14th World Congress on Computational Mechanics (WCCM)
ECCOMAS Congress 2020
Virtual Congress: 11-15 January 2021
F. Chinesta, R. Abgrall, O. Allix and M. Kaliske (Eds)

VERIFICATION OF BDF TIME-INTEGRATION METHOD WITH VARIABLE STEP SIZE

CHRISTIAAN M. KLAIJ[1]

Maritime Research Institute Netherlands[1]
P.O. Box 28, 6700 AA Wageningen, The Netherlands
c.klaij@marin.nl, www.marin.nl

Key words: Time integration, Backward Differentiation Formulas, variable step size, incompressible flow, Navier-Stokes equations

Abstract. Solution methods for systems of ordinary differential equations with variable step size base their choice for the step size on (an estimate of) the local truncation error. Significant savings in computational effort can be obtained as the step size is only refined when necessary. We first summarize the information that is needed for a variable step size BDF2 method in a way that is directly suitable for implementation in an existing code. We then propose a solution verification procedure that does not require direct control of the step size to carry out refinement studies and apply it to a textbook example. Finally, we explore the use of the variable step size BDF2 method for time integration of the Navier-Stokes equations in the CFD package REFRESCO and find plausible results for vortex shedding in the wake of a cylinder, but no major savings for this particular case.

1 INTRODUCTION

With the ever increasing computational power, unsteady simulations are coming within reach of research and commercial CFD applications in the maritime industry. This trend motivates the study of time integration methods, in an effort to better understand the time step dependence of such simulations and to explore more efficient alternatives.

The time integration methods of choice in many CFD packages are the first-order backward Euler method and the second-order BDF2 method, due to their robustness and proven ability to solve the unsteady (Reynolds-averaged) Navier-Stokes equations, typically discretized with second-order finite-volume methods. Both methods are members of the family of n-step Backward Differentiation Formulas (BDFn), backward-Euler being equivalent to BDF1. This family of implicit time integration methods is favored for its ability to solve non-stiff and stiff problems, with constant and variable step size and with an accuracy ranging from first up to sixth order. These methods and their advantages compared to other methods are well-described in the textbooks by Hairer, Nørsett and Wanner [4, 5].

As a first step, the variable step size BDF2 method was selected for implementation in MARIN's CFD software package REFRESCO [7]. It is a logical extension of the current method from constant to variable step size. The step size selection is guided by an estimate of the local truncation error. We hope for significant gains in efficiency by adjusting the step size to the flow instead of having a constant (small) step size during the entire simulation. In Section 2 of this paper, we present a variable step size BDF2

method in a way that is directly suitable for implementation in REFRESCO or any other code.

The relation between the (estimated) local truncation error and the global error is not always trivial. So the fact that a certain tolerance is met for the error estimator does not exempt us from the next step: solution verification. Verification and Validation [8] studies provide confidence in the overall quality of the numerical modeling and solution. The intricacies of solution verification procedures for unsteady flows have recently been discussed in [3]. In essence, these procedures rely on systematic mesh and/or time-step refinement studies, where the observed order of convergence is compared to the theoretical order. This raises the question of how to perform a refinement study when the step size is chosen automatically, instead of being controlled by the user. We will attempt to answer this question in Section 3 and show solution verification results for the textbook demonstration case known as the Brusselator, consisting of two coupled nonlinear ordinary differential equations.

As final step, we explore the applicability of the variable step size BDF2 method to time-integration of the incompressible Navier-Stokes equations in Section 4. After discussing some of the challenges encountered during the implementation, we show some preliminary results for the archetypal unsteady flow in the wake behind a circular cylinder before drawing conclusions in Section 5.

2 VARIABLE STEP SIZE BDF2 METHOD

Solution methods for (systems of) ordinary differential equations are well-described in the textbooks by Hairer, Nørsett and Wanner [4, 5], including the Backward Differentiation Formulas (BDF) with variable step size that are of interest here. The information, however, is scattered over hundreds of pages, with methods written in general form for arbitrary order of accuracy, using sums and products with recursively defined terms. Besides, the step size selection procedure is mainly described for Adams methods, not for BDF methods. Therefore, in this section, we gather the information that is needed for a variable step size BDF2 method and present it in a way that is directly suitable for implementation in REFRESCO or any other code.

2.1 BDF methods with constant step size

Consider the ordinary differential equation $y'(x) = f(y(x))$ where $(\cdot)'$ denotes differentiation with respect to x, with f a given function and $y(x)$ the unknown solution, with given initial value $y(x_0) = y_0$. This is the notation from the textbook [4, Sec.I.1, p.2] that we will follow in the remainder. The six stable BDF methods with constant step size are listed in [4, Sec.III.1, Eq.(1.22"), p.365]. The first three BDF methods are

$$\begin{aligned} y_{n+1} - y_n &= h f_{n+1} \\ \frac{3}{2} y_{n+1} - 2 y_n + \frac{1}{2} y_{n-1} &= h f_{n+1} \\ \frac{11}{6} y_{n+1} - 3 y_n + \frac{3}{2} y_{n-1} - \frac{1}{3} y_{n-2} &= h f_{n+1} \end{aligned}$$

where y_n is the numerical approximation to the exact solution $y(x_n)$, $f_n = f(y_n)$ and h is the constant step size of the uniform mesh $x_i = x_0 + ih$ with $i = 0, 1, \ldots$ These implicit non-linear equations are solved (iteratively) for y_{n+1}. The methods are respectively first-, second- and third-order accurate.

2.2 Error estimator

The local truncation error (LTE) for step n is defined as $y(x_n) - y_n$. By approximating the exact solution $y(x_n)$ with the higher-order BDF3 method, we get an error estimator $y_{\mathrm{BDF3}} - y_{\mathrm{BDF2}}$ for the lower-order BDF2 method:

$$y(x_{n+1}) - y_{n+1} \approx \frac{1}{3}y_{n+1} - y_n + y_{n-1} - \frac{1}{3}y_{n-2} \tag{1}$$

Notice that this error estimator can be directly calculated once y_{n+1} has been obtained with BDF2 and y_n, y_{n-1} and y_{n-2} are kept in memory. It does not require an additional iterative solution with BDF3, but merely the additional storage of y_{n-2}. The idea of approximating the LTE as the difference of a higher- and lower-order method, assuming a constant step size, is described in [4, Sec.III.7, p.421] for Adams methods.

2.3 BDF2 method with variable step size

The second BDF method with variable step size is given in [4, Sec.III.5, Eq.(5.14), p.401] and can be written as:

$$\frac{1+2w_n}{1+w_n}y_{n+1} - \frac{(1+w_n)^2}{1+w_n}y_n + \frac{w_n^2}{1+w_n}y_{n-1} = h_n f_{n+1} \tag{2}$$

with $w_n = h_n/h_{n-1}$ the step size ratio and $h_n = x_{n+1} - x_n$. The other BDF methods are not explicitly listed, but given in general form using (recursive) divided differences in [4, Sec.III.5, Eq.(5.12), p.400].

2.4 Stability

The BDF methods with variable step size are stable under certain restrictions [4, Sec.III.5, Th.5.5, p.402] of the variation:

$$\omega \leq h_n/h_{n-1} \leq \Omega$$

The actual values for the lower bound ω and upper bound Ω are given in [4, Sec.III.5, Tab.5.1, p.405]. For the BDF2 method these values are quite mild:

$$0 \leq h_n/h_{n-1} \leq 1+\sqrt{2} \approx 2.414$$

This states that the current step can be arbitrarily smaller than the previous one, but at most 2.4 times larger. For third- and higher-order BDF methods these values are much more restrictive.

2.5 Convergence

The global error can be found by considering the local error at each step, propagating these errors and summing them at the final time. Under certain conditions expressed in [4, Sec.III.5, Th.5.8, p.407], the global error satisfies

$$\|y(x_n) - y_n\| \leq Ch^p \tag{3}$$

with $h = \max h_i$ the maximum step size during the computation and p the theoretical order of grid convergence, with $p = 2$ for the variable step size BDF2 method.

2.6 Step size selection

A procedure for automatic step size selection is given in [4, Sec.II.4, p.167] for Runge-Kutta methods and in [4, Sec.III.7, p.421] for Adams methods. The main idea is that a starting size is chosen and the solution and its error estimator are computed. The step is accepted if the error is below the desired tolerance and the size can be increased for the next step. Otherwise, the step is rejected and the size is decreased for a new try. The change in step size depends on the difference between the error and the desired tolerance. For a step of size h, the relative error is defined as

$$\text{err} = \frac{|y(x_n) - y_n|}{\text{sc}} \qquad \text{with} \qquad \text{sc} = A_{\text{tol}} + \max\{|y_n|, |y_{n-1}|\} R_{\text{tol}} \tag{4}$$

and the new step size is obtained by

$$h_{\text{new}} = h \min\left\{F_{\max}, \max\left\{F_{\min}, (1/\text{err})^{\frac{1}{p+1}} F\right\}\right\} \tag{5}$$

with factors $F = 0.8$, $F_{\min} = 0$ and $F_{\max} = 2.414$, if the previous step was accepted, or $F_{\max} = 1$ if it was rejected. Notice that the order p leads to the power $(\cdot)^{1/3}$ for the BDF2 method. If y is a vector, the error can be evaluated in the l_2 or l_∞ norm.

2.7 Dense output

Remember that the error estimator $y(x_n) - y_n$ assumes a uniform grid with step size h. This requires dense output: the accurate interpolation of intermediate values from the given non-uniform distribution. Dense output using Hermite interpolation is discussed in detail for Runge-Kutta and other methods [4, Sec.II.6], but not for BDF methods. The reason is probably that BDF methods offer dense output by definition. The BDF3 method on which the error estimator is based by definition constructs a polynomial $q(t)$ through the points (x_{n+1}, y_{n+1}), (x_n, y_n), (x_{n-1}, y_{n-1}), (x_{n-2}, y_{n-2}) such that $q'(x_{n+1}) = f(y_{n+1})$, see [4, Sec.III.5, Eq.(5.11), p.400]. Therefore, the solution at another point t in the interval can simply be obtained by evaluating $q(t)$. The polynomial $q(t)$ for the BDF3 method is given as a sum over j of divided differences δ^j:

$$\begin{aligned}
j = 0: &\quad \delta^0_{n+1} \\
j = 1: &\quad + (t - x_{n+1})\delta^1_{n+1} \\
j = 2: &\quad + (t - x_{n+1})(t - x_n)\delta^2_{n+1} \\
j = 3: &\quad + (t - x_{n+1})(t - x_n)(t - x_{n-1})\delta^3_{n+1} = q(t)
\end{aligned} \tag{6}$$

The divided differences are recursively defined as:

$$\begin{aligned}
j = 3: &\quad \delta^3_{n+1} = \frac{\delta^2_{n+1} - \delta^2_n}{x_{n+1} - x_{n-2}} \\
j = 2: &\quad \delta^2_{n+1} = \frac{\delta^1_{n+1} - \delta^1_n}{x_{n+1} - x_{n-1}} \qquad \delta^2_n = \frac{\delta^1_n - \delta^1_{n-1}}{x_n - x_{n-2}} \\
j = 1: &\quad \delta^1_{n+1} = \frac{\delta^0_{n+1} - \delta^0_n}{x_{n+1} - x_n} \qquad \delta^1_n = \frac{\delta^0_n - \delta^0_{n-1}}{x_n - x_{n-1}} \qquad \delta^1_{n-1} = \frac{\delta^0_{n-1} - \delta^0_{n-2}}{x_{n-1} - x_{n-2}} \\
j = 0: &\quad \delta^0_{n+1} = y_{n+1} \qquad \delta^0_n = y_n \qquad \delta^0_{n-1} = y_{n-1} \qquad \delta^0_{n-2} = y_{n-2}
\end{aligned}$$

2.8 Algorithm

The algorithm for integration with automatic step size selection can be summarized as follows:

1. Given y_n, y_{n-1} and step size h_n, compute y_{n+1} by solving Eq. (2).
2. Calculate the values $y^*_{n-1} = q(x_n - h_n)$ and $y^*_{n-2} = q(x_n - 2h_n)$ using Eq. (6).
3. Calculate the scaled error from Eq. (4) using Eq. (1) with y_{n+1}, y_n, y^*_{n-1} and y^*_{n-2}.
4. Calculate the new step size from Eq. (5).
   (a) if err ≤ 1, accept the current step and take the next step with size h_{new}
   (b) if err > 1, reject the current step and retake with size h_{new}

2.9 Error estimator with variable step size?

In [2], the LTE is approximated as the difference between the BDF2 and BDF3 method with variable step size, which seems logical at first sight. We tried this approach but found that the refinement process fails when (very) small values are chosen for the relative tolerance. After a number of accepted steps, the new step does not satisfy the tolerance, and is decreased. But even a decrease to zero fails! This behavior was not due to an implementation issue, nor to a faulty expression of the LTE, but rather to a reasoning mistake: the LTE estimator does *not* tend to zero if the last step h_n is decreased, *while* h_{n-1} *and* h_{n-2} *remain fixed*. That is why the procedure proposed in [4] interpolates the previous values y_{n-1} and y_{n-2} to a uniform grid with step size h_n before computing the error estimate. It ensures that the error estimator tends to zero as h_n tends to zero. Therefore, any tolerance can be satisfied by reducing the step size sufficiently.

3 VERIFICATION METHOD FOR VARIABLE STEP SIZE

3.1 Brusselator example

To gain some experience with this method, we first tried the so-called Brusselator example [4, Sec.I.16, Eq.(16.12), p.116]. Here, y is a vector with two components and the non-linear function $f(y)$ is given by

$$y_1' = A + y_1^2 y_2 - (B+1)y_1$$
$$y_2' = By_1 - y_1^2 y_2$$

with parameters $A = 1$, $B = 3$ and initial position $(1.5, 3.0)$. The solution is periodic and describes an oval trajectory in the (y_1, y_2)-plane, see Figure 1. A Newton solver is used for the non-linear system. Figure 2 shows the solution $y_1(x), y_2(x)$ for the first three cycles. A relative tolerance of 10^{-3} was prescribed, meaning that the estimated error is less than 0.1% of the solution at each step. A total of 179 steps was needed, with 15 rejected steps. These rejections typically occur around the sharp bends in the trajectory where the step size must be refined to meet the tolerance. The step size varies between 0.01 and 1 which suggests huge savings compared to a constant step size method with step size fixed to the minimum of 0.01. Table 1 shows how the number of steps increases as the relative tolerance is tightened. Notice that (very) small tolerances are achieved without a dramatic increase in the number of rejected steps. Exact

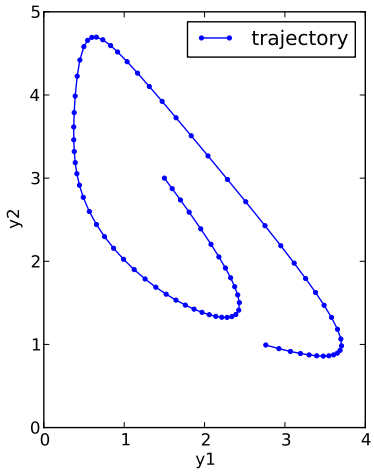

Figure 1: Trajectory of the Brusselator example from starting point (1.5,3.0) to position at time $x = 7.8$ using the automatic variable step size BDF2 method with $R_{\text{tol}} = 10^{-3}$.

Table 1: Number of steps for the Brusselator example for various tolerances.

R_{tol}	steps	accepted	rejected
10^{-1}	34	27	7
10^{-2}	100	86	14
10^{-3}	179	164	15
10^{-5}	745	740	5
10^{-8}	7381	7377	4

comparison with [4] is not possible because the Adams methods are used there instead and both step size and order of the method are adjusted. Nevertheless, the results are in overall agreement: similar step sizes and similar refinement when the solution varies strongly.

3.2 Solution verification method: constant versus variable step size

As convincing as the demonstration above may be, it does not consider the quality of the solution, which is addressed here using solution verification. Although 'exact' solutions exist for the Brusselator example in the form of infinite series [1], we will assume the more general situation where an exact solution is not available and continue with solution verification based on refinement studies. Such studies do not require access to the source code and can therefore be carried out by any user for any case. The main question to be answered here is how to perform a refinement study when the step size is chosen automatically instead of being chosen by the user.

As quantity of interest, we choose the norm of the position vector (y_1, y_2) at 'time' $x = 7.8$, which covers the start-up and the three sharp bends of the trajectory. The observed convergence rate can be computed as

$$\text{rate} = \frac{|\phi_{4h} - \phi_{2h}|}{|\phi_{2h} - \phi_h|}$$

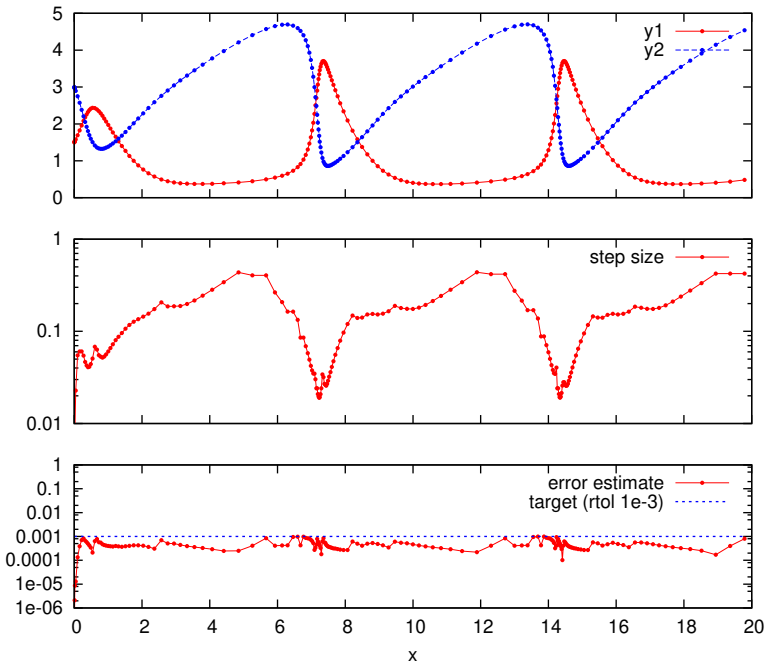

Figure 2: Three cycles of the Brusselator example using the variable step size BDF2 method with $R_{\text{tol}} = 10^{-3}$.

with $\phi_h = \|(y_1, y_2)\|$ the norm of the position vector at $x = 7.8$. For the constant step size BDF2 method, the subscript h denotes the step size. Since the global error scales with Ch^p in the asymptotic regime and the order p is equal to two for the BDF2 method, the observed rate should be around four if the method is correctly implemented. This is indeed the case as seen in Table 2. Notice that we do not compute the order p itself because doing so would assume equal step size ratio's, see [3, Eq. (7)], which is trivial for constant but not for variable step size.

For a variable step size, the user controls the tolerance R_{tol}, not the step size itself. Therefore, the first (naïve) approach would be to interpret the subscript h as the prescribed relative tolerance, and successively halve it to carry out the refinement study. The surprising result of this approach is shown in Table 3: the observed convergence rate has dropped well below two! However, this does not indicate an implementation issue. Remember from Eq. (3) that the global error is supposed to scale with Ch^p with $h = \max h_i$ and $p = 2$ for variable step size BDF2. But in Table 3, we notice that $\max h_i$ is not halved when the relative tolerance is halved. Instead, as shown by Eq. (4) and Eq. (5), the step size h scales with $(1/\text{err})^{1/3}$ and $(1/\text{err})$ scales with R_{tol}. Thus, halving R_{tol} only reduces the step size h by a factor $2^{1/3} \approx 1.25$. As a result, the observed convergence rate becomes $2^{2/3} \approx 1.58$ instead.

This relation between the relative tolerance and the (maximum) steps size directly points to a more savvy approach: reduce the relative tolerance by a factor eight instead of two. This should indirectly reduce the step size by a factor $8^{1/3} = 2$ and show the expected convergence rate of four if the method is correctly implemented. This is confirmed by the results in Table 4. Notice that the first two steps in the simulations from Tables 3 and 4 are taken with the constant step size h of the corresponding row in Table 2. This explains the small differences in results for $R_{\text{tol}} = 2^{-15}$ and 2^{-18}. More elaborate starting procedures can

Table 2: Standard convergence study of position at $x = 7.8$ with constant step size BDF2.

h	steps	$\|(y_1, y_2)\|$	rate
2^{-4}	125	**2.9**3359910	–
2^{-5}	250	**2.94**047485	–
2^{-6}	500	**2.943**01065	2.71
2^{-7}	1000	**2.943**73770	3.49
2^{-8}	2000	**2.9439**3036	3.77
2^{-9}	4000	**2.9439**7984	3.89
2^{-10}	8000	**2.94399**237	3.95
2^{-11}	16000	**2.94399**553	3.97
2^{-12}	32000	**2.94399**632	3.99

Table 3: Naïve convergence study of position at $x = 7.8$ with variable step size BDF2 (halving R_{tol}).

R_{tol}	$\max h_i$	steps	$\|(y_1, y_2)\|$	rate
2^{-12}	0.34	123	**2**.89705272	–
2^{-13}	0.26	157	**2.9**2275801	–
2^{-14}	0.27	199	**2.9**3345705	2.40
2^{-15}	0.18	252	**2.9**3839611	2.17
2^{-16}	0.14	317	**2.94**087052	2.00
2^{-17}	0.11	398	**2.94**218145	1.89
2^{-18}	0.093	501	**2.94**291838	1.78
2^{-19}	0.075	630	**2.943**4639	1.72
2^{-20}	0.059	792	**2.943**60079	1.68

be found in[4, Sec II.4,p.169].

Now that the calculations with both constant and variable step size have been verified, we face the fact that the performance of the variable step size BDF2 method is disappointing. For instance, imagine that four significant digits would be desired for the position at $x = 7.8$. With the constant step size method, we can see in Table 2 that we would need $h = 2^{-7} \approx 0.0078$ and 1000 steps to achieve this level of precision. With the variable step size method, Table 4 shows that we would need $R_{\text{tol}} = 2^{-21} \approx 4.7 \cdot 10^{-7}$ and 995 steps, despite the fact that the maximum steps size of 0.046 is significantly higher than the constant step size of 0.0078. Clearly, there must also be a large number of significantly lower step sizes involved. At this point, it is unclear whether this is the expected behavior for this particular case. It points to a wider shortcoming of the verification process: by refinement studies we can verify stability (the method converges) and consistency (it converges with the expected rate) but not actual performance.

4 APPLICATION TO NAVIER-STOKES SOLVER

Time integration with automatic step size selection is standard in the solution of ordinary differential equations, but it is certainly not standard in CFD packages. Besides fundamental reasons, if any, this may be explained by practical challenges that we discuss first.

Table 4: Savvy convergence study of position at $x = 7.8$ with variable step size BDF2 (indirectly halving $\max h_i$).

R_{tol}	$\max h_i$	steps	$\|(y_1, y_2)\|$	rate
2^{-12}	0.34	123	**2**.89705272	–
2^{-15}	0.18	250	**2.9**3533722	–
2^{-18}	0.093	499	**2.94**210815	5.65
2^{-21}	0.046	995	**2.943**5214	4.69
2^{-24}	0.023	1988	**2.943**8859	4.29
2^{-27}	0.011	3972	**2.9439**6996	4.13
2^{-30}	0.0059	7940	**2.9439**8997	4.06
2^{-33}	0.0029	15875	**2.94399**494	4.03
2^{-36}	0.0014	31743	**2.94399**617	4.02

4.1 Challenges

The target CFD package REFRESCO solves the incompressible (Reynolds-averaged) Navier-Stokes equations, discretized with a finite volume method on unstructured meshes of arbitrary polyhedral cells with all variables co-located in the cell centers. The equations are linearized with Picard's method and solved iteratively with pressure-correction type methods [6]. All governing equations except the pressure correction equation are cast in the generic form

$$\int_V \frac{\partial \phi}{\partial t} dV + \int_S \phi \mathbf{v} \cdot \mathbf{n} dS - \int_S \mu \nabla \phi \cdot \mathbf{n} dS = \int_V s dV$$

with ϕ the generic field, $\mathbf{v}$ the velocity, μ the (effective) viscosity, s a source term and V a control volume with surface S. This equation can be recast in the form $\phi' = f(\phi)$ where the $(\cdot)'$ denotes differentiation with respect to t. Replacing t by x and ϕ by y, we arrive at the textbook notation $y'(x) = f(y(x))$ for ODE's that was introduced in Section 2.

The application of the variable step size BDF2 method requires *(a)* additional storage of the solution at level $n-2$ for the error estimator and *(b)* the ability to reject a time step and return to the previous state to retake the step with a modified size. Point *(a)* mainly raises concerns about memory usage and was tackled by defining a master variable, tentatively set to the momentum magnitude $\|\rho\mathbf{v}\|$, for which an additional level $n-2$ is stored. This only adds a single scalar field to the storage and covers the three momentum equations at once. Since the mass equation does not have a time derivative, it was not considered. The error is solely estimated for this master variable to control the step size selection. Point *(b)* is more challenging, as it requires the storage of the exact state at the beginning of each time step and precludes the overwriting of quantities such as fluxes and gradients, during the iterative process between time steps. In case of mesh motion, deformation and adaptive refinement (not considered here), this also concerns the overwriting of geometrical quantities that change at the beginning of a new time step. Thus, it may be tempting to apply the method without rejection first. However, we found that this leads to very irregular patterns where a large step, that should have been rejected, is typically followed by a series of steps that are rather small.

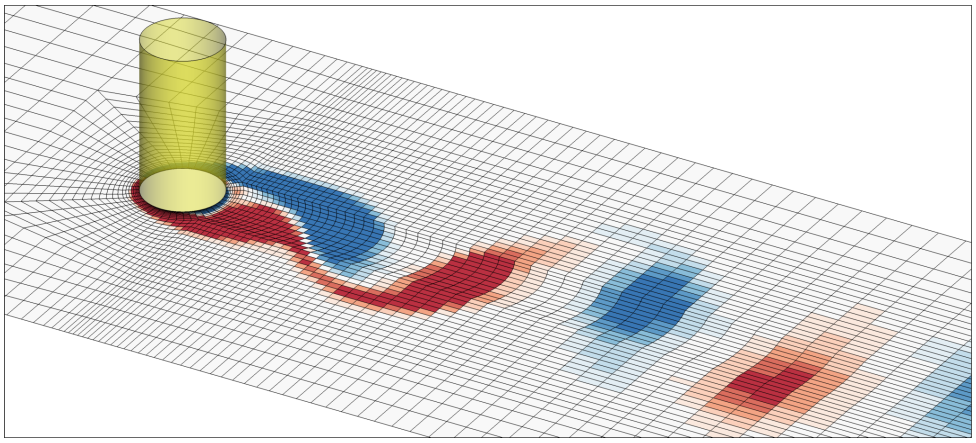

Figure 3: Impression of mesh and vortex shedding behind circular cylinder at $\mathrm{Re} = 200$.

Table 5: Total number of time steps and outer loops for the cylinder case during $t\,U_{\mathrm{ref}}/D = 10$.

R_{tol}	steps	outer loops	step size ($\Delta t\,U_{\mathrm{ref}}/D$)	
			min	max
n.a.	200	20749	constant 0.05	
10^{-3}	64 accpt + 0 rejct	19078	0.12	0.19
$5 \cdot 10^{-4}$	94 accpt + 2 rejct	19534	0.077	0.14
$2.5 \cdot 10^{-4}$	171 accpt + 22 rejct	20795	0.037	0.10

4.2 Vortex shedding example

Vortex shedding flow around a circular cylinder was selected as the archetypal unsteady case that is well documented and easy to replicate for any code. The Reynolds number is 200 and a very coarse block-structured mesh is used as shown in Figure 3. When the target tolerance is set to 10^{-3}, we see in Figure 4 that the time-step size varies between $\Delta t U_{\mathrm{ref}}/D = 0.01$ and 0.02 and repeats its variation together with the shedding cycles. There seems to be some relation between the step size and the lift and drag coefficients, with smaller step sizes around their minimum and maximum. Compared to Figure 2 of the Brusselator example, the signals are much smoother and hence do not require significant variation in step size.

Table 5 shows the effect of the relative tolerance: for lower tolerances, there seems to be a larger variation in step size and an increase in the number of rejected steps. The performance is measured in terms of the number of outer loops. At each time step, an iterative convergence tolerance of 10^{-6} must be satisfied: for the constant step size of 0.05 this typically requires ten outer loops, but this number increases as the step size increases. For fair comparison, we also count the outer loops carried out during the rejected time steps. As a result, the variable step size BDF2 method is not significantly cheaper because the saving in number of steps is almost entirely negated by the increase in outer loops per time step. The (almost linear) increase of the number of outer loops with the step size is probably related to the Picard linearization of the momentum equations.

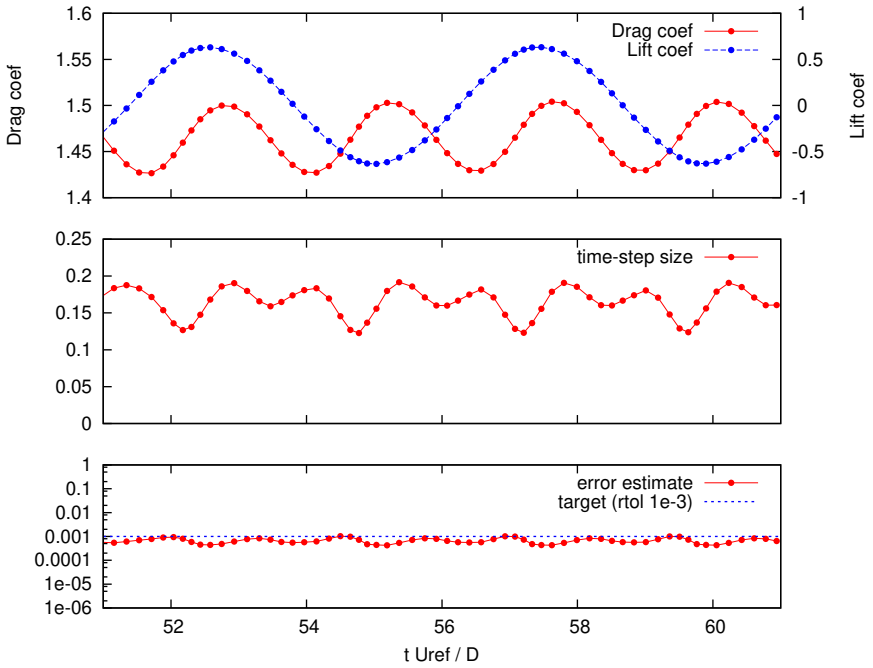

Figure 4: Cylinder lift and drag coefficients using the automatic variable step size BDF2 method with $R_{\text{tol}} = 10^{-3}$.

4.3 Solution verification?

Solution verification of periodic unsteady flow is a delicate matter as pointed out in [3] for a one-dimensional manufactured solution. The initial condition already exerts an influence, iterative errors may propagate to the next time step and statistical errors may occur. All of these errors must be quantified and controlled before the discretization error can be measured. An additional complication here, is that the period is now part of the solution instead of being known from the manufactured solution. Although these difficulties can be overcome, for instance by using some signal analysis software, it defeats the purpose of presenting an easily replicable benchmark case. Added to the previous observation that the flow does not require significant step size variation, we can only conclude that this case is not ideal for evaluating adaptive time-integration methods.

5 CONCLUSIONS

In this paper, we gathered from Hairer, Nørsett and Wanner's textbooks all the information needed for the implementation of a second-order Backward Differentiation Formula (BDF2) with automatic step size variation, guided by an estimate of the local truncation error. Along the way, we clarified some points that can easily be misunderstood, such as why a BDF3 solve is not necessary despite the error estimate being based on the difference between second- and third-order solutions, and why dense output is used to evaluate the error estimate with equidistant spacing.

This variable step size BDF2 method was then applied to the textbook example known as the Brusselator to verify that the step size is refined and coarsened in overall agreement with the textbook results. How-

ever, the fact that a certain tolerance is met for the (estimated) local truncation error does not exempt us from solution verification, which concerns the global error. Since the usual approach of successively halving the (constant) step size to measure the convergence rate cannot be applied to variable step size methods, we proposed a solution verification method that indirectly halves the maximum step size by reducing the tolerance with a specific factor. The method was called 'savvy' as opposed to the 'naïve' idea of halving the target tolerance because it requires knowledge of the step size selection procedure to determine the appropriate factor. This information should not be hidden from end-users if we expect them to carry out their own solution verification studies.

Finally, we explored the use of the variable step size BDF2 method for time integration of the Navier-Stokes equations in the CFD package REFRESCO by simulating laminar flow around a two-dimensional circular cylinder. Although there are some practical issues, such as the ability to reject time steps that fail to meet the target tolerance, we did find the method to be feasible and obtained plausible results for the variation in step size. This example probably did not show significant savings in computational time because the flow is quite regular, which does not leave much room for step size variation. Also, solution verification is not straightforward for this case. Further work should aim at defining a benchmark case that does highlight the benefits of variable step size, yet is easy to replicate with various codes and more amenable to solution verification.

REFERENCES

[1] G. Adomian. The diffusion-Brusselator equation. *Computers & mathematics with applications*, 29(5):1–3, 1995.

[2] E. Alberdi Celaya, J. J. Anza Aquirrezabala, and P. Chatzipantelidis. Implementation of an adaptive BDF2 formula and comparison with the MATLAB Ode15s. *Procedia Computer Science*, 29:1014–1026, 2014.

[3] L. Eça, G. Vaz, S. L. Toxopeus, and M. Hoekstra. Numerical errors in unsteady flow simulations. *Journal of Verification, Validation and Uncertainty Quantification*, 4, 2019.

[4] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving ordinary differential equations I. Nonstiff problems*. Springer-Verlag Berlin Heidelberg, 1993.

[5] E. Hairer and G. Wanner. *Solving ordinary differential equations II. Stiff and differential-algebraic problems*. Springer-Verlag Berlin Heidelberg, 1996.

[6] C. M. Klaij and C. Vuik. SIMPLE-type preconditioners for cell-centered, colocated finite volume discretization of incompressible Reynolds-averaged Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 71(7):830–849, 2013.

[7] Maritime Research Institute Netherlands. ReFRESCO Web page. `http://www.refresco.org`, 2020. Accessed: July 9, 2020.

[8] P. J. Roache. *Verification and Validation in Computational Science and Engineering*. Hermosa Publishers, Albuquerque, New Mexico, 1998.