the more general procedure presented here.

2.2.2. Semi-implicit treatment of the mapping

This section describes an approach to solve the equations where the mapping terms arising from the convective part of the equation are treated explicitly, while the pressure and viscous terms are treated implicitly, maintaining the characteristics of the original splitting scheme. This is a modified version of the method used by Carlson et al. [2] and later restated by Koberg [8] with a notation similar to that original presented by Karniadakis et al. [6]. The main difference from their approach is that by following the framework from section 2.1, we can obtain the appropriate pressure boundary conditions directly, while in their method it is not clear what these conditions should be for this semi-implicit formulation.

The first step is to note that the operator $\bar{\mathbf{L}}$ can be decomposed as:

$$\bar{\mathbf{L}}(\mathbf{u}) = g^{jm} u^i_{,mj} = g^{ji} (u^m_{,m})_{,j} - \varepsilon^{imn} \varepsilon^{ljk} g_{nl} g_{kp} u^p_{,jm}, \quad (25)$$

where g_{ij} is the metric tensor and $\varepsilon^{ijk} = g^{-1/2} \epsilon^{ijk}$, with ϵ^{ijk} being the permutation symbol, is a generalization of the permutation symbol. Noting that $u^m_{,m} = D(\mathbf{u}) = 0$, we conclude that this is a generalised form of the decomposition employed in the Cartesian case. Defining $\mathbf{Q} = \varepsilon^{imn} \varepsilon^{ljk} g_{nl} g_{kp} u^p_{,jm}$, we propose to restate the velocity-correction scheme of equations (10) and (11) as

$$\begin{cases} \frac{\gamma_0 \bar{\mathbf{u}}^{n+1} - \mathbf{u}^+}{\Delta t} + \bar{\mathbf{G}}(p^{n+1}) + \nu \mathbf{Q}^* - \bar{\mathbf{N}}^* = 0 & \text{in } \Omega, \\ D(\bar{\mathbf{u}}^{n+1}) = 0 & \text{in } \Omega, \\ \bar{\mathbf{u}}^{n+1} \cdot \mathbf{n} = \mathbf{u}_{\mathcal{D}} \cdot \mathbf{n} & \text{on } \Gamma_{\mathcal{D}}, \end{cases} \quad (26)$$

and

$$\begin{cases} \frac{\gamma_0 (\mathbf{u}^{n+1} - \bar{\mathbf{u}}^{n+1})}{\Delta t} - \nu \bar{\mathbf{L}}(\mathbf{u}^{n+1}) - \nu \mathbf{Q}^* = 0 & \text{in } \Omega, \\ \mathbf{u}^{n+1} = \mathbf{u}_{\mathcal{D}} & \text{on } \Gamma_{\mathcal{D}}, \\ \frac{\partial \mathbf{u}^{n+1}}{\partial \mathbf{n}} = \mathbf{u}_{\mathcal{N}} & \text{on } \Gamma_{\mathcal{N}}. \end{cases} \quad (27)$$

The first step is solved using the following iteration:

$$\nabla p_{s+1}^{n+1} = \nabla p_s^{n+1} + J \left[\frac{\mathbf{u}^+ - \gamma_0 \bar{\mathbf{u}}^{n+1}}{\Delta t} - \nu \mathbf{Q}^* + \bar{\mathbf{N}}^* - \bar{\mathbf{G}}(p_s^{n+1}) \right], \quad (28)$$

where s is the iteration counter. Instead of using this iterative procedure, it would be possible to perform this step directly, by following a procedure similar to the one in the original velocity-correction scheme. However, this would involve assembling the matrix representing the operator $D(\bar{\mathbf{G}}(p))$, which in a 2.5D problem would couple the different Fourier modes. Therefore, performing this step directly would eliminate any performance gains we might obtain by using the coordinate transformation. Also, the direct approach would not be practical for time-dependent transformations, since the system would have to be updated at every time-step.

Dotting equation (28) with $\nabla \phi$ and integrating to obtain the weak form, and after using the identities $\nabla \cdot (J \bar{\mathbf{u}}^{n+1}) = 0$ and $D(\mathbf{Q}) = 0$, the equation becomes:

$$\begin{aligned} \int_{\Omega} \nabla p_{s+1}^{n+1} \cdot \nabla \phi \, d\Omega &= \int_{\Omega} \phi \left[JD \left(\frac{-\hat{\mathbf{u}}}{\Delta t} \right) + JD(\bar{\mathbf{G}}(p_s^{n+1})) - \nabla^2 p_s^{n+1} \right] d\Omega \\ &+ \int_{\Gamma} \phi \left[J \left(\frac{\hat{\mathbf{u}} - \gamma_0 \bar{\mathbf{u}}^{n+1}}{\Delta t} \right) - \nu J \mathbf{Q}^* - J \bar{\mathbf{G}}(p_s^{n+1}) + \nabla p_s^{n+1} \right] \cdot \mathbf{n} \, dS, \end{aligned} \quad (29)$$

where $\hat{\mathbf{u}} = \mathbf{u}^+ + \Delta t \bar{\mathbf{N}}^*$. This Poisson equation needs to be solved at each iteration; however, most of the terms in the right-hand-side are not modified during the iterative procedure, and therefore only need to be computed once per time-step.

Similarly, the velocity system of the second step can be solved using the following iterative procedure:

$$\frac{\gamma_0 \mathbf{u}_{s+1}^{n+1}}{\Delta t} - \nu \mathbf{L}(\mathbf{u}_{s+1}^{n+1}) = \frac{\hat{\mathbf{u}}}{\Delta t} - \bar{\mathbf{G}}(p^{n+1}) + \nu \bar{\mathbf{L}}(\mathbf{u}_s^{n+1}) - \nu \mathbf{L}(\mathbf{u}_s^{n+1}), \quad (30)$$

where each iteration consists of solving a Helmholtz equation for each velocity component. Also, we note that the iterative procedures of equations (28) and (30) can be modified to include a relaxation parameter, making them more robust.

3. Test cases

This section presents results of tests employing the previous methods in three different types of flows, the first two validating the formulation, with the third demonstrating how this approach is able to deal with very general problems with time-dependent mappings. The first case considered is the Kovasznay flow, for which there is an exact analytical solution, allowing for the accuracy of the methods to be evaluated. The second case is the flow around an infinite square cylinder deformed along the span direction. This is a more complex flow with results at low Reynold numbers available in the literature [3], allowing to assess how the method performs in a more practical scenario. The last simulation presented is the two-dimensional flow around two circular cylinders in tandem, with the upstream cylinder subject to forced oscillations, while the downstream cylinder is held fixed.

All simulations employ the spatial discretisation of the spectral/hp method discussed in Karniadakis and Sherwin [7], with the three-dimensional cases using a Fourier expansion in the third direction, as presented by Karniadakis [5]. However, we note that the methods proposed here do not depend on this particular choice of discretisation.

3.1. Kovasznay flow

The Kovasznay flow, presented by Kovasznay [9], consists in a steady analytical solution for the two-dimensional Navier–Stokes equations with a periodic direction, which can be viewed as a representation of the flow behind a two-dimensional grid. Considering a three-dimensional domain with the flow in the xz plane, with z being the periodic direction, the exact solution is

$$\begin{aligned} u &= 1 - e^{kx} \cos(2\pi z), \\ v &= 0, \\ w &= \frac{k}{2\pi} e^{kx} \sin(2\pi z), \\ p &= \frac{1}{2} \left(1 - e^{2kx} \right), \end{aligned} \quad (31)$$

where the constant k is defined as

$$k = \frac{Re}{2} - \sqrt{\frac{Re^2}{4} + 4\pi^2}, \quad (32)$$

with Re being the Reynolds number.

Simulations for this flow were performed in four different situations. The first did not employ any mapping, and therefore serves as a reference to compare the methods proposed here. The second case considered a divergence-free mapping, using the transformation

$$\bar{x} = x + \xi(z) = x - 0.05 \cos(2\pi z). \quad (33)$$

The third scenario considered an explicit treatment of the mapping, while the fourth dealt with the semi-implicit case, both using the mapping

$$\bar{x} = f(x, z) = x - 0.05 \frac{\tanh(1-x)}{\tanh(1)} \cos(2\pi z). \quad (34)$$

In all cases the y and z coordinates were not modified.

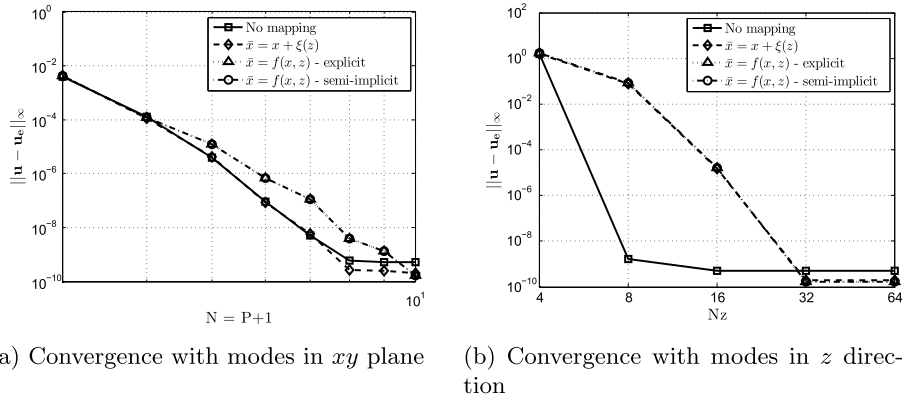


Fig. 2. Convergence on the infinity norm of the u velocity error for the Kovaszny flow with different mappings.

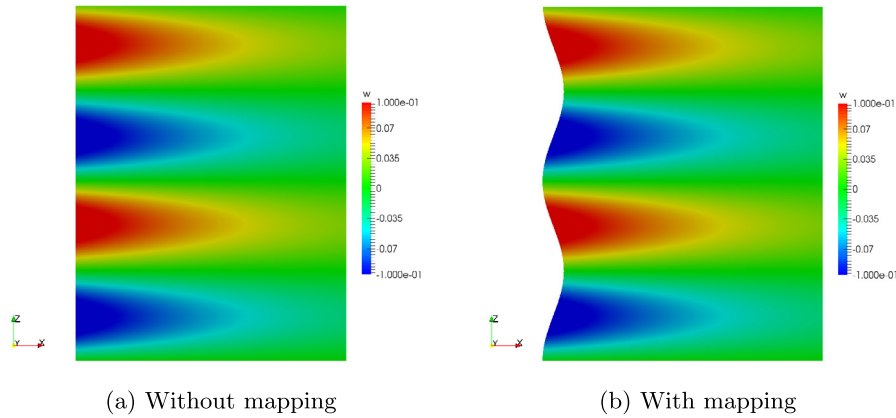


Fig. 3. Contours of the w velocity component of the Kovaszny flow in the xz plane for the simulations with no mapping and with the mapping of equation (34).

It is clear that the study of this simple flow does not benefit from the use of these transformations, and therefore the only motivation for using them here is for testing the accuracy of the methods.

The simulations consisted of advancing the equations in time for 15 time units using a time-step of 0.001 with a first-order time integration, with $Re = 40$. The computational domain extends from -0.5 to 1.0 in x , from -0.5 to 1.5 in y and from 0 to 2 in z , with the xy plane represented by 12 quadrilateral elements and the z direction discretized using a Fourier expansion. The reference resolution considered 10 modes per direction in each element of the xy plane (9^{th} order polynomials) with 64 degrees of freedom for the Fourier expansion in the z direction. From this reference, either the polynomial order or the number of Fourier modes were reduced, in order to determine how these parameters affect the error.

Fig. 2 presents the results in term of the infinity norm (maximum value in the quadrature points) of the error for the u velocity. We note that for the polynomial order, the mapping $\bar{x} = f(x, z)$ has a slight loss of accuracy, but all cases still show a good convergence. In the case of the z direction resolution, the use of the mappings results in higher errors when the number of modes is small, although this is likely due to the fact that in the absence of any mapping the solution can be exactly represented by the Fourier expansion with just the mean mode and the second harmonic, corresponding to $N_z = 6$. As the number of modes increase, once again the errors are comparable in all cases. It is also clear that the choice between the explicit or semi-implicit treatment of the mappings has little effect on the accuracy of these results. In fact, the difference between the solution from the two methods is negligible, with the infinity norm of the difference in the u velocity usually being at least two orders of magnitude lower than the discretisation error from Fig. 2.

Fig. 3 presents contours of the w velocity for the first case (no mapping) and the fourth case (mapping of equation (34)), both for the highest spatial resolution considered. We note that although the physical domain is different (since the computational domain is the same), the results are equivalent, showing how the exact solution is obtained even in the deformed geometry.

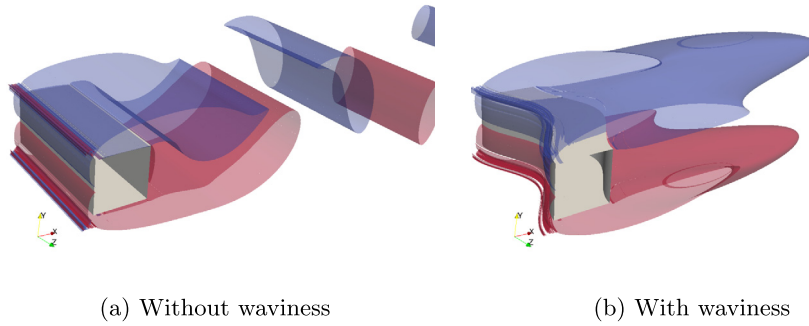


Fig. 4. Contours of spanwise vorticity for square cylinder with and without spanwise waviness with $Re = 100$. The waviness has a wavelength $\frac{\lambda}{h} = 2.8$ and amplitude $\frac{W}{\lambda} = 0.2$.

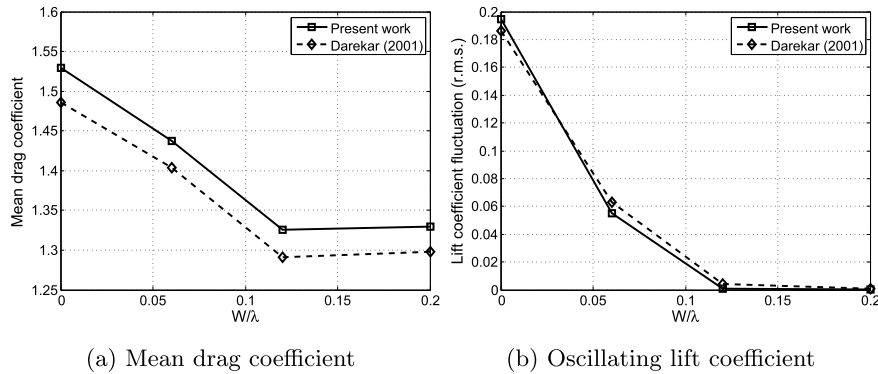


Fig. 5. Comparison of results for a square cylinder with waviness of wavelength $\frac{\lambda}{h} = 2.8$ at $Re = 100$, with the results from [3]. Each case consists of a different amplitude W .

3.2. Square cylinder with waviness

The flow around an infinite square cylinder with spanwise sinusoidal waviness was studied numerically by Darekar and Sherwin [3], for a Reynolds number $Re = 100$. As noted previously, the method they employed is a particular case of the explicit method of section 2.2.1, and therefore this is a good test to check if the current implementation is consistent with theirs. A few cases considered by them were studied here, all of them with a waviness of wavelength $\frac{\lambda}{h} = 2.8$, where h is the height of the cylinder. For each case, a coordinate transformation is employed to remove the waviness from the cylinder, allowing for the use of a Fourier discretisation in the span direction. Fig. 4 shows contours of spanwise vorticity for the cylinder without any waviness and for a waviness with amplitude $\frac{W}{\lambda} = 0.2$, illustrating the geometry being considered, and how vortex shedding is suppressed in the modified geometry. The results of mean drag and oscillating lift coefficients for different waviness amplitudes are presented in Fig. 5, showing a good agreement with the results from Darekar and Sherwin [3]. Although there is a small discrepancy (approximately 3%) in the drag coefficient, this is likely due to different resolutions and domain sizes in both sets of simulations, since their convergence studies indicate errors around 2%, while the simulations presented here were not preceded by any rigorous convergence studies. The fact that this systematic error persists even for a zero amplitude, where no mapping is employed, further supports the claim that the discrepancies are not caused by the treatment of the waviness.

3.3. Flow around moving cylinders

To demonstrate the possibility of using the techniques presented in this paper to time-dependent transformations, a simulation of the two-dimensional flow around a pair of moving circular cylinders was performed, with Reynolds number $Re = 100$. The centre-to-centre distance is 3 diameters, the downstream cylinder is held fixed, and a forced oscillation in the y direction with non-dimensional frequency 0.3 and amplitude of 0.75 diameter was imposed on the upstream cylinder. Instead of using a moving mesh to solve this problem, a fixed mesh where the displacement of the cylinders is zero was used, with a mapping accounting for the movement of the upstream cylinder. Before each time-step, the displacements on the boundaries were used as boundary conditions to solve a Laplace equation, leading to a global representation of the mapping which was used to solve the equations with the semi-implicit method of section 2.2.2. Fig. 6 shows instantaneous contours of vorticity for this case, exhibiting a behaviour that is compatible with what is expected for this flow. The effect

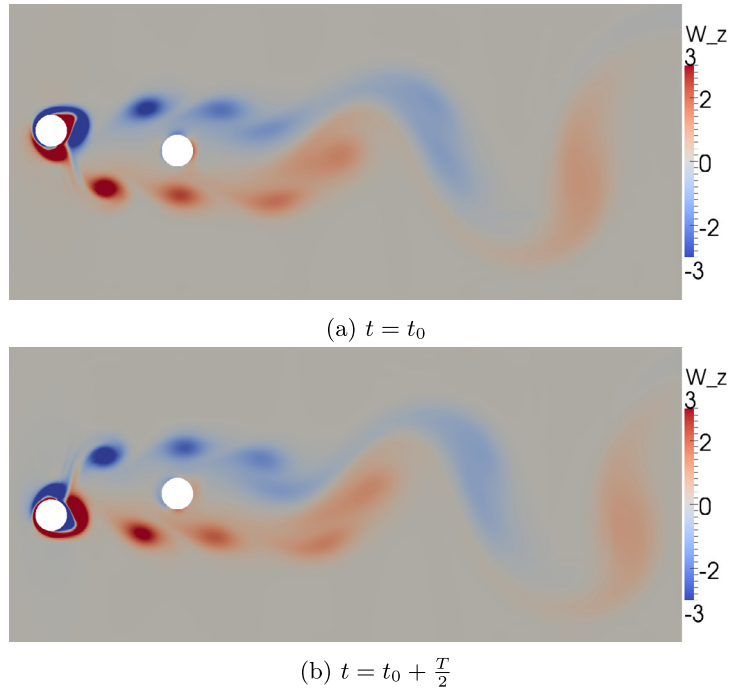


Fig. 6. Instantaneous contours of vorticity for flow around two circular cylinder in tandem with $Re = 100$. The downstream cylinder is fixed, while the upstream cylinder oscillates with frequency 0.3 and amplitude 0.75.

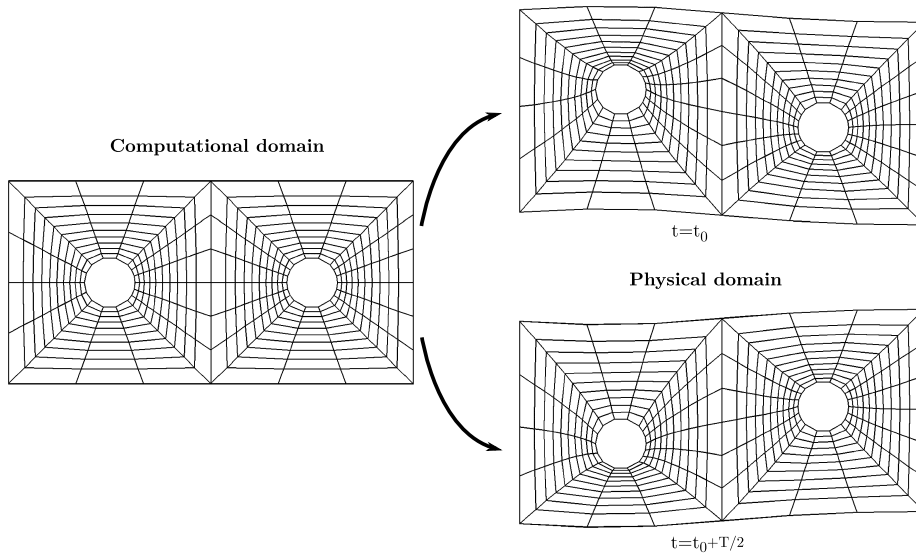


Fig. 7. Detail of computational mesh used in the simulation of the flow around two circular cylinders, and the same mesh after applying the coordinate transformation at two different time instants.

of the transformation is illustrated in Fig. 7, showing the computational mesh in the region close to the cylinders, along with the corresponding representations in the physical domain at two different time instants.

4. Discussion

The computational cost of advancing the equations by one time-step using the semi-implicit method of section 2.2.2 is clearly higher than the one for the explicit approach of section 2.2.1. Therefore, it is important to address the question of under what circumstances the former should be employed. Although section 3.1 showed that both schemes maintain the accuracy of the underlying discretisation, the explicit scheme becomes unstable as the coordinate transformation becomes

more energetic. Therefore, it is reasonable to anticipate that the explicit method is only applicable to less energetic or smaller coordinate changes, while the semi-implicit method with its higher computational cost is more capable of handling more complex or energetic deformations. In these more complex cases, the iterative procedures employed in the semi-implicit method can also become unstable as the deformation of the mapping becomes even larger; however, this issue can be moderated by including a relaxation technique.

To illustrate the limitation of the explicit and semi-implicit schemes, we consider the performance of both methods for the square cylinder case of section 3.2. For this particular case, the maximum amplitude that could be stably handled using the explicit method is $W = 0.25\lambda$ (where $\lambda = 2.8h$), while the semi-implicit method remained stable for a significantly higher amplitude of $W = 0.75\lambda$, after which convergence becomes increasingly difficult. In terms of computational cost, in our implementation the explicit method takes approximately 1.7 times the time required for the straight cylinder case. For the semi-implicit method, the computational cost depends significantly on the number of iterations required in the iterative solutions, with typical cost of between 5 and 10 times that of the undeformed case. Obviously, these increases in cost have to be weighted against the gains obtained by being able to employ a quasi-3D discretisation instead of a full three-dimensional simulation.

When considering the computational costs, it is important to note that the cost is influenced by the particular problem being considered and by the characteristics of the numerical implementation being employed, specially by the balance between the costs of the implicit and explicit parts of the algorithm. In our simulations, the implicit equations were solved using an efficient direct solver with multi-level static condensation, as described in Karniadakis and Sherwin [7]. Because of this, the time spent calculating the advection terms in the original solver was significant (around 30%), emphasizing increases in the cost of the explicit calculations. If the implicit equations were solved using a less efficient iterative solver, as is often the case in large massively-parallel simulations, the solution of the implicit systems would dominate the cost, and thus the performance of our explicit formulation compared to the straight cylinder case would improve. Therefore, for the explicit method the ratio between the cost of computing the mapping terms and computing the advection terms might be a more representative measure of the performance of the method. Determining this ratio, which is related to the number of derivatives that need to be calculated, we observe it to be close to 2.5.

Another important point is that the two methods can be combined, leading to a formulation where only the pressure or only the viscous terms are treated implicitly. This can be useful if evidence indicates that the convergence problems caused by the explicit treatment of mappings with high amplitude are related to only one of these terms. In this situation, only the term leading to stiffness of the system needs to be treated implicitly, while all the other terms remain explicit, reducing the computational cost.

Finally, we note that the example of section 3.3 shows how the methods proposed can lead to a general framework to dealing with fluid-structure interaction without employing moving meshes at each time-step, which can be computationally expensive, due to the requirement of reassembling the pressure and velocity systems to account for the new geometry.

5. Conclusions

We have presented two methods to include generalised coordinate transformations, which can include divergent mappings, to the velocity-correction scheme usually employed in the solution of the unsteady Navier–Stokes equations for incompressible fluids. In one of the methods, all of the terms coming from the mapping are treated explicitly, leading to a lower computational cost. The other method treats the convective terms explicitly, and the pressure and viscous terms implicitly. This requires the use of iterative procedures, making the method more computationally expensive, although it is more robust. Numerical examples demonstrated how these formulations maintain the accuracy of the high-order discretisation employed. These methods are useful when the coordinate transformation leads to a simplification of the geometry, allowing for more efficient numerical techniques such as a Fourier expansion to be employed, compensating for the extra computational cost from computing the mapping terms. Also, by using time-dependent transformations it is possible to solve fluid-structure interaction problems without resorting to moving meshes at each time-step.

Acknowledgements

D.S. and J.R.M. are grateful for the support received from CNPq (grants 231787/2013-8 and 312755/2014-7) and FAPESP (grants 2012/23493-0 and 2014/50279-4). S.J.S. would like to acknowledge support under the Royal Academy of Engineering Research Chair Scheme and support under EPSRC grant EP/K037536/1.

References

- [1] R. Aris, *Vectors, Tensors, and the Basic Equations of Fluid Mechanics*, Dover Publications, Inc., 1989.
- [2] H.A. Carlson, G. Berkooz, J.L. Lumley, Direct numerical simulation of flow in a channel with complex, time-dependent wall geometries: a pseudospectral method, *J. Comput. Phys.* 121 (1995) 155–175.
- [3] R.M. Dakekar, S.J. Sherwin, Flow past a square-section cylinder with a wavy stagnation face, *J. Fluid Mech.* 426 (2001) 263–295.
- [4] J.L. Guermond, J. Shen, Velocity-correction projection methods for incompressible flows, *SIAM J. Numer. Anal.* 41 (1) (2003) 112–134.
- [5] G.E. Karniadakis, Spectral element-Fourier methods for incompressible turbulent flows, *Comput. Methods Appl. Mech. Eng.* 80 (1–3) (1990) 367–380.
- [6] G.E. Karniadakis, M. Israeli, S.A. Orszag, High-order splitting methods for the incompressible Navier–Stokes equations, *J. Comput. Phys.* 97 (2) (1991) 414–443.

- [7] G.E. Karniadakis, S.J. Sherwin, *Spectral/hp Element Methods for Computational Fluid Dynamics*, 2nd edition, Oxford University Press, 2005.
- [8] H. Koberg, *Turbulence control for drag reduction with active wall deformation*, PhD, Imperial College of Science, Technology and Medicine, 2007.
- [9] L.I.G. Kovasznay, Laminar flow behind a two-dimensional grid, *Math. Proc. Camb. Philos. Soc.* 44 (1) (1948) 58–62.
- [10] H. Luo, T.R. Bewley, On the contravariant form of the Navier–Stokes equations in time-dependent curvilinear coordinate systems, *J. Comput. Phys.* 199 (2004) 355–375.
- [11] D.J. Newman, G.E. Karniadakis, A direct numerical simulation study of flow past a freely vibrating cable, *J. Fluid Mech* 344 (1997) 95–136.
- [12] A. Poux, S. Glockner, E. Ahusborde, M. Azaez, Open boundary conditions for the velocity-correction scheme of the Navier–Stokes equations, *Comput. Fluids* 70 (2012) 29–43.