

# New explicit Runge-Kutta methods for the incompressible Navier-Stokes equations

B. Sanderse<sup>\*,\*\*</sup> and B. Koren<sup>\*,\*\*\*</sup>  
Corresponding author: Barry.Koren@cwi.nl

\* Energy research Centre of the Netherlands (ECN), Petten, The Netherlands.

\*\* Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands.

\*\*\* Eindhoven University of Technology, Eindhoven, The Netherlands.

**Abstract:** New explicit Runge-Kutta methods are presented for the time integration of the incompressible Navier-Stokes equations. The differential-algebraic nature of these equations requires that additional order conditions are satisfied compared to the classical order conditions for ordinary differential equations. The methods presented in this work are high-order accurate for both velocity and pressure, even when the boundary conditions or the mesh are time-dependent. Computations for an actuator disk in a time-dependent inflow support the correctness of the analytically derived methods.

*Keywords:* Incompressible Navier-Stokes, Time integration, Temporal accuracy, Runge-Kutta.

## 1 Introduction

In this paper we discuss the application of explicit Runge-Kutta methods to the time discretization of the incompressible Navier-Stokes equations. Explicit Runge-Kutta methods are a popular choice for the time discretization of the Navier-Stokes equations, because they are cheap compared to implicit methods if flow problems are not stiff, which is the case for convection-dominated flows not involving solid boundaries. Compared with (explicit) multi-step methods, Runge-Kutta methods have in general better stability properties, do not have a start-up problem, and easily allow for adaptive time stepping, although they generally require the solution to a Poisson equation for the pressure at each stage of the Runge-Kutta method. Examples of Runge-Kutta methods applied to the incompressible Navier-Stokes equations are Wray's third-order method (sometimes combined with an implicit method for the diffusion terms) [1, 2, 3, 4], a third-order accurate semi-implicit method [5], and the classic fourth-order method [6, 7]. These three- and four-stage methods have the favorable property for convection-dominated flows that the linear stability domain contains part of the imaginary axis.

The application of explicit Runge-Kutta methods to the incompressible Navier-Stokes equations is not straightforward because of the differential-algebraic nature of the equations. It is common practice to explicitly advance the velocity at each stage as if the discretized equations are a system of ordinary differential equations, and subsequently solve a Poisson equation for the pressure to make the velocity field divergence-free. However, it is not clear if and how this approach influences the temporal order of accuracy of the velocity and pressure. The accuracy of the velocity is often silently assumed to be unaffected by the differential-algebraic nature of the incompressible Navier-Stokes equations, and the temporal accuracy of the pressure is often not reported. A temporally accurate pressure is however of interest in many flow simulations, such as those involving unsteady lift and drag computations or fluid-structure interactions. We therefore thoroughly analyze the accuracy of both velocity and pressure by applying the convergence theory developed for index 2 differential algebraic equations [8, 9] to the incompressible Navier-Stokes equations. We discuss the treatment of unsteady boundary conditions and time-varying meshes, which has not been clearly reported in literature so far, and investigate if they influence the order of accuracy.

The outline of this paper is as follows. First, in section 2 we develop a general formulation for explicit Runge-Kutta methods applied to the incompressible Navier-Stokes equations, which includes the case of unsteady boundary conditions for the continuity and momentum equations. Subsequently, in section 3 we evaluate the order conditions for velocity and pressure, and in section 4 we propose different methods to compute a high-order accurate pressure. In section 5 we show the results of two test cases which confirm our theoretical findings.

## 2 Explicit Runge-Kutta methods applied to incompressible Navier-Stokes

### 2.1 Governing equations

The governing equations for incompressible flow are the conservation of mass and momentum,

$$\nabla \cdot \mathbf{u} = 0, \quad (1)$$

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u}, \quad (2)$$

collectively called the incompressible Navier-Stokes equations. For the spatial discretization of (1)-(2) we employ a second-order finite volume method on staggered Cartesian grids, similar to the method of Harlow and Welch [10]. The results from this paper are equally valid though for finite difference and finite element methods, as long as the spatial discretization leads to a semi-discrete system (method of lines) that can be written as:

$$M\dot{u}(t) = r_1(t), \quad (3)$$

$$\Omega \dot{u}(t) = -C(u(t)) + \nu Du(t) - Gp(t) + r_2(u(t), t), \quad (4)$$

supplemented with suitable initial conditions.  $M$ ,  $C$ ,  $D$  and  $G$  represent the discrete divergence, convection, diffusion and gradient operators, respectively.  $r_1(t)$  is a vector with boundary conditions for the continuity equation and  $r_2(u, t)$  is a vector with boundary conditions and forcing terms for the momentum equation.  $\Omega$  is a matrix with on its diagonal the finite volume sizes. In finite element methods this is the mass matrix.

In contrast to the compressible Navier-Stokes equations, equations (3)-(4) are not a set of ordinary differential equations (ODEs), but a set of differential algebraic equations (DAEs). This is due to the incompressibility constraint (3). The *index* of this DAE system is 2 (see e.g. [8, 9, 11]):  $u$  plays the role of the differential variable, and  $p$  the role of the algebraic variable. Following the literature on DAEs, the semi-discrete equations are written in short as

$$0 = g(u, t), \quad (5)$$

$$\dot{u} = f(u, p, t), \quad (6)$$

where  $g(u, t) = Mu - r_1(t)$ ,  $f(u, p, t) = F(u, t) - Gp$  and  $F(u, t) = -C(u) + \nu Du + r_2(u, t)$ .  $\Omega^{-1}$  has been absorbed in the definition of  $C$ ,  $D$ ,  $G$  and  $r_2$ . The explicit presence of unsteady boundary conditions for the divergence equation,  $r_1(t)$ , is often omitted in literature, but will turn out to be an important factor when deriving order conditions. An example of a nonzero  $r_1(t)$  is a time-varying inflow condition such as a turbulent inflow field.

An instantaneous equation for the algebraic variable, the pressure, is found by applying the divergence-free constraint to the momentum equation:

$$Lp = MF(u, t) - \dot{r}_1(t), \quad (7)$$

where  $L = MG$  and we have used that  $M$  is not depending on  $t$ . For more information on time-dependent operators we refer to [12]. The pressure can be eliminated from the system of equations, by solving equation (7) and inserting it into (6):

$$\dot{u} = PF(u, t) + GL^{-1}\dot{r}_1(t), \quad (8)$$

where the projection operator  $P$ , defined by

$$P = I - GL^{-1}M, \quad (9)$$

projects velocity fields onto the space of divergence-free fields; the divergence of this projection is zero ( $MP = 0$ ). Although (8) is now an ODE to which a Runge-Kutta method can be applied, one should *not* start with equation (8). The differentiation of the constraint, necessary to arrive at (8), effectively lowers the index of the DAE system, and upon discretization its solutions do not necessarily satisfy the constraint (5). It is therefore desirable to discretize the original DAE system (5)-(6) (the one with highest index), because its solutions will satisfy all the derived lower index systems [13].

The initial conditions at  $t = t_0$  should be consistent with equations (5)-(6) and (7):

$$Mu_0 = r_1(t_0), \quad (10)$$

$$Lp_0 = MF(u_0, t_0) - \dot{r}_1(t_0). \quad (11)$$

Equation (11) expresses that the initial pressure cannot be chosen freely, but has to be calculated based on  $u_0$ .

## 2.2 Explicit Runge-Kutta methods

Application of an explicit Runge-Kutta method to DAE systems is not always straightforward. A guideline is given by the theory of Hairer et al. [8], who call Runge-Kutta methods for index 2 DAEs *half-explicit Runge-Kutta methods*. Application of a half-explicit Runge-Kutta method to equations (5)-(6) leads to the following method:

$$\tilde{U}_i = u_n + \Delta t \sum_{j=1}^i \tilde{a}_{ij} (\tilde{F}_{j-1} - G\tilde{\psi}_j), \quad (12)$$

$$M\tilde{U}_i = r_1(\tilde{t}_i). \quad (13)$$

Here we have introduced the shifted Butcher tableau  $\tilde{A}$ ,

$$\tilde{A} = \begin{pmatrix} a_{21} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ a_{s1} & \dots & a_{s,s-1} & 0 \\ b_1 & \dots & b_{s-1} & b_s \end{pmatrix}, \quad (14)$$

since the first stage of a Runge-Kutta method is trivial for explicit methods, and we employ the convention

$$\tilde{c}_i = \sum_{j=1}^i \tilde{a}_{ij}. \quad (15)$$

Furthermore, we have the shifted vectors

$$\tilde{U} = \begin{pmatrix} \tilde{U}_1 \\ \vdots \\ \tilde{U}_{s-1} \\ \tilde{U}_s \end{pmatrix} = \begin{pmatrix} U_2 \\ \vdots \\ U_s \\ u_{n+1} \end{pmatrix}, \quad \tilde{p} = \begin{pmatrix} \tilde{p}_1 \\ \vdots \\ \tilde{p}_{s-1} \\ \tilde{p}_s \end{pmatrix} = \begin{pmatrix} p_2 \\ \vdots \\ p_s \\ p_{n+1} \end{pmatrix}. \quad (16)$$

$\tilde{U}_i$  and  $u_n$  are approximations to the exact values  $u(\tilde{t}_i)$  and  $u(t_n)$ , respectively, with  $\tilde{t}_i = t_n + \tilde{c}_i \Delta t$  and  $\tilde{F}_j = F(\tilde{U}_j, \tilde{t}_j)$ .  $A = (a_{ij})$ ,  $b_i$  and  $c_i$  are the coefficients of the Runge-Kutta method. The pressure-like variable  $\psi$  is introduced to distinguish it from the pressure  $p$ , which satisfies

$$L\tilde{p}_i = M\tilde{F}_i - \dot{r}_1(\tilde{t}_i). \quad (17)$$

This equation is generally not satisfied by  $\psi$ .

Now the Runge-Kutta method has been applied, we can again eliminate the pressure, leading to

$$\tilde{U}_i = u_n + \Delta t \sum_{j=1}^i \tilde{a}_{ij} P \tilde{F}_{j-1} + GL^{-1}(r_1(\tilde{t}_i) - r_1(t_n)). \quad (18)$$

Since  $MP = 0$ , equation (18) satisfies  $M\tilde{U}_i = r_1(\tilde{t}_i)$  at all intermediate stages. However, if a Runge-Kutta method were applied to equation (8) instead of equations (3)-(4), the last term in (18) would change to  $GL^{-1} \sum_j a_{ij} \dot{r}_1(t_j)$ , and the constraint is only satisfied if  $\dot{r}_1(t) = 0$  (both formulations are equal in that case). Again, we stress that *the Runge-Kutta method should be applied to the DAE of highest index*.

Upon substituting  $P = I - GL^{-1}M$  into (18) we obtain a formulation that can be used to define a new pressure-like variable (which will be used later to construct a higher order accurate pressure). When comparing

$$\tilde{U}_i = u_n + \Delta t \sum_{j=1}^i \tilde{a}_{ij} \tilde{F}_{j-1} - GL^{-1} \left( \Delta t \sum_{j=1}^i \tilde{a}_{ij} M \tilde{F}_{j-1} - (r_1(\tilde{t}_i) - r_1(t_n)) \right) \quad (19)$$

to the ‘exact’ equation for the pressure at each stage, equation (19) is rewritten as the following two steps:

$$\tilde{U}_i = u_n + \Delta t \sum_{j=1}^i \tilde{a}_{ij} \tilde{F}_{j-1} - \tilde{c}_i \Delta t G \tilde{\phi}_i, \quad (20)$$

with the new pressure-like variable  $\tilde{\phi}_i$  defined by

$$L \tilde{\phi}_i = \sum_{j=1}^i \frac{1}{\tilde{c}_i} \tilde{a}_{ij} M \tilde{F}_{j-1} - \frac{r_1(\tilde{t}_i) - r_1(t_n)}{\tilde{c}_i \Delta t}. \quad (21)$$

Equation (21) is simply the divergence of (20) supplemented with the additional information  $M\tilde{U}_i = r_1(\tilde{t}_i)$ . Each  $\tilde{\phi}_i$  is a Lagrange multiplier to make  $\tilde{U}_i$  divergence free and each  $\tilde{U}_i$  is independent of the value of  $\tilde{\phi}_j$  for  $j \neq i$ . This is the advantage of using  $\phi$  instead of  $\psi$ . It should be stressed that the presence of the  $c$  coefficients in the pressure term is *not* necessary to obtain the correct velocity field. The reason to introduce the  $c$  coefficients is that for explicit methods it yields a  $\tilde{\phi}_i$  which is a consistent approximation to  $\tilde{p}_i$ . The relation between  $\psi$  and  $\phi$  is given by

$$\tilde{A} \tilde{\psi} = \text{diag}(\tilde{c}_1, \dots, \tilde{c}_s) \tilde{\phi}. \quad (22)$$

The use of  $\phi$  instead of  $\psi$  limits the accuracy of the pressure to first order. When comparing equation (21) to (17), the first term on the right side of (21) is recognized as an approximation to  $M\tilde{F}_i$  and the second term as an approximation to  $\dot{r}_1(\tilde{t}_i)$ . The second term is clearly a first-order approximation, since

$$\dot{r}_1(\tilde{t}_i) = \frac{r_1(\tilde{t}_i) - r_1(t_n)}{\tilde{c}_i \Delta t} + \mathcal{O}(\Delta t). \quad (23)$$

The first term is also a first-order approximation, which we show by following an argument employed in [14, 15].  $\tilde{F}_i$  in (17) is  $F$  evaluated at  $(\tilde{U}_i, \tilde{t}_i)$ , whereas  $\sum_{j=1}^i \frac{1}{\tilde{c}_i} \tilde{a}_{ij} \tilde{F}_{j-1}$  is an approximation to the *average value* of  $F$  from  $t_n$  to  $\tilde{t}_i$ . Assuming that  $F$  is continuous over the interval  $[t_n, \tilde{t}_i]$ , this average equals the value of  $F$  at some point  $\hat{t} \in [t_n, \tilde{t}_i]$  (according to the integral version of the mean value theorem) and as such is an  $\mathcal{O}(\Delta t)$  approximation to  $\tilde{F}_i$ :

$$M \tilde{F}_i = \sum_{j=1}^i \frac{1}{\tilde{c}_i} \tilde{a}_{ij} M \tilde{F}_{j-1} + \mathcal{O}(\Delta t). \quad (24)$$

As a consequence the Lagrange multiplier  $\phi$  is a first-order approximation to the pressure  $p$ :

$$\tilde{\phi}_i = \tilde{p}_i + \mathcal{O}(\Delta t). \quad (25)$$

Of course, by virtue of the midpoint method,  $\phi_i$  is a second-order approximation to the pressure at  $t_n + \frac{1}{2}c_i\Delta t$ , as long as the stage order of the method is at least 2 (this will be detailed in section 4). We will call the approach, where one uses  $\tilde{\phi}_s$  as approximation to  $p_{n+1}$ , the ‘standard’ approach. The first-order accuracy of this approach is independent of the particular coefficients of the Runge-Kutta method. It results from the fact that (21) contains an approximation to the integral  $\int F dt$ , whereas (17) contains  $F$  evaluated at a certain time instance. This is because the pressure has an *instantaneous* character: its value is such that the velocity field is divergence free at each time instant, and is independent of the pressure at any previous time. The equation for the velocity is, on the contrary, an evolution equation.

By combining the different  $\phi_i$  values (e.g. by using  $\psi$ ) one can obtain a better than first order accurate approximation to  $p_{n+1}$ . This poses certain requirements on the coefficients of the Butcher tableau, which will be detailed in sections 3 and 4.

To conclude, we write down the solution algorithm that we use in practice, obtained by rewriting (20)-(21):

$$\tilde{V}_i = u_n + \Delta t \sum_{j=1}^i \tilde{a}_{ij} \tilde{F}_{j-1}, \quad (26)$$

$$L\tilde{\phi}_i = \frac{1}{\tilde{c}_i\Delta t} (M\tilde{V}_i - r_1(\tilde{t}_i)), \quad (27)$$

$$\tilde{U}_i = \tilde{V}_i - \tilde{c}_i\Delta t G\tilde{\phi}_i, \quad i = 1, 2 \dots s, \quad (28)$$

optionally followed by equation (22). This sequence of first computing a tentative velocity, then the pressure, and finally correcting the tentative velocity is similar to fractional step methods (see e.g. [14]). However, in fractional step methods the diffusive and/or convective terms are often taken implicitly, and a splitting error then results from uncoupling the solution of velocity and pressure. Here all terms are handled explicitly (except the pressure) and consequently there is no splitting error involved. It is therefore unnecessary to solve a coupled system for  $\tilde{U}_i$  and  $\tilde{\phi}_i$ . The half-explicit nature of the method is now clear: the differential variable is advanced with an explicit method (equations (26) and (28)) while the algebraic variable is handled implicitly (equation (27)). The implicit equation for the pressure has to be solved at each stage, so in total this results in  $s$  Poisson equations. The resulting  $\tilde{U}_s = u_{n+1}$  and  $\tilde{\phi}_s = \phi_{n+1}$  (or  $\tilde{\psi}_s$ ) are approximations to  $u(t_{n+1})$  and  $p(t_{n+1})$ . The order of accuracy of  $\tilde{\psi}_s$  and  $\tilde{U}_s$  will be considered next.

## 3 Order conditions

### 3.1 A short introduction to trees

For general index 2 DAEs of the form (5)-(6) the classical order conditions (‘classical’ referring to non-stiff ODEs, see e.g. [16]) for the coefficients of the Butcher tableau are not sufficient to guarantee the correct order of accuracy for both the differential and algebraic variable. The work of Hairer et al. [8] and Hairer and Wanner [9] provides local and global error analyses for index 2 DAEs and identifies in which cases *order reduction* can occur. We focus on the local error, because for half-explicit methods the error propagation from local to global error is the same as for non-stiff ODEs, see [17].

For Runge-Kutta methods applied to ODEs of the form  $\dot{u} = f(u)$ , the local error can be investigated by expanding both the exact and numerical solution in a Taylor series and comparing until which order they agree. This requires that  $\ddot{u}$ ,  $\dddot{u}$ , etc. are written in terms of  $f$  and its derivatives:

$$\dot{u} = f, \quad \ddot{u} = f_u f, \quad \dddot{u} = f_u f_u f + f_{uu}(f, f). \quad (29)$$

Since  $f$  and  $u$  are vectors, the first derivatives in this expression should be interpreted as Jacobian matrices, the second derivatives as bilinear maps, and  $(f, f)$  as a tensor product. The number of elementary differentials that appear in this process grows rapidly when high orders are compared. With each differential there is an associated order condition. An efficient way to handle the order conditions for ODEs was introduced by Butcher with the concept of rooted trees [16, 18]. Given a certain tree, the elementary differential and the

order condition corresponding to it can be easily written down. For example, (29) becomes in terms of trees

$$\dot{u} = \bullet \quad \ddot{u} = \begin{array}{c} \bullet \\ | \\ \bullet \end{array} \quad \dddot{u} = \begin{array}{c} \bullet \\ | \\ \bullet \\ | \\ \bullet \end{array} + \begin{array}{c} \bullet \quad \bullet \\ \diagdown \quad / \\ \bullet \end{array} \quad (30)$$

The order conditions for these trees are the well-known third order conditions

$$\sum b_i = 1, \quad \sum b_i c_i = \frac{1}{2}, \quad \sum b_i a_{ij} c_j = \frac{1}{6}, \quad \sum b_i c_i^2 = \frac{1}{3}. \quad (31)$$

In all cases, the summation is over all indices present in the summand.

The extension of the analysis with trees to DAEs was done by Hairer et al. [8] and will be used here. Hairer et al. [8] consider the autonomous index 2 DAE

$$0 = g(u), \quad (32)$$

$$\dot{u} = f(u, p). \quad (33)$$

The non-autonomous system (5)-(6) can be written in this form by adding  $\dot{t} = 1$  so that equations (32)-(33) hold by redefining  $u := \begin{pmatrix} u \\ t \end{pmatrix}$  and  $f := \begin{pmatrix} f \\ 1 \end{pmatrix}$ . In the Taylor expansion of the exact and numerical solution,  $\dot{p}$ ,  $\ddot{p}$ , ... appear next to  $\dot{u}$ ,  $\ddot{u}$ , ... Here we list the first few derivatives (see [8, 9]):

$$\dot{u} = f, \quad (34)$$

$$\dot{p} = (-g_u f_p)^{-1} (g_{uu}(f, f) + g_u f_u f), \quad (35)$$

$$\ddot{u} = f_u f + f_p (-g_u f_p)^{-1} (g_{uu}(f, f) + g_u f_u f). \quad (36)$$

For DAEs, the number of differentials grows even more rapidly for higher order derivatives. Trees still provide a compact way to represent these derivatives, when extended to contain both meagre (solid) and fat (open) vertices:

$$\dot{u} = \bullet \quad \dot{p} = \begin{array}{c} \bullet \quad \bullet \\ \diagdown \quad / \\ \circ \end{array} + \begin{array}{c} \bullet \\ | \\ \circ \end{array} \quad \ddot{u} = \begin{array}{c} \bullet \\ | \\ \bullet \end{array} + \begin{array}{c} \bullet \quad \bullet \\ \diagdown \quad / \\ \circ \end{array} + \begin{array}{c} \bullet \\ | \\ \circ \\ | \\ \bullet \end{array} \quad (37)$$

As an example, the above trees correspond to the following order conditions:

$$\dot{u} : \quad \sum b_i = 1, \quad (38)$$

$$\dot{p} : \quad \sum b_i \omega_{ij} \omega_{jk} c_k^2 = 2, \quad \sum b_i \omega_{ij} \omega_{jk} a_{kl} c_l = 1, \quad (39)$$

$$\ddot{u} : \quad \sum b_i c_i = \frac{1}{2}, \quad \sum b_i \omega_{ij} c_j^2 = 1, \quad \sum b_i \omega_{ij} a_{jk} c_k = \frac{1}{2}, \quad (40)$$

the summation being again over all indices. We see that next to the classical order conditions (represented by trees with only meagre vertices) additional order conditions appear, corresponding to trees with fat vertices, which include the inverse of matrix  $A$  ( $\omega_{ij}$  denotes the entries of  $A^{-1}$ ). The order conditions for the  $p$ -component are especially difficult due to the presence of  $(A^{-1})^2$ .

Fortunately, some of these additional trees do not pose additional constraints on the coefficients because they reduce to classical order conditions. For example, the last condition in equation (40) can be written as

$$\sum b_i \omega_{ij} a_{jk} c_k = b^T A^{-1} A c = \sum b_i c_i = \frac{1}{2}, \quad (41)$$

so it reduces to the classical second-order condition. The additional order conditions that cannot be simplified to classical order conditions are of interest to us. To find these remaining conditions we used the software described in [19]. We found that for the  $u$ -component there is no additional tree for order 1, but there is 1 for order 2, there are 4 for order 3 and 17 for order 4. For the  $p$ -component there are 2 trees for order 1, 6 for order 2, 21 for order 3, and 81 for order 4. This large number of additional order conditions can still be considerably reduced when taking into account the specific form of the Navier-Stokes equations.

We stress that the construction of order conditions outlined above holds for general (implicit and explicit) Runge-Kutta methods. For half-explicit methods the construction of the order conditions changes slightly since we work with the shifted Butcher tableau  $\tilde{A}$  instead of  $A$ : if a meagre vertex follows a fat vertex then  $a_{jk}$  changes to  $\tilde{a}_{jk}$  (or  $c_j$  to  $\tilde{c}_j$ ).

### 3.2 Application to the incompressible Navier-Stokes equations

For the spatially discretized Navier-Stokes equations we know that  $f(u, p, t) = F(u, t) - Gp$ , which means that  $f_p = G$  is a constant matrix. All derivatives of  $f_p$ , such as  $f_{pu}$ ,  $f_{pp}$ , etc., are therefore zero, and trees which have a meagre vertex as root or as branch and connected to it a fat vertex and at least one other meagre or fat vertex need not be considered. We note that higher derivatives of  $g_u$ , such as  $g_{uu}$ , do *not* vanish. This is due to the fact that in the case of non-autonomous systems,  $g_{uu}$  consists of  $g_{uu}$ ,  $g_{ut}$  and  $g_{tt}$ .  $g_{uu}$  and  $g_{ut}$  are zero for the Navier-Stokes equations, but  $g_{tt} = \ddot{r}_1(t)$  is in general not, and therefore trees that have a fat vertex with more than one meagre vertex connected to it do not vanish. The special case of  $\dot{r}_1(t) = 0$ , so that  $g_{tt} = 0$ , will be discussed in section 4.3. After removing all trees that contain derivatives of  $f_p$ , and simplifying the resulting order conditions, it turns out that all additional order conditions for the velocity are trivially satisfied, at least up to and including order 4 [12]. For the pressure, on the other hand, a number of additional order conditions remains; these are shown in table 1. Only certain Butcher tableaux will yield a higher-order accurate pressure - this will be detailed in section 4.

A more general case, where the gradient operator  $G(t)$  is time-dependent, can be treated in an analogous fashion. Derivatives of  $f_p$  with respect to  $u$ , such as  $f_{pu}$ , do not vanish. Such a time-dependent gradient operator can appear on meshes that are changing in time. The additional trees that result when  $f_p$  is not constant but depends on time leads to the trees shown in table 2. In contrast to the case  $f_p = \text{constant}$ , the order conditions associated with these trees do *not* reduce to classical order conditions. This table is similar to table 1 in [17], but with the difference that in that work trees containing  $f_{pp}$ ,  $f_{ppp}$  and  $f_{ppu}$  are also present. We present this table here as a reference for practitioners of Runge-Kutta methods for time integration of the incompressible Navier-Stokes equations on time-varying meshes. For a third order method there is one additional tree, denoted by number 7. Evaluating the condition associated with this tree for a three-stage method, together with the four classical order conditions for third order methods, leads to a solution family with  $c_3 = 1$ , and  $c_2$  as a free parameter ( $c_2 \neq 0$ ,  $c_2 \neq \frac{2}{3}$ ,  $c_2 \neq 1$ ):

$$a_{21} = c_2 \qquad a_{31} = \frac{3c_2 - 3c_2^2 - 1}{c_2(2 - 3c_2)} \qquad a_{32} = \frac{1 - c_2}{c_2(2 - 3c_2)}, \quad (42)$$

$$b_1 = \frac{3c_2 - 1}{6c_2} \qquad b_2 = \frac{1}{6c_2(1 - c_2)} \qquad b_3 = \frac{2 - 3c_2}{6(1 - c_2)}. \quad (43)$$

This family excludes Wray's popular third-order method [2]. Wray's method reduces to second order for time-dependent operators, such as moving meshes, and is therefore *not* recommended in this case.

For a fourth order method six additional trees appear due to the time dependency of  $G$ . It is proven in [17] that the order condition corresponding to tree number 12 cannot be satisfied with a four-stage, fourth-order method. An example of a five-stage, fourth-order method that satisfies the conditions corresponding to trees 7-12 is the HEM4 method [17]. It does not satisfy the conditions corresponding to trees 3 and 4, so it is second order accurate for the pressure.

Note that the time-dependence of  $f_p$  also leads to additional trees for the pressure, next to those already mentioned in table 1. They are of order 2 or higher.



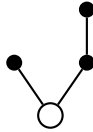


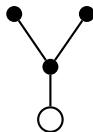
	Tree	Order of tree	Condition	Simplifies to	Differential
1.		1	$\sum b_i \tilde{\omega}_{ij} \tilde{\omega}_{jk} \tilde{c}_k^2 = 2$	$\sum \tilde{\omega}_{si} \tilde{c}_i^2 = 2$	$(-g_u f_p)^{-1} g_{uu}(f, f)$
2.		1	$\sum b_i \tilde{\omega}_{ij} \tilde{\omega}_{jk} \tilde{a}_{kl} c_l = 1$	$c_s = 1$	$(-g_u f_p)^{-1} g_u f_u f$
3.		2	$\sum b_i \tilde{\omega}_{ij} \tilde{\omega}_{jk} \tilde{c}_k \tilde{a}_{kl} c_l = \frac{3}{2}$	$\sum \tilde{\omega}_{si} \tilde{c}_i \tilde{a}_{ij} c_j = \frac{3}{2}$	$(-g_u f_p)^{-1} g_{uu}(f, f_u f)$
4.		2	$\sum b_i \tilde{\omega}_{ij} \tilde{\omega}_{jk} \tilde{c}_k^3 = 3$	$\sum \tilde{\omega}_{si} \tilde{c}_i^3 = 3$	$(-g_u f_p)^{-1} g_{uuu}(f, f, f)$
5.		2	$\sum b_i \tilde{\omega}_{ij} \tilde{\omega}_{jk} \tilde{a}_{kl} a_{lm} c_m = \frac{1}{2}$	$\sum a_{si} c_i = \frac{1}{2}$	$(-g_u f_p)^{-1} g_u f_u f_u f$
6.		2	$\sum b_i \tilde{\omega}_{ij} \tilde{\omega}_{jk} \tilde{a}_{kl} c_l^2 = 1$	$c_s^2 = 1$	$(-g_u f_p)^{-1} g_u f_{uu}(f, f)$

Table 1: Trees and order conditions for  $p$ -component up to and including order 2 when  $f_p = \text{constant}$ .



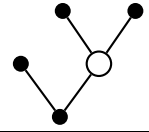
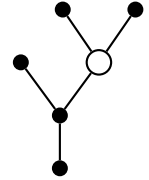
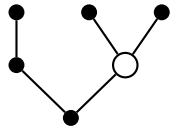
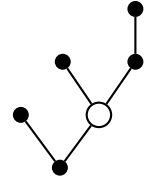
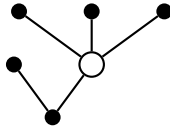
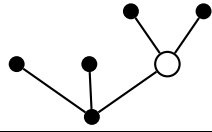
	Tree	Order of tree	Condition	Differential
7.		3	$\sum b_i c_i \tilde{\omega}_{ij} \tilde{c}_j^2 = \frac{2}{3}$	$f_{pu}(f, (-g_u f_p)^{-1} g_{uu}(f, f))$
8.		4	$\sum b_i a_{ij} c_j \tilde{\omega}_{jk} \tilde{c}_k^2 = \frac{1}{6}$	$f_u f_{pu}(f, (-g_u f_p)^{-1} g_{uu}(f, f))$
9.		4	$\sum b_i a_{ij} c_j \tilde{\omega}_{ik} \tilde{c}_k^2 = \frac{1}{4}$	$f_{pu}(f_u f, (-g_u f_p)^{-1} g_{uu}(f, f))$
10.		4	$\sum b_i c_i \tilde{\omega}_{ij} \tilde{c}_j \tilde{a}_{jk} c_k = \frac{3}{8}$	$f_{pu}(f, (-g_u f_p)^{-1} g_{uu}(f, f_u f))$
11.		4	$\sum b_i c_i \tilde{\omega}_{ij} \tilde{c}_j^3 = \frac{3}{4}$	$f_{pu}(f, (-g_u f_p)^{-1} g_{uuu}(f, f, f))$
12.		4	$\sum b_i c_i^2 \tilde{\omega}_{ij} \tilde{c}_j^2 = \frac{1}{2}$	$f_{puu}(f, f, (-g_u f_p)^{-1} g_{uu}(f, f))$

Table 2: Additional trees and order conditions for  $u$ -component up to and including order 4 when  $f_p$  is a function of  $t$ .

## 4 The accuracy of the pressure

### 4.1 Single Butcher tableau for velocity and pressure (Method 1)

In section 2.2 we mentioned that  $\tilde{\phi}_s$  is a first order approximation to  $p_{n+1}$ , but that  $\tilde{\psi}_s$  can be better than first order accurate. We call this the ‘single Butcher tableau’ approach, a name that will be explained in section 4.2. Here we evaluate which order conditions should be satisfied to achieve this. Some of the order conditions in table 1 can be simplified, but in contrast to the  $u$ -component, additional order conditions remain, even for a second-order method. This is not a surprise when considering that the Lagrange multipliers  $\phi$  and  $\psi$  are of a different nature than the pressure  $p$  (integral versus point value).

We should note that all additional order conditions for the pressure can be circumvented *entirely* by solving an additional Poisson equation, equation (7), at  $t_{n+1}$ :

$$Lp_{n+1} = MF_{n+1} - \dot{r}_1(t_{n+1}). \quad (44)$$

Given an  $r$ -th order accurate velocity field  $u_{n+1}$ , the resulting pressure  $p_{n+1}$  is of the same order of accuracy. However, there are two issues in solving equation (44). Firstly it is required that  $r_1(t)$  can be differentiated (analytically or numerically), something which is not required in the computation of  $u$  and  $\phi$ . In many practical computations, for example involving a prescribed turbulent inflow,  $\dot{r}_1(t)$  might not be available. Secondly, solving equation (44) amounts to the solution of an additional Poisson equation, which is computationally costly. We will therefore look at the additional conditions of table 1, which, when satisfied, give a higher order accurate pressure without solving equation (44).

#### 4.1.1 Two-stage methods

For two-stage, second-order methods we have the classical conditions  $b_1 + b_2 = 1$  and  $b_2c_2 = \frac{1}{2}$ . For a second-order accurate pressure the conditions corresponding to trees 1 and 2 have to be satisfied as well. The additional order condition corresponding to tree 1 can be written as

$$\frac{a_{21} - b_1c_2}{a_{21}b_2} = 2. \quad (45)$$

Combining this condition with the order condition for tree 2 ( $c_2 = 1$ ) and the classical order conditions leads to a contradiction. It is therefore not possible to obtain better than first-order accuracy for the pressure with a two-stage explicit method.

#### 4.1.2 Three-stage methods

Butcher [16] lists three cases for which a three-stage, third-order explicit method exists. Only in the ‘case I’ family there is a solution that allows  $c_3 = 1$  (the condition corresponding to tree 2), which is the same as solution family (42)-(43). Evaluating the order condition corresponding to tree 1 for this family leads to

$$\frac{3c_2^2 - 7c_2 + 4}{3c_2 - 2} = 2, \quad (46)$$

which has only one valid solution, being  $c_2 = \frac{1}{3}$ . The resulting Butcher tableau is

$$\begin{array}{c|ccc} 0 & 0 & & \\ \frac{1}{3} & \frac{1}{3} & & \\ 1 & -1 & 2 & \\ \hline & 0 & \frac{3}{4} & \frac{1}{4} \end{array} \quad (47)$$

which satisfies indeed trees 1 and 2. Evaluating equation (22) gives the second order accurate pressure

$$p_{n+1} = \tilde{\psi}_3 = -\frac{3}{2}\tilde{\phi}_1 - \frac{3}{2}\tilde{\phi}_2 + 4\tilde{\phi}_3. \quad (48)$$

The conditions corresponding to trees 3, 4 and 5 are not satisfied, so third order accuracy cannot be achieved.

### 4.1.3 Four-stage methods

For explicit four-stage, fourth-order methods the classical order conditions require  $c_4 = 1$  (see e.g. [16]), so that tree 2 (and 6) are automatically satisfied. We found three methods that also satisfy the condition corresponding to tree number 1; they read:

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 1 & 1 & & \\
 \frac{1}{2} & \frac{3}{8} & \frac{1}{8} & \\
 1 & -\frac{1}{8} & -\frac{3}{8} & \frac{3}{2} \\
 \hline
 & \frac{1}{6} & -\frac{1}{18} & \frac{2}{3} \quad \frac{2}{9}
 \end{array}
 \quad
 \begin{array}{c|ccc}
 0 & 0 & & \\
 \frac{2}{3} & \frac{2}{3} & & \\
 \frac{7}{12} & \frac{91}{192} & \frac{7}{64} & \\
 1 & \frac{1}{7} & -2 & \frac{20}{7} \\
 \hline
 & \frac{5}{28} & -\frac{3}{4} & \frac{48}{35} \quad \frac{1}{5}
 \end{array}
 \quad
 \begin{array}{c|ccc}
 0 & 0 & & \\
 \frac{3}{4} & \frac{3}{4} & & \\
 \frac{5}{9} & \frac{100}{243} & \frac{35}{243} & \\
 1 & \frac{4}{75} & -\frac{19}{21} & \frac{324}{175} \\
 \hline
 & \frac{8}{45} & -\frac{16}{63} & \frac{243}{280} \quad \frac{5}{24}
 \end{array}
 \tag{49}$$

For example, evaluating equation (22) for the left tableau gives

$$p_{n+1} = \tilde{\psi}_4 = \frac{1}{2}\tilde{\phi}_1 - 2\tilde{\phi}_2 - 2\tilde{\phi}_3 + \frac{9}{2}\tilde{\phi}_4. \tag{50}$$

As can be readily calculated, none of the above methods satisfies the conditions corresponding to trees 3, 4 and 5. Therefore, with a four-stage, fourth-order explicit method the pressure is, again, at best second-order accurate.

## 4.2 Reconstructing instantaneous pressure values from time averages (Method 2)

We mentioned in section 2.2 that  $\tilde{\phi}_i$  defined by (21) is only first-order accurate in time but found that second order accurate pressures  $\tilde{\psi}_i$  are possible with the methods mentioned in section 4. In this section we propose a different approach by reconstructing the point value  $p_{n+1}$  from the integral averages  $\tilde{\phi}_i$ . This approach can be seen as a generalization of equation (18) by introducing a separate Butcher tableau  $\tilde{A}^p$  for the pressure term:

$$\tilde{U}_i = u_n + \Delta t \sum_{j=1}^i \tilde{a}_{ij} \tilde{F}_{j-1} - \Delta t \sum_{j=1}^i \tilde{a}_{ij}^p G \tilde{\psi}_j. \tag{51}$$

Now the name of Method 1, which uses  $A^p = A$ , is clear: a single Butcher tableau for velocity and pressure. In practice it is not necessary to derive  $A^p$  completely; we only need its last row. This will be detailed below.

First we consider the *exact* integration of equation (7) from  $t_n$  to  $\tilde{t}_i$ , which reads

$$L \int_{t_n}^{\tilde{t}_i} p(t) dt = M \int_{t_n}^{\tilde{t}_i} F(t) dt - (r_1(\tilde{t}_i) - r_1(t_n)), \tag{52}$$

and we denote the exact average of  $p$  over this interval by  $\phi(\tilde{t}_i)$  (the exact counterpart of the approximation  $\tilde{\phi}_i$ ):

$$\phi(\tilde{t}_i) = \frac{1}{\tilde{c}_i \Delta t} \int_{t_n}^{\tilde{t}_i} p(t) dt. \tag{53}$$

The challenge is to find a higher order accurate point value  $p_{n+1}$  from the time average values  $\phi(\tilde{t}_i)$ . Such an approximation of point values from integral averages is well-known in the field of Essentially Non-Oscillatory (ENO) conservative finite difference schemes and is called *reconstruction*. We follow [20] to perform this reconstruction, and refer to that work for more details. Denoting the primitive function of  $p(t)$  by  $P(t)$ , we can write

$$\phi(\tilde{t}_i) (\tilde{c}_i \Delta t) = \int_{t_n}^{\tilde{t}_i} p(t) dt = P(\tilde{t}_i) - P(t_n), \quad i = 1, 2, \dots, s. \tag{54}$$

We then construct an interpolation polynomial  $H(t)$  through the abscissa ( $c$ -values) of the Runge-Kutta

method. The derivative of  $H(t)$ ,  $h(t)$ , is an approximation to  $p(t)$  (for details, see [12]):

$$h(t) = \sum_{k \in K'} \phi(\tilde{t}_k) \tilde{c}_k \Delta t \ell'_k(t), \quad (55)$$

where  $\ell_k$  are Lagrange polynomials,

$$\ell_k(t) = \prod_{j \in K, j \neq k} \frac{t - \tilde{t}_j}{\tilde{t}_k - \tilde{t}_j}, \quad (56)$$

and  $K = \{k_1, \dots, k_m\}$  is the set of points that will be used in the interpolation. Given the values  $\phi(\tilde{t}_k)$  and the points  $\tilde{t}_k$ , equation (55) can be evaluated at  $t_{n+1}$ , which provides the approximation we are looking for:

$$p_{n+1} = h(t_{n+1}). \quad (57)$$

In practice we cannot use the *exact* average  $\phi(\tilde{t}_i)$  to find  $p_{n+1}$ .  $\phi(\tilde{t}_i)$  is approximated by  $\tilde{\phi}_i$ , whose order of accuracy depends on the *stage order* of the method. This stage order can be expressed by making use of the so-called *simplifying condition*  $C_i(\xi)$  [21], which reads in terms of the shifted Butcher tableau:

$$\tilde{C}_i(\xi) : \quad \sum_{j=1}^s \tilde{a}_{ij} c_j^{k-1} = \frac{1}{k} \tilde{c}_i^k \quad (1 \leq k \leq \xi). \quad (58)$$

If  $\tilde{C}_i(q)$  holds then polynomials of degree lower than  $q$  are exactly interpolated at stage  $i$ . With the notation of the shifted tableau,  $\tilde{C}_s(q)$  expresses the order of quadrature of the final step in the Runge-Kutta method, so  $\tilde{C}_s(s)$  trivially holds for an  $s$ -stage,  $s$ -th order method. Assuming that  $\tilde{C}_i(q)$  holds, the equation for  $\phi(\tilde{t}_i)$  can be written as

$$L\phi(\tilde{t}_i) = \frac{1}{\tilde{c}_i} \sum_{j=1}^i \tilde{a}_{ij} M F_j - \frac{r_1(\tilde{t}_i) - r_1(t_n)}{\tilde{c}_i \Delta t} + \mathcal{O}(\Delta t^q), \quad (59)$$

so that the difference between the exact integral and its numerical approximation is  $\mathcal{O}(\Delta t^q)$ :

$$\phi(\tilde{t}_i) = \tilde{\phi}_i + \mathcal{O}(\Delta t^q). \quad (60)$$

To summarize, it is possible to obtain a higher order accurate pressure at  $t_{n+1}$  by combining the average values  $\tilde{\phi}_i$  from the different stages. To attain a certain order  $r$  requires at least  $r$  distinct stages (i.e. with different  $c_i$ ), and each individual stage  $i$  should have stage order  $r$ , i.e., satisfy  $C_i(r)$ . We will now check if this is possible for methods with two, three and four stages.

#### 4.2.1 Two-stage methods

For two-stage, second-order methods  $\tilde{C}_2(2)$  is obviously satisfied, but  $\tilde{C}_1(2)$  cannot be satisfied because the equation for  $\tilde{U}_1$  is simply a Forward Euler step, which is first-order accurate (this is always the case for explicit methods).

#### 4.2.2 Three-stage methods

Since  $\tilde{C}_1(2)$  cannot be satisfied, we require  $\tilde{C}_2(2)$  to be satisfied, i.e.,

$$a_{32}c_2 = \frac{1}{2}c_3^2, \quad (61)$$

together with the condition  $c_3 \neq 1$  to have distinct  $c$ 's. Using the third-order conditions

$$b_3 a_{32} c_2 = \frac{1}{6}, \quad b_2 c_2^2 + b_3 c_3^2 = \frac{1}{3}, \quad (62)$$

this leads to  $b_1 = \frac{1}{4}$ ,  $b_2 = 0$ ,  $b_3 = \frac{3}{4}$  and  $c_3 = \frac{2}{3}$ .  $c_2$  can be chosen freely ( $\neq 0$ ), and then determines  $a_{31}$  and  $a_{32}$ . Wray's popular third-order method [2] falls in this category, with  $c_2 = \frac{8}{15}$ :

$$\begin{array}{c|cc} 0 & 0 & \\ \frac{8}{15} & \frac{8}{15} & \\ \frac{2}{3} & \frac{1}{4} & \frac{5}{12} \\ \hline & \frac{1}{4} & 0 & \frac{3}{4} \end{array} \quad (63)$$

Another possibility is to take  $c_2 = c_3$ , which saves an evaluation of boundary conditions and forcing terms:

$$\begin{array}{c|cc} 0 & 0 & \\ \frac{2}{3} & \frac{2}{3} & \\ \frac{2}{3} & \frac{1}{3} & \frac{1}{3} \\ \hline & \frac{1}{4} & 0 & \frac{3}{4} \end{array} \quad (64)$$

The interpolation polynomial  $h(t)$  from equation (55) is independent of  $c_2$  and given by:

$$h(t) = - \left( \frac{2 \frac{t-t_n}{\Delta t} - 1}{1 - c_3} \right) \tilde{\phi}_2 + \left( \frac{2 \frac{t-t_n}{\Delta t} - c_3}{1 - c_3} \right) \tilde{\phi}_3, \quad (65)$$

and  $p_{n+1}$  follows with  $c_3 = \frac{2}{3}$  as

$$p_{n+1} = h(t_{n+1}) = -3\tilde{\phi}_2 + 4\tilde{\phi}_3. \quad (66)$$

This equation provides a new way to obtain a second-order accurate pressure by combining two first-order accurate pressures of a three-stage method. It is also valid when Wray's method is used only for the convective terms and an appropriate implicit method for the diffusive terms, such as the method from [1].

If one wants to maximize stability instead of order of accuracy (second-order accuracy is sufficient in many practical applications), one can use three-stage methods that are second order for the velocity. Combining the condition for maximum stability along the imaginary axis ( $b_3 a_{32} c_2 = \frac{1}{4}$ ) with condition (61) yields a family of methods with  $c_2$  and  $c_3$  as free parameters. An example of a low-storage method satisfying these conditions is presented in Perot and Nallapati [22], but it has  $c_3 = 1$  so a second-order accurate pressure cannot be obtained. We propose the following alternative method

$$\begin{array}{c|cc} 0 & 0 & \\ \frac{1}{2} & \frac{1}{2} & \\ \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \hline & 0 & -1 & 2 \end{array} \quad (67)$$

which we obtained by requiring  $b_1 = 0$  and  $c_2 = c_3$ . The requirement  $b_1 = 0$  leads to the same storage requirements as [22], and  $c_2 = c_3$  has the advantage that only one intermediate boundary condition evaluation is needed.

#### 4.2.3 Four-stage methods

Four-stage, fourth-order methods have  $c_4 = 1$ , so that even if  $\tilde{C}_3(2)$  would hold, it cannot be used, because the abscissa in the reconstruction should be distinct. We therefore look again for methods that satisfy  $\tilde{C}_2(2)$ . Combining condition (61) with the fourth-order conditions leads to  $c_3 = \frac{1}{2}$  and two families of solutions

result, corresponding to the ‘case II’ and ‘case IV’ solutions found by Kutta [16]:

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 c_2 & c_2 & & \\
 \frac{1}{2} & \frac{1}{2} - \frac{1}{8c_2} & \frac{1}{8c_2} & \\
 1 & \frac{1}{2c_2} - 1 & -\frac{1}{2c_2} & 2 \\
 \hline
 & \frac{1}{6} & 0 & \frac{2}{3} \quad \frac{1}{6}
 \end{array} \quad c_2 \neq 0, \quad (68)$$

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 1 & 1 & & \\
 \frac{1}{2} & \frac{3}{8} & \frac{1}{8} & \\
 1 & 1 - \frac{1}{4b_4} & -\frac{1}{12b_4} & \frac{1}{3b_4} \\
 \hline
 & \frac{1}{6} & \frac{1}{6} - b_4 & \frac{2}{3} \quad b_4
 \end{array} \quad b_4 \neq 0. \quad (69)$$

The upper tableau with  $c_2 = c_3 = \frac{1}{2}$  is attractive because it requires only one intermediate evaluation of boundary conditions and forcing terms. In a similar fashion as the three-stage method (equation (65) with  $\tilde{\phi}_3$  replaced by  $\tilde{\phi}_4$ ),  $p_{n+1}$  follows as

$$p_{n+1} = -2\tilde{\phi}_2 + 3\tilde{\phi}_4. \quad (70)$$

Again, this equation provides a new way to obtain a second-order accurate pressure by combining two first-order accurate pressures of a four-stage method. The number of methods that are allowed by tableaux (68)-(69) is much larger than with the tableaux from method 1 (see equation (49)).

### 4.3 Steady boundary conditions for the continuity equation (Method 3)

An important case for the incompressible Navier-Stokes equations is when the boundary conditions for the continuity equation are steady, i.e., equation (3) can be written as

$$Mu = r_1, \quad (71)$$

where  $r_1$  is independent of  $t$ . Equation (5) then reads  $g(u) = 0$ , with  $g$  linear in  $u$ . This means that  $g_u = \text{constant}$  and all partial derivatives of  $g_u$  ( $g_{uu}$ ,  $g_{ut}$ ,  $g_{tt}$ , etc.) are zero. As before, all additional trees for the  $u$ -component disappear, but most of the trees for the  $p$ -component (table 1) also vanish. Only trees 2, 5 and 6 remain, and it is possible to find higher order accurate methods for the pressure. For example, with two stages a second-order accurate pressure is possible ( $c_2 = 1$ ), and with four stages a third-order accurate pressure is possible. However, it is not necessary to consider such methods, because in the case  $g_u = \text{constant}$  the pressure can be computed to the same order of accuracy as the velocity, without additional cost. This can be seen by comparing equation (21) for  $i = 1$  with equation (44):

$$Lp_n = MF_n - \dot{r}_1(t_n), \quad (72)$$

$$L\tilde{\phi}_1 = \frac{\tilde{a}_{11}}{\tilde{c}_1} MF_1 - \frac{r_1(\tilde{t}_1) - r_1(t_n)}{\tilde{c}_1 \Delta t}. \quad (73)$$

Considering that  $F_1$  is equal to  $F_n$  (determined at the end of the previous time step) and that  $\tilde{a}_{11} = \tilde{c}_1$ , these expressions are equal if  $r_1$  is independent of time (or a linear function of  $t$ ). This means that  $\tilde{\phi}_1$  is actually the  $r$ -th order accurate pressure at  $t_n$ , and the additional Poisson solve associated with equation (44) can be avoided. An existing implementation could remain unaltered; instead of taking  $\tilde{\phi}_s = \phi_{n+1}$ , the pressure that makes  $u_{n+1}$  divergence free, one should take  $\tilde{\phi}_1$  from the next time step to have a higher order accurate pressure at the end of the current time step. We prefer to compute  $p_{n+1}$  and then skip the computation of  $\tilde{\phi}_1$  in the next time step. This works for any explicit Runge-Kutta method with at least two stages, thus providing a simple way to improve the temporal accuracy of the pressure without increasing computational cost.

## 4.4 Summary

We classify the methods analyzed so far as follows. The methods that were derived in section 4.1 will be indicated by M1, the methods derived in section 4.2 are indicated by M2, and in case section 4.3 applies we write M3. We then write  $MmSsRr$  to indicate that an  $s$ -stage explicit Runge-Kutta method of type  $m$  is used with order  $s$  for the velocity and order  $r$  for the pressure. For example, there are three methods of type M1S4R2 and they are given by the tableaux in (49). In case  $m = 3$ , we can always make  $r$  equal to  $s$  with the approach of section 4.2, and any existing  $s$ -stage,  $s$ -th order method can be used. In case  $r = 1$  we have the standard approach with  $p_{n+1} = \tilde{\phi}_s$ , which can be used with any method.

## 5 Results

### 5.1 Taylor vortex

The Taylor-Green vortex in two dimensions is an exact solution to the Navier-Stokes equations:

$$u(x, y, t) = -\sin(\pi x) \cos(\pi y) e^{-2\pi^2 t / \text{Re}}, \quad (74)$$

$$v(x, y, t) = \cos(\pi x) \sin(\pi y) e^{-2\pi^2 t / \text{Re}}, \quad (75)$$

$$p(x, y, t) = \frac{1}{4}(\cos(2\pi x) + \cos(2\pi y))e^{-4\pi^2 t / \text{Re}}. \quad (76)$$

The domain on which we define the solution is the square  $[\frac{1}{4}, 2\frac{1}{4}] \times [\frac{1}{4}, 2\frac{1}{4}]$  with either time-dependent Dirichlet or periodic boundary conditions. Prescribing both inflow and outflow with Dirichlet conditions is normally not a good idea, see e.g. [23], but we did not find any negative effects in this test case. We deliberately do not take the square  $[0, 2] \times [0, 2]$  (or  $[-1, 1] \times [-1, 1]$ ) as domain, because it leads to  $\mathbf{u} \cdot \mathbf{n} = 0$  on the entire boundary, meaning that  $r_1(t) = 0$ . In all simulations we use  $\text{Re} = 100$  and we integrate from  $t = 0$  to  $t = 1$ . The Poisson equation is solved by LU-decomposition of  $L$ , which is the most efficient way for this relatively small problem.

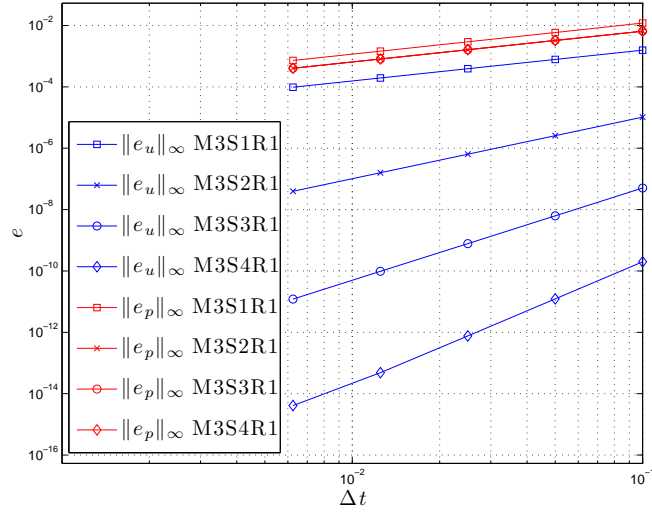
We take a coarse mesh with  $20 \times 20$  volumes and vary the time step to investigate the temporal accuracy. The spatial error clearly overwhelms the temporal error and in order to compute the latter, we subtract the solution from a simulation with a small time step ( $\Delta t = 10^{-3}$ ), so that the spatial error is effectively eliminated.

The first test concerns periodic boundary conditions, such that the observations from section 4.3 apply and the methods are characterized as M3. Four  $s$ -stage,  $s$ th-order Runge-Kutta methods are tested, with  $s = 1, 2, 3, 4$ . For  $s = 1$  we take Forward Euler, for  $s = 2$  modified Euler (explicit trapezoidal, Heun's method), for  $s = 3$  Wray's method, and for  $s = 4$  the classical fourth-order method. In all cases the number of Poisson solves is the same as the number of stages. Figure 1a shows that with our current approach both pressure and velocity attain the classical order of convergence, whereas the standard method ( $p_{n+1} = \tilde{\phi}_s$ ) leads to only first order convergence of the pressure, see figure 1b. The velocity error is unaffected by the accuracy of the pressure. For this very smooth test case the error of higher order methods does not only converge faster upon time step refinement (as predicted by theory), but the magnitude of the error for the largest time step is also much smaller. We only show here the  $L_\infty$ -norm because the  $L_2$ -norm shows exactly the same behavior.

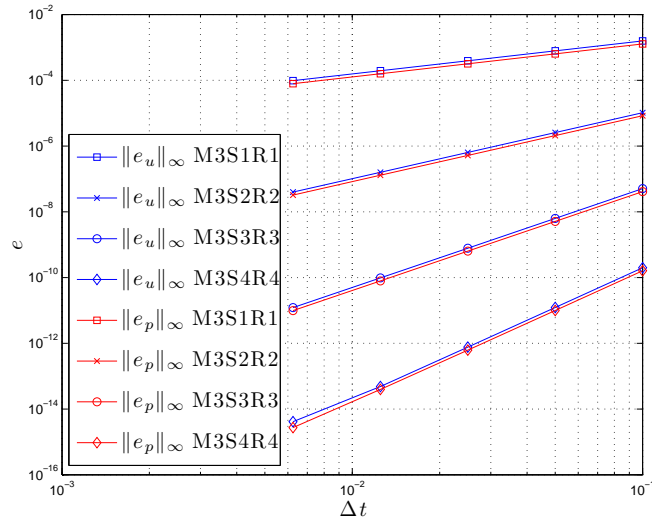
The second test concerns unsteady Dirichlet conditions with methods of type M1. For  $s = 1$  and  $s = 2$  only  $r = 1$  is possible, so we focus on  $s = 3$  and  $s = 4$  with  $r = 2$ . For M1S3R2 the only solution is (47) with (48) for the pressure, for M1S4R2 we take the first tableau in (49) (because it has the simplest coefficients) and (50) for the pressure.

The third test concerns unsteady Dirichlet conditions with methods of type M2. As for methods of type M1, we focus on  $s = 3$  and  $s = 4$ . For M2S3R2 we take Wray's method with (66) for the pressure and for M2S4R2 we take (68) with  $c_2 = \frac{1}{4}$  and (70) for the pressure.

Figures 2a and 2b then show the order of accuracy of the velocity and pressure for these two methods, in case of the 'standard' approach (R1), our approach (R2) and in case of an additional Poisson solve (R3 or R4). The additional Poisson solve can be performed because the explicit dependence of  $r_1$  on  $t$  is known so that  $\dot{r}_1(t)$  can be calculated. The velocity error is in all cases again independent of the particular approach



(a) Standard method

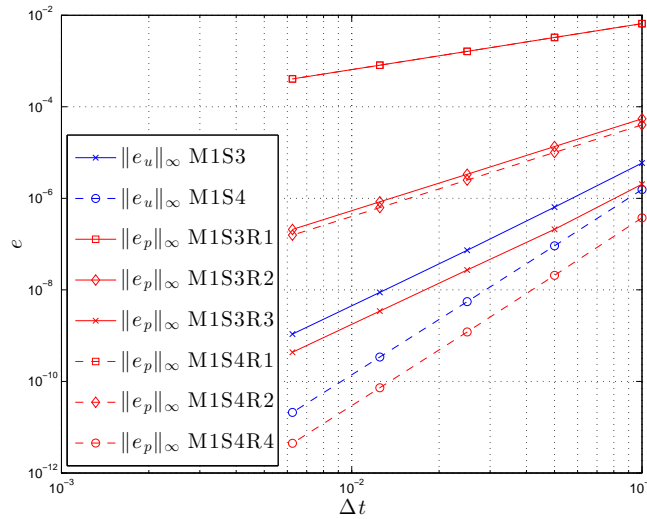


(b) New method for steady boundary conditions

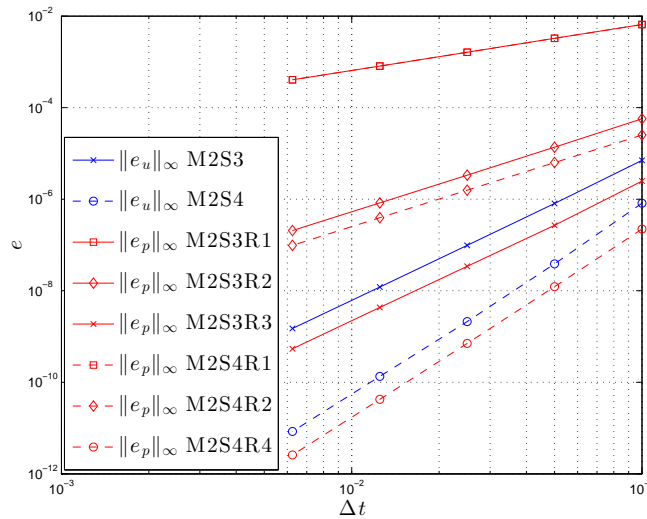
Figure 1: Convergence of temporal error for Taylor-Green problem with steady boundary conditions.



for the pressure. It is confirmed that both methods M1 and M2 indeed lead to a second-order accurate pressure when the proper Butcher tableaux are chosen. The difference in accuracy between the results of M1 and M2 is small, but this depends on the test case under consideration. The computational effort is for both very similar. These second-order schemes greatly improve the accuracy with respect to the standard first-order approach without additional cost. The effort of an additional Poisson solve can only be justified in case higher-order accurate (third or fourth order) pressure solutions are required.



(a) New method, with single Butcher array for velocity and pressure (M1)



(b) New method, with reconstruction of instantaneous pressure values from time averages (M2)

Figure 2: Convergence of temporal error for Taylor-Green problem with unsteady Dirichlet boundary conditions.

## 5.2 An actuator disk in an unsteady inflow field

A practically relevant situation with a temporally varying inflow appears when simulating the flow of air through wind turbines operating in a turbulent atmospheric wind field. Such simulations of wind-turbine wakes can contribute to the understanding of the effect of wakes on power production and blade loading. Different methods exist for modeling the effect of the wind turbines on the flow, such as actuator methods and direct methods [24]. In the former, the action of the turbine is modeled with a body force, such as an actuator surface, and in the latter the action of the turbine is modeled by computing the actual flow around it, for example by applying a body fitted grid around the turbine blades. In both cases accurate knowledge of the pressure field is necessary to calculate the forces that act on the blade, such as lift and drag.

Here we study a simplified case of an actuator disk in a laminar flow. The domain is  $[0, 10] \times [-2, 2]$ , the Reynolds number is 100 and the thrust coefficient of the turbine is  $C_T = \frac{1}{2}$ . The actuator disk is located at  $x = 2$  and has unit length, see figure 3. On all boundaries, except the inflow boundary at  $x = 0$ , we prescribe the following outflow conditions (see e.g. [23]):

$$y = -2, 2 : \quad \frac{\partial u}{\partial y} = 0 \quad p - \frac{1}{\text{Re}} \frac{\partial v}{\partial y} = p_\infty \quad (77)$$

$$x = 10 : \quad p - \frac{1}{\text{Re}} \frac{\partial u}{\partial x} = p_\infty \quad \frac{\partial v}{\partial x} = 0. \quad (78)$$

For a verification study of the actuator disk in a laminar flow with steady inflow and these boundary conditions we refer to [25]. In the current test, the inflow conditions are given by:

$$x = 0 : \quad u_b(t) = \cos \alpha(t), \quad v_b(t) = \sin \alpha(t), \quad (79)$$

where  $\alpha(t) = \frac{\pi}{6} \sin(t/2)$ . This describes a time-varying inflow with constant magnitude but changing direction, see figure 4a.

First we perform a simulation from  $t = 0$  to  $t = 4\pi$ , with a uniform mesh having  $200 \times 80$  volumes ( $\Delta x = \Delta y = 1/20$ ) and 10,000 time steps ( $\Delta t = 4\pi/10000$ ), using the M2S4R4 method of equation (68), again with  $c_2 = \frac{1}{4}$ . We focus on methods of type M2, because they still allow for some freedom in the choice of the coefficients of the Butcher tableau, in contrast to methods of type M1. In figure 4b the normalized kinetic energy of the flow (integrated over the entire domain) is shown, from which it can be concluded that the flow becomes periodic with period  $2\pi$  after approximately  $t = 4\pi$ . The velocity and pressure field at this time instant are shown in figures 5 and 6. The wake has been deflected downwards due to the inflow with negative  $v_b$  that was present from  $t = 2\pi$  to  $4\pi$ . The presence of the actuator disk is clearly seen in the pressure contours; they are discontinuous across the disk.

Due to the very small time step, these velocity and pressure fields have a negligible temporal error compared to the spatial error, and are therefore used to compute the temporal error in the velocity and pressure field for larger time steps. The resulting convergence of the velocity and pressure error is shown in figure 7, for methods (63) and (68). As before, we see that the velocity attains its classical order of accuracy, i.e., third order for the three-stage method, and fourth order for the four-stage method. The pressure can be computed to the same order as the velocity, but this requires an additional Poisson solve and an expression for  $\dot{r}_1(t)$ . Since  $\dot{r}_1(t)$  contains only the normal velocity component on the boundary, it is sufficient to derive the expression for  $\dot{u}_b(t)$ :

$$\dot{u}_b(t) = -\frac{\pi}{12} \sin(\alpha(t)) \cos(t/2). \quad (80)$$

On the other hand, the standard approach is only first order and starts with a large error at large time steps. Our proposed approach, corresponding to the lines M2S3R2 and M2S4R2, does not require any significant additional computational effort (no additional Poisson solve, no evaluation of  $\dot{r}_1(t)$ ), it clearly shows second-order accuracy and starts with a small error already at the largest time step considered. This time step,  $\Delta t = 4\pi/200$ , is the largest step for which stable solutions could be obtained. It is determined by the convective terms, showing the benefit of explicit Runge-Kutta methods for this test case.

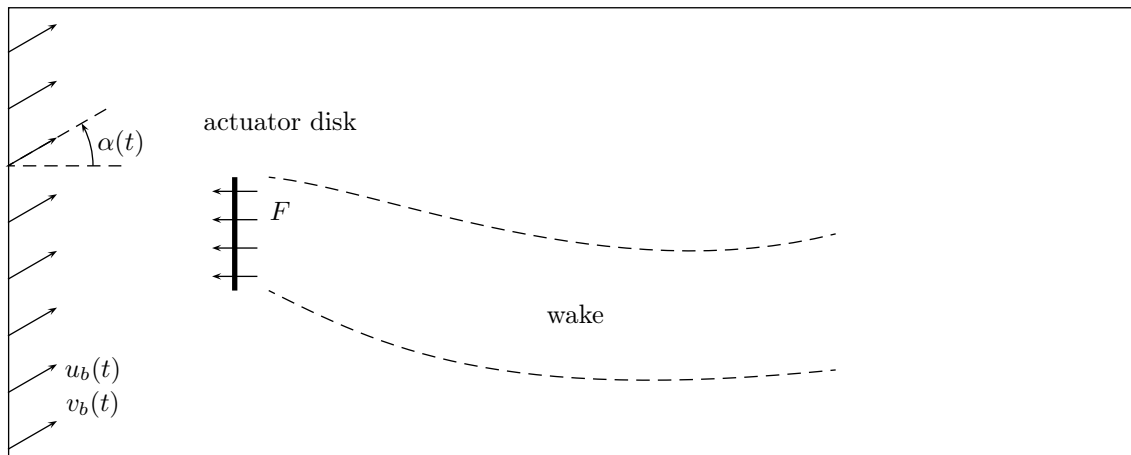
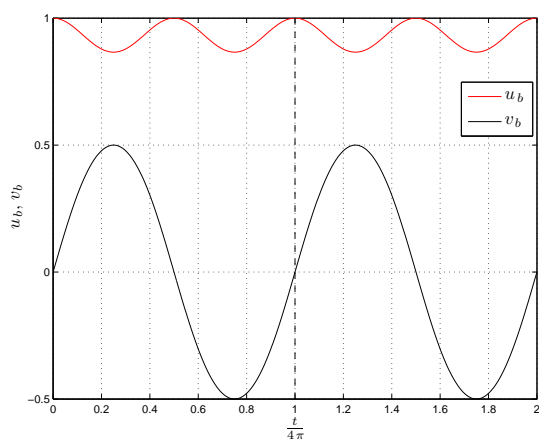
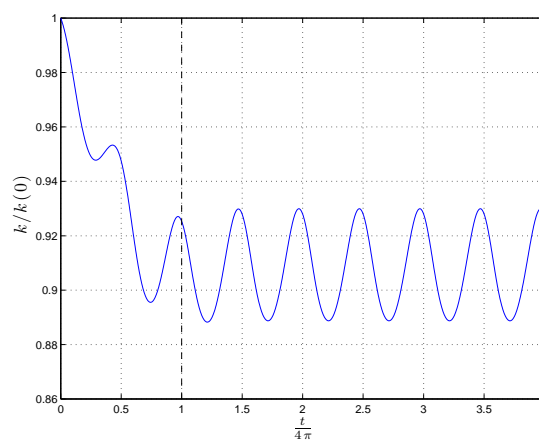


Figure 3: Actuator disk in an unsteady inflow field.



(a) Velocity components



(b) Total kinetic energy

Figure 4: Time evolution of inflow and total kinetic energy.

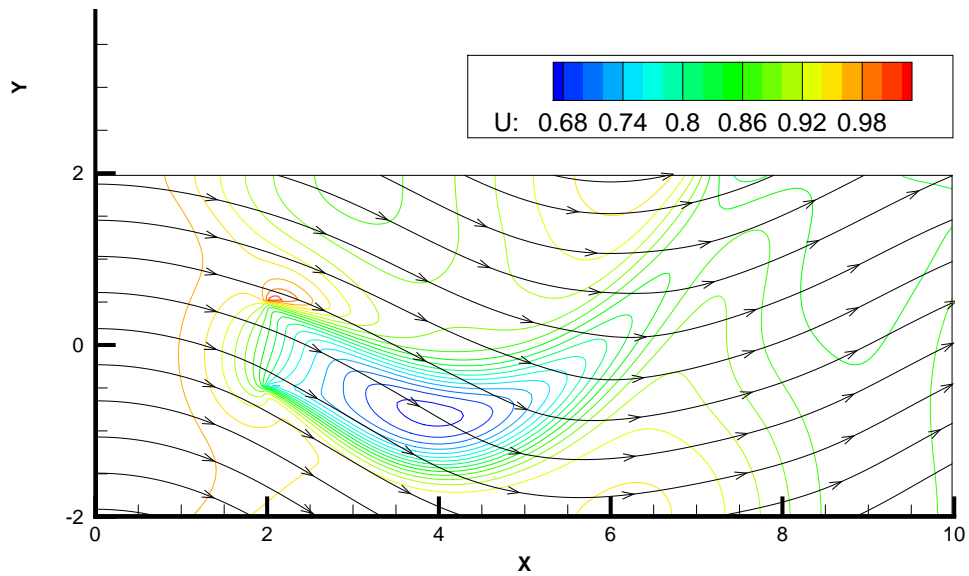


Figure 5: Streamlines and  $u$ -contour lines at  $t = 4\pi$ .

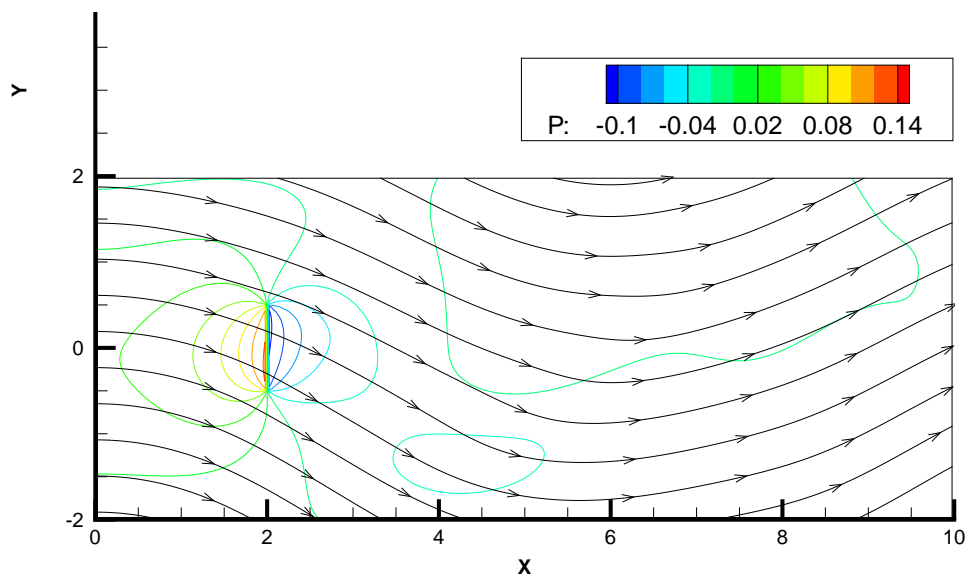


Figure 6: Streamlines and  $p$ -contour lines at  $t = 4\pi$ .

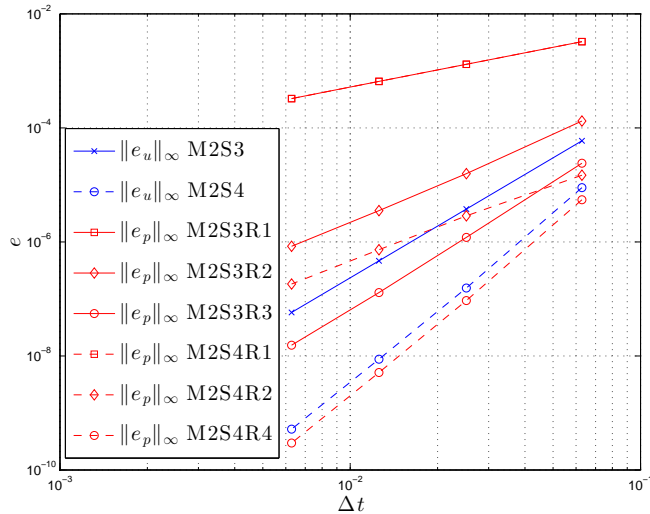


Figure 7: Velocity and pressure error at  $t = 4\pi$  for a selection of methods.

## 6 Conclusions

In this paper we have analyzed the temporal order of accuracy of the velocity and pressure when explicit Runge-Kutta methods are applied to the incompressible Navier-Stokes equations. It is shown that the order of accuracy of the velocity is not affected by the differential-algebraic nature of the incompressible Navier-Stokes equations and is therefore the same as for non-stiff ordinary differential equations. However, if the semi-discrete equations involve time-dependent operators, then additional order conditions appear for orders higher than two. These conditions restrict three-stage, third-order methods to a one-parameter family of methods, to which the popular method of Wray does not belong. Four-stage, fourth-order methods for time-varying operators *do not exist*, and one has to resort to five stages to achieve fourth order.

The pressure always suffers from the problem that upon time-stepping a time-average pressure is computed, instead of a point value. Therefore, achieving higher than first order accuracy for the pressure imposes additional conditions on the coefficients of the Runge-Kutta method compared to the classical order conditions. Fortunately, if the boundary conditions for the continuity equation are independent of time, then the pressure can be determined to the same order of accuracy as the velocity, without requiring an additional solution of a Poisson problem. However, if the boundary conditions for the continuity equation depend on time, then additional order conditions for the pressure appear. These are not satisfied by most existing explicit Runge-Kutta methods, so that the pressure is typically only first-order accurate in time. Second-order accuracy can be achieved by only one three-stage, and only three four-stage methods.

A new approach is to reconstruct instantaneous pressure values from time-average values. We showed that this reconstruction, based on Lagrange polynomials, can be of the same order as the number of stages, but that the stage order of the method limits the accuracy of the pressure. These methods can be interpreted as having a different Butcher tableau for velocity and pressure, in contrast to the foregoing single-Butcher array approach. Three- and four-stage methods with second-order stage order were derived, leading to a much larger class of methods that have second-order accuracy for the pressure. Furthermore, a distinct advantage of this new class of methods is that they can be directly applied to implicit and implicit-explicit (IMEX) Runge-Kutta methods as well.

In all cases considered here third- or fourth-order accuracy could not be obtained with a three- or four-stage method without resorting to an additional Poisson solve. Such an additional solve is not always straightforward in practical computations, because it requires the derivative of the boundary conditions for the continuity equation with respect to time.

To conclude, we think that the ‘best’ explicit Runge-Kutta method for many incompressible Navier-Stokes problems is a three-stage method of type M2, that is third order for the velocity and second order for the

pressure. It combines stability (includes the imaginary axis), accuracy and flexibility ( $c_2$  can still be chosen, in contrast to the three-stage method of type M1). A fourth-order method might lead to unnecessarily accurate solutions for the velocity without improving the order of accuracy of the pressure. For time-dependent operators, the three-stage method derived in section 4.1.2 is to be preferred: it maintains third-order accuracy on time-varying meshes, and is second-order accurate for the pressure. Of course, the coefficients of a Runge-Kutta method can be chosen on other grounds than accuracy only, for example low storage, low dispersion or built-in error estimation with adaptive step-size control. Such arguments have not been considered in this work.

## References

- [1] P.R. Spalart, R.D. Moser, and M.M. Rogers. Spectral methods for the Navier-Stokes equations with one infinite and two periodic directions. *J. Comput. Phys.*, 96(2):297–324, 1991.
- [2] H. Le and P. Moin. An improvement of fractional step methods for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 92(2):369–379, 1991.
- [3] Y. Morinishi, T.S. Lund, O.V. Vasilyev, and P. Moin. Fully conservative higher order finite difference schemes for incompressible flows. *J. Comput. Phys.*, 143:90–124, 1998.
- [4] R. Knikker. Study of a staggered fourth-order compact scheme for unsteady incompressible viscous flow. *Int. J. Numer. Meth. Fluids*, 59:1063–1092, 2009.
- [5] N. Nikitin. Third-order-accurate semi-implicit Runge-Kutta scheme for incompressible Navier-Stokes equations. *Int. J. Numer. Meth. Fluids*, 51:221–233, 2006.
- [6] J.M.C. Pereira, M.H. Kobayashi, and J.C.F. Pereira. A fourth-order-accurate finite volume compact method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 167:217–243, 2001.
- [7] N.A. Kampanis and J.A. Ekaterinaris. A staggered, high-order accurate method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 215:589–613, 2006.
- [8] E. Hairer, C. Lubich, and M. Roche. *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods*. Springer, 1989.
- [9] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems*. Springer, 1996.
- [10] F.H. Harlow and J.E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluids*, 8:2182–2189, 1965.
- [11] U.M. Ascher and L.R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, 1998.
- [12] B. Sande and B. Koren. Accuracy analysis of explicit Runge-Kutta methods applied to the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 231:3041–3063, 2012.
- [13] P.M. Gresho and R.L. Sani. *Incompressible Flow and the Finite Element Method. Volume 2: Isothermal Laminar Flow*. Wiley, 2000.
- [14] J.B. Perot. An analysis of the fractional step method. *J. Comput. Phys.*, 180:51–58, 1993.
- [15] J.B. Perot. Comments on the fractional step method. *J. Comput. Phys.*, 121:190–191, 1995.
- [16] J.C. Butcher. *Numerical Methods for Ordinary Differential Equations*. Wiley, 2003.
- [17] V. Brasey and E. Hairer. Half-explicit Runge-Kutta methods for differential-algebraic systems of index 2. *SIAM J. Numer. Anal.*, 30(2):538–552, 1993.
- [18] J.C. Butcher. Coefficients for the study of Runge-Kutta integration processes. *J. Australian Math. Soc.*, 3:185–201, 1963.
- [19] F. Cameron. A Matlab package for automatically generating Runge-Kutta trees, order conditions, and truncation error coefficients. *ACM Trans. Math. Software*, 32(2):274–298, 2006.
- [20] C.-W. Shu. Essentially Non-Oscillatory and Weighted Essentially Non-Oscillatory schemes for hyperbolic conservation laws. Technical Report NASA/CR-97-206253, ICASE Report no. 97-65, NASA Langley Research Center, 1997.
- [21] J.C. Butcher. Implicit Runge-Kutta processes. *Math. Comput.*, 18:50–64, 1964.
- [22] B. Perot and R. Nallapati. A moving unstructured staggered mesh method for the simulation of incompressible free-surface flows. *J. Comput. Phys.*, 184:192–214, 2003.
- [23] P. Wesseling. *Principles of Computational Fluid Dynamics*. Springer, 2001.

- [24] B. Sanderse, S.P. van der Pijl, and B. Koren. Review of computational fluid dynamics for wind turbine wake aerodynamics. *Wind Energ.*, 14:799–819, 2011.
- [25] B. Sanderse. ECNS: Energy-Conserving Navier-Stokes Solver. Verification of steady laminar flows. Technical Report ECN-E-11-042, Energy research Centre of the Netherlands, 2011.