TIME STEP ADAPTIVITY IN THE METHOD OF DAHLQUIST, LINIGER AND NEVANLINNA

WILLIAM LAYTON*, WENLONG PEI†, AND CATALIN TRENCHEA‡

Abstract. Dahlquist, Liniger and Nevanlinna devised a family of one-leg two-step methods (DLN) that is second order, A- and G- stable for arbitrary, non-uniform time steps. The DLN method thus has strong potential for use in adaptive codes, but its adaptive step size selection is little explored. This report develops two approaches for the efficient local error estimation in the DLN method, and tests their use in a standard adaptivity framework. Many methods of error estimation are possible; herein we focus on two complementary estimators which involve minimal extra storage and computations. First we evaluate the local truncation error of the DLN method by Milne's device, using the difference between the solution of the DLN method and the solution of a variable-step, explicit, second-order Adams-Bashforth-like method. Second, we use a recent refactorization of the DLN method, which eases implementation of DLN in legacy codes, to obtain an effective error estimation at no extra cost. We perform a number of numerical tests, comparing the two time adaptive DLN algorithms with some standard numerical ODE packages. Our tests indicate that the adaptive DLN method, with error estimated by Milne's device, is an efficient and reliable method, even for stiff and unstable problems.

Key words. *G*-stability, second-order, time adaptivity, long-time simulations

AMS subject classifications. 65L05, 65L20, 65L80

1. Introduction.

Time step adaptivity (adjusting the time step according to a required local/global error tolerance) is an essential algorithmic feature in numerical simulations, since it the improves the long time accuracy, and keeps low the computational cost. To our knowledge, the variable time-stepping DLN method is the **only** two-step scheme which is nonlinearly (G-) stable and second order accurate. Hence it is the ideal candidate for time-step adaptivity for very large systems, where large time step ratios would reduce computational and space complexity. The adaptivity process, based on a predictor-corrector approach [72], involves two issues: the (local and/or global) error estimator and the time step controller.

We estimate the local truncation error (LTE) of DLN, using the differentiation defect [9, 58], in two different ways.

First we approximate it by using Milne's device [65] via a second order, two-step, explicit, variable step Adams-Bashforth 2-like method [9, 11, 32, 72]. The second method uses the difference between the first and second order approximations, embedded in the refactorized DLN method [9, 58], to estimate the error and adapt the time step. We stress that this second approach, although it gives a pessimistic estimator of the local errors, requires no extra work or storage, and is intended for applications with severe memory limitations precluding the first.

For the time step controller, we employ the classical controller in [46], which guarantees the zero–stability of general one–leg variable step size methods (see e.g., [19, 33, 45, 52, 55]), and enhances algorithmic robustness. Improved controllers have been developed in [75] for smooth non-uniform grids. We use the standard controller to focus on stiff problems on the DLN method and its estimators.

* Department of Mathematics, University of Pittsburgh, Pittsburgh, PA 15260, USA. Email: wjl@pitt.edu. Partially supported by NSF grant DMS 2110379.

† Department of Mathematics, Ohio State University, Columbus, OH 43210, USA. Email: pei.176@osu.edu. Partially supported by NSF grant DMS 2110379.

‡ Department of Mathematics, University of Pittsburgh, Pittsburgh, PA 15260, USA. Email: trenchea@pitt.edu. Partially supported by NSF grant DMS-2208220.

1.1. The DLN Method. To begin, consider the initial value problem

$$y'(t) = f(t, y(t)), \quad y(0) = y_0, \tag{1.1}$$

where $y : [0,\infty) \to \mathbb{R}^d$ and $f : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$, and $y_0 \in \mathbb{R}^d$ is a given initial condition. Let $\{t_n\}_{n=0}^N$ be the grid on the time interval $[0,T]$ and $k_n = t_n - t_{n-1}$ be the local time step. The variable step, one parameter DLN family (with $\delta \in [0,1]$) reads

$$\begin{aligned} &\frac{\alpha_2 y_{n+1} + \alpha_1 y_n + \alpha_0 y_{n-1}}{\widehat{k}_n} \\ &= f\Big(\beta_2^{(n)} t_{n+1} + \beta_1^{(n)} t_n + \beta_0^{(n)} t_{n-1}, \beta_2^{(n)} y_{n+1} + \beta_1^{(n)} y_n + \beta_0^{(n)} y_{n-1}\Big). \end{aligned} \tag{DLN}$$

The coefficients α_ℓ and β_ℓ are

$$\begin{bmatrix} \alpha_2 \\ \alpha_1 \\ \alpha_0 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(\delta+1) \\ -\delta \\ \frac{1}{2}(\delta-1) \end{bmatrix}, \quad \begin{bmatrix} \beta_2^{(n)} \\ \beta_1^{(n)} \\ \beta_0^{(n)} \end{bmatrix} = \begin{bmatrix} \frac{1}{4}\Big(1 + \frac{1-\delta^2}{(1+\varepsilon_n\delta)^2} + \varepsilon_n^2 \frac{\delta(1-\delta^2)}{(1+\varepsilon_n\delta)^2} + \delta\Big) \\ \frac{1}{2}\Big(1 - \frac{1-\delta^2}{(1+\varepsilon_n\delta)^2}\Big) \\ \frac{1}{4}\Big(1 + \frac{1-\delta^2}{(1+\varepsilon_n\delta)^2} - \varepsilon_n^2 \frac{\delta(1-\delta^2)}{(1+\varepsilon_n\delta)^2} - \delta\Big) \end{bmatrix}. \tag{1.2}$$

The step size variability ε_n and average step $\widehat{k}_n$ are

$$\varepsilon_n = (k_n - k_{n-1})/(k_n + k_{n-1}) \in (-1,1), \qquad \widehat{k}_n = \alpha_2 k_n - \alpha_0 k_{n-1}. \tag{1.3}$$

The coefficients α_ℓ are functions of δ, while $\beta_\ell^{(n)}$ are functions of δ, ε_n, and are constructed to ensure second order accuracy. When $\delta = 1$ and $\delta = 0$, the DLN method reduces to the one-step midpoint rule, and the two-step midpoint rule, respectively [58].

We now recall that the (DLN) method can be refactorized as a backward Euler solve, complemented by a pre- and a post-process step, see e.g. [58] and Figure 3.1. Given previous two step solutions y_{n-1}, y_n and time grid points t_{n-1}, t_n, t_{n+1},

$$\begin{aligned} &\textbf{Step 1 (Pre-process):} && (\Delta t)_n^{\mathrm{BE}} = b^{(n)} \widehat{k}_n, \\ &&& t_{n+1}^{\mathrm{new}} = \beta_2^{(n)} t_{n+1} + \beta_1^{(n)} t_n + \beta_0^{(n)} t_{n-1}, \\ &&& y_n^{\mathrm{old}} = a_1^{(n)} y_n + a_0^{(n)} y_{n-1}, \\ &\textbf{Step 2 (BE solver):} && \frac{y_{n+1}^{\mathrm{new}} - y_n^{\mathrm{old}}}{(\Delta t)_n^{\mathrm{BE}}} = f\big(t_{n+1}^{\mathrm{new}}, y_{n+1}^{\mathrm{new}}\big), \\ &&& (\text{on interval } [t_{n+1}^{\mathrm{new}} - (\Delta t)_n^{\mathrm{BE}}, t_{n+1}^{\mathrm{new}}]) \\ &\textbf{Step 3 (Post-process):} && y_{n+1}^{\mathrm{DLN}} = c_2^{(n)} y_{n+1}^{\mathrm{new}} + c_1^{(n)} y_n + c_0^{(n)} y_{n-1}, \\ &&& (\text{on interval } t_{n+1}^{\mathrm{new}}, t_{n+1}). \end{aligned} \tag{refactorized DLN}$$

The coefficients in the DLN refactorization process are

$$\begin{aligned} &a_1^{(n)} = \beta_1^{(n)} - \alpha_1 \beta_2^{(n)}/\alpha_2, \quad a_0^{(n)} = \beta_0^{(n)} - \alpha_0 \beta_2^{(n)}/\alpha_2, \quad b^{(n)} = \beta_2^{(n)}/\alpha_2, \\ &c_2^{(n)} = 1/\beta_2^{(n)}, \quad c_1^{(n)} = -\beta_1^{(n)}/\beta_2^{(n)}, \quad c_0^{(n)} = -\beta_0^{(n)}/\beta_2^{(n)}. \end{aligned}$$

The paper is organized as follows. In Section 2, we evaluate the error estimator of the DLN method by several explicit second-order schemes. The adaptivity of DLN based on the first- order solution, provided by the refactorization, is introduced in Section 3. The numerical tests for the adaptive variable step DLN algorithms are presented in Section 4.

1.2. Related Work. Many numerical methods have been developed to approximate the solution of differential equations, e.g., [1, 11, 32, 39, 44–47]. In complex applications it is however still common to use simple methods, such as the constant time step backward Euler [16, 71], the midpoint rule [4, 7, 18, 42, 48, 68], the trapezoid rule [34, 49] or, increasingly, the second-order backward difference (BDF2) method [5, 17, 28, 29, 59, 78, 79, 83]. It is well known [67] that for linear multistep methods, unfavorable combinations of variable steps can lead to instability. For example, the variable-step trapezoid rule loses stability for some specific preset steps [25, 77], and BDF2 is unstable if the step size ratio is relatively large [12, 13, 30, 31, 40, 41]. Watanabe and Sheikh [82] point out other instabilities that can occur, even with adaptive algorithms and one-leg $k-$steps methods.

For non-uniform time grids, most linear multistep methods are unstable, at least for particular choices of time steps. In 1983, Dahlquist, Liniger and Nevanlinna [25] designed a *one parameter family* of one-leg, two step schemes (the (DLN) method) which is *G*-stable (nonlinearly stable) [20–22] and second order accurate, for *variable steps* and *arbitrary step-size ratios*. Its ability to increase step sizes rapidly, when local solution behavior warrants, is an attractive feature for large (storage limited) systems. The simple tests of the variable step DLN method in [6, 8, 52, 55] have confirmed its potential. The work herein is complemented by further DLN analysis applied to fluids problems in [57, 70], and refactorization to reduce the cognitive complexity of implementation in [42, 58].

The fact that the whole DLN family is (nonlinearly) G-stable and (linearly) A-stable is of paramount importance for accurate long-time simulations, and provides optimal computational options for either dissipative or conservatives physical systems.

One main remaining question, addressed herein, is how to actually estimate accurately the local errors, and adapt the time step, within the limitations on computational, space and cognitive complexity in large applications.

In general [72], the error estimators are evaluated either by using the same scheme with two different time steps, or by combining schemes with different order of accuracies, or by Milne's device [65], i.e., using two different schemes with the same order of accuracy but different error constants. The first method is robust, but has increased computational and space complexity. We do not address it.

The importance of error estimators and tolerances is widely recognized for stiff and unstable problems: 'the detection of positive eigenvalues is sensitive to the error tolerance and the details of the Newton iteration', see e.g., [82]. The refactorization of the DLN method in [7, 9, 58] indicated that the correct estimator for the local truncation error is the differentiation defect, and it does not involve the interpolation defect [23]. There are several notable members of the DLN family. The value $\delta = \frac{2}{3}$ was suggested in [25], to minimize the error constant and preserve good stability properties. The value $\delta = \frac{2}{\sqrt{5}}$ in [53], ensures the best stability at infinity (a property close to *L*-stability [27]). With $\delta = 1$, the DLN method reduces to the midpoint rule, which has the *smallest error constant* and *conserves all quadratic Hamiltonians*.

Time-adaptive numerical simulations of Navier-Stokes equations (NSE) are often memory bound. Following Shampine and Gordon [32, 72], Gresho, Griffiths, Silvester and Kay used Milne's device to estimate the error in the trapezoidal method [35–38, 50], and showed, how it could be implemented in a memory efficient way. Recently, Bukač and Trenchea [6–8] designed a time adaptive, strongly-coupled partitioned midpoint method for fluid and structure interaction, [3, 10, 26, 64], based on the Milne's device and the AB2 estimator. Using similar ideas for adaptivity, Park, Salgado and Wise introduced in [69] a fast solver for the phase field crystal and functionalized Cahn-Hilliard equations. The G-stability analysis of the DLN method in [58] gave a precise form of its numerical dissipation for incompressible

flows. Using this DLN adaptivity, based on the minimal dissipation criterion of [14] has been tested in [57].

2. Error Estimation using Explicit Schemes. Time adaptivity requires a reliable error estimator. Using the difference between solutions obtained by the DLN method and another higher order method introduces more time levels to be stored, and also depends on the stability of the higher order explicit method.

The first estimator. To address these issues, we use Milne's device [39, 65, 72] to estimate the local truncation error at t_{n+1} by evaluating the difference between the $\mathscr{O}(\Delta t^2)$ DLN-solution y_{n+1}^{DLN} and another second-order approximation, $y_{n+1}^{\widetilde{AB2}}$, obtained by a variable-step Adams-Bashforth 2-like method. Let $\Pi_1(t)$ be the polynomial interpolating $f(t,y(t))$ at nodes $\{t_{n-1}^{\texttt{new}}, t_n^{\texttt{new}}\}$ and values $\{f(t_{n-1}^{\texttt{new}}, y_{n-1}^{\texttt{new}})), f(t_n^{\texttt{new}}, y_n^{\texttt{new}})\}$, which by (refactorized DLN) denote:

$$f(t_{n-1}^{\texttt{new}}, y_{n-1}^{\texttt{new}}) = \frac{\alpha_2 y_{n-1} + \alpha_1 y_{n-2} + \alpha_0 y_{n-3}}{\alpha_2 k_{n-2} - \alpha_0 k_{n-3}},$$
$$f(t_n^{\texttt{new}}, y_n^{\texttt{new}}) = \frac{\alpha_2 y_n + \alpha_1 y_{n-1} + \alpha_0 y_{n-2}}{\alpha_2 k_{n-1} - \alpha_0 k_{n-2}}.$$

Then the solution to the variable step AB2-like method is

$$\begin{aligned}
y_{n+1}^{\widetilde{AB2}} &= y_n + \int_{t_n}^{t_{n+1}} \Pi_1(t)dt \qquad \text{(AB2-like)}\\
&= y_n + \frac{t_{n+1}-t_n}{2(t_n^{\texttt{new}} - t_{n-1}^{\texttt{new}})}\big(- f(t_{n-1}^{\texttt{new}}, y_{n-1}^{\texttt{new}})(t_{n+1}+t_n-2t_n^{\texttt{new}})\\
&\qquad\qquad + f(t_n^{\texttt{new}}, y_n^{\texttt{new}})(t_{n+1}+t_n-2t_{n-1}^{\texttt{new}})\big)\\
&= \Big(1+\alpha_2\frac{(t_{n+1}-t_n)(t_{n+1}+t_n-2t_{n-1}^{\texttt{new}})}{2(t_n^{\texttt{new}}-t_{n-1}^{\texttt{new}})(\alpha_2 k_{n-1}-\alpha_0 k_{n-2})}\Big) y_n\\
&\quad + \frac{(t_{n+1}-t_n)}{2(t_n^{\texttt{new}}-t_{n-1}^{\texttt{new}})}\Big(\alpha_1\frac{(t_{n+1}+t_n-2t_{n-1}^{\texttt{new}})}{(\alpha_2 k_{n-1}-\alpha_0 k_{n-2})} - \alpha_2\frac{(t_{n+1}+t_n-2t_n^{\texttt{new}})}{(\alpha_2 k_{n-2}-\alpha_0 k_{n-3})}\Big) y_{n-1}\\
&\quad + \frac{(t_{n+1}-t_n)}{2(t_n^{\texttt{new}}-t_{n-1}^{\texttt{new}})}\Big(\alpha_0\frac{(t_{n+1}+t_n-2t_{n-1}^{\texttt{new}})}{(\alpha_2 k_{n-1}-\alpha_0 k_{n-2})} - \alpha_1\frac{(t_{n+1}+t_n-2t_n^{\texttt{new}})}{(\alpha_2 k_{n-2}-\alpha_0 k_{n-3})}\Big) y_{n-2}\\
&\quad - \alpha_0\frac{(t_{n+1}-t_n)(t_{n+1}+t_n-2t_n^{\texttt{new}})}{2(t_n^{\texttt{new}}-t_{n-1}^{\texttt{new}})(\alpha_2 k_{n-2}-\alpha_0 k_{n-3})} y_{n-3}.
\end{aligned}$$

In the particular case when $\delta = 1$, the AB2-like method generalizes the estimator in [9, 69] for the midpoint method.

Consequently, under the 'localization assumption', i.e., back values are exact, see e.g., [39, p. 70], [56, p. 56]), we have the following result.

PROPOSITION 2.1. *The local truncation error of the AB2-like method is:*

$$y(t_n+1) - y_{n+1}^{\widetilde{AB2}} \approx y'''(t_n) k_n^3 \mathscr{R}^{(n)}, \tag{2.1}$$

where

$$\mathscr{R}^{(n)} = \frac{1}{12}\Big[2 + \frac{3}{\tau_n}\Big(1 - \beta_2^{(n-2)}\frac{1}{\tau_{n-1}} + \beta_0^{(n-2)}\frac{1}{\tau_{n-2}}\frac{1}{\tau_{n-1}}\Big)\times$$
$$\times\Big(1 - \beta_2^{(n-1)}\frac{1}{\tau_n} + \beta_0^{(n-1)}\frac{1}{\tau_{n-1}}\frac{1}{\tau_n}\Big)$$

$$+\frac{3}{\tau_n}\Big(1+\frac{1}{\tau_n}-\beta_2^{(n-2)}\frac{1}{\tau_{n-1}}\frac{1}{\tau_n}+\beta_0^{(n-2)}\frac{1}{\tau_{n-2}}\frac{1}{\tau_{n-1}}\frac{1}{\tau_n}\Big)\times$$
$$\times\Big(-\beta_2^{(n-1)}+\beta_0^{(n-1)}\frac{1}{\tau_{n-1}}\Big)\Big].$$

We recall [58, Proposition 3.1] that the LTE for the variable step DLN method is

$$y(t_{n+1})-y_{n+1}^{\mathrm{DLN}}\approx -G^{(n)}y'''(t_n)k_n^3, \tag{2.2}$$

where

$$G^{(n)}=\Big(\frac{1}{2}-\frac{\alpha_0}{2\alpha_2}\frac{1-\varepsilon_n}{1+\varepsilon_n}\Big)\Big(\beta_2^{(n)}-\beta_0^{(n)}\frac{1-\varepsilon_n}{1+\varepsilon_n}\Big)^2+\frac{\alpha_0}{6\alpha_2}\Big(\frac{1-\varepsilon_n}{1+\varepsilon_n}\Big)^3-\frac{1}{6}.$$

Subtracting (2.2) from (2.1) yields

$$y'''(t_n)k_n^3\approx\frac{y_{n+1}^{\mathrm{DLN}}-y_{n+1}^{\widetilde{\mathrm{AB2}}}}{G^{(n)}+\mathscr{R}^{(n)}}, \tag{2.3}$$

and therefore combining (2.2) and (2.3), we obtain the following estimator of the LTE for the DLN method.

PROPOSITION 2.2. *The local truncation error of the DLN method can be evaluated by*

$$\textbf{\textit{Estimator 1:}}\quad \widehat{T}_{n+1}=\frac{-G^{(n)}}{G^{(n)}+\mathscr{R}^{(n)}}(y_{n+1}^{\mathrm{DLN}}-y_{n+1}^{\widetilde{\mathrm{AB2}}}). \tag{2.4}$$

The Second Estimator. The critical issue for the above technique is that we need a second order, explicit time-stepping scheme for which the LTE takes the form $\mathrm{LTE}=Ck_n^3+\cdots$. According to this principle, we have many other choices. We develop this next for another explicit, 2-step scheme. Recall [32, 45] the variable step BDF2 method

$$\frac{1+2\tau_n}{1+\tau_n}y_{n+1}-(1+\tau_n)y_n+\frac{\tau_n^2}{1+\tau_n}y_{n-1}=k_nf(t_{n+1},y_{n+1}). \tag{2.5}$$

To derive a second order explicit scheme, we approximate the right hand part of (2.5) by a second-order linear extrapolation, i.e.,

$$f(t_{n+1},y_{n+1})\approx(1+\tau_n)f(t_n,y_n)-\tau_nf(t_{n-1},y_{n-1}),$$

to obtain the following explicit scheme

$$\begin{aligned}\frac{1+2\tau_n}{1+\tau_n}y_{n+1}^{\mathrm{ExBDF2}}-(1+\tau_n)y_n+\frac{\tau_n^2}{1+\tau_n}y_{n-1}\\ =k_n\Big[(1+\tau_n)f(t_n,y_n)-\tau_nf(t_{n-1},y_{n-1})\Big].\end{aligned} \tag{2.6}$$

The LTE of the (2.6) scheme is

$$y(t_{n+1})-y_{n+1}^{\mathrm{ExBDF2}}\approx\frac{(1+\tau_n)^2}{3\tau_n(1+2\tau_n)}y'''(t_n)k_n^3,$$

and the estimator of LTE writes

$$\textbf{Estimator 2:}\quad \widehat{T}_{n+1}=\frac{-G(\varepsilon_n)}{G(\varepsilon_n)+\frac{(1+\tau_n)^2}{3\tau_n(1+2\tau_n)}}(y_{n+1}^{\mathrm{DLN}}-y_{n+1}^{\mathrm{ExBDF2}}). \tag{2.7}$$

Time-step Controllers. The main principle for adaptivity is to adjust the step-size such that the estimator of LTE by DLN is less than or equal to the given tolerance (Tol) [39, 72]. The basic step-size controller for next step size k_{n+1} is

$$\textbf{Basic Controller:}\quad k_{n+1} = \kappa k_n \Big(\mathrm{Tol}/\|\widehat{T}_{n+1}\|_2\Big)^{1/3}, \tag{2.8}$$

where the safety factor $\kappa \in (0,1]$ is selected to minimize the number of step rejections, and $\|\cdot\|_2$ denotes the Euclidean norm in $\mathbb{R}^d$. At each time-step computing, if $\|\widehat{T}_{n+1}\|_2 > \mathrm{Tol}$, then the solution at current time is rejected and the current step k_n is adjusted by (2.8) for recalculation. For the robustness of computation, we may employ the floor for step size k_n, especially for stiff problems. To remove the limit of step size to a large extent, the following improved time step controller, based on the basic step controller (2.8) was proposed by Hairer and Wanner in [46]:

Improved Controller:

$$k_{n+1} = k_n \cdot \min\Big\{1.5, \max\Big\{0.2, \kappa\big(\mathrm{Tol}/\|\widehat{T}_{n+1}\|\big)^{1/3}\Big\}\Big\}. \tag{2.9}$$

Another way of avoiding the minimum time-step restriction was proposed by Söderlind and Wang in [74, 76]. They replaced the term $(\mathrm{Tol}/\|\widehat{T}_{n+1}\|)$ in (2.8) by the geometric average of of previous LTEs $(\mathrm{Tol}/\|\widehat{T}_{n+1}\|)$, $(\mathrm{Tol}/\|\widehat{T}_n\|)$ and $(\mathrm{Tol}/\|\widehat{T}_{n-1}\|)$, namely,

$$k_{n+1} = k_n \big(\mathrm{Tol}/\|\widehat{T}_{n+1}\|\big)^{\lambda_1} \big(\mathrm{Tol}/\|\widehat{T}_n\|\big)^{\lambda_2} \big(\mathrm{Tol}/\|\widehat{T}_{n-1}\|\big)^{\lambda_3} \tau_n^{-\eta_2} \tau_{n-1}^{-\eta_3}.$$

The values of λ and η are decided by the order of dynamics of the closed loop system [74].

We summarize the adaptive algorithm of the DLN method with **Estimator 1** (2.4) or **Estimator 2** (2.7) of LTE and the step controller (2.9) in Algorithm 1.

Algorithm 1: Adaptivity with **Estimator 1** or **Estimator 2** of LTE and step size controller in (2.9)

Input: tolerance Tol, initial value y_1, initial stepsize k_1, safety factor κ, time interval $[T_1, T_2]$;

$n \Leftarrow 1$; $t_n \Leftarrow T_1$;

$t_{n+1} \Leftarrow t_n + k_n$; `// update the current time`

compute y_{n+1} by one-step method (e.g. backward Euler);

$n \Leftarrow n+1$, $k_n \Leftarrow k_{n-1}$;

while $t_n + k_n < T_2$ **do**

$t_{n+1}^{\text{temp}} = t_n + k_n$; $\tau_n = k_n / k_{n-1}$; `// update current time and step size ratio`

compute y_{n+1}^{DLN} ; `// find the DLN solution`

compute y_{n+1}^{AB2} if **Estimator 1** used or compute y_{n+1}^{ExBDF2} if **Estimator 2** used ;

$\widehat{T}_{n+1} \Leftarrow \left\| \frac{G^{(n)}}{G^{(n)} + (\frac{1}{6} + \frac{1}{4\tau_n})} (y_{n+1}^{\text{DLN}} - y_{n+1}^{\text{AB2}}) \right\|$ or

$\widehat{T}_{n+1} \Leftarrow \left\| \frac{G(\varepsilon_n)}{G(\varepsilon_n) + \frac{(1+\tau_n)^2}{3\tau_n(1+2\tau_n)}} (y_{n+1}^{\text{DLN}} - y_{n+1}^{\text{ExBDF2}}) \right\|$;

if $\widehat{T}_{n+1} < \text{Tol}$ **then**

$t_{n+1} \Leftarrow t_{n+1}^{\text{temp}}$; `// accept current estimator for LTE`

$k_{n+1} \Leftarrow k_n \cdot \min\left\{1.5, \max\left\{0.2, \kappa\left(\frac{\text{Tol}}{\|\widehat{T}_{n+1}\|}\right)^{1/3}\right\}\right\}$; `// adjust step by (2.9)`

$y_{n+1} \Leftarrow y_{n+1}^{\text{DLN}}$; `// accept result`

$n \Leftarrow n+1$; `// come to next time step`

else

$k_n \Leftarrow k_n \cdot \min\left\{1.5, \max\left\{0.2, \kappa\left(\frac{\text{Tol}}{\|\widehat{T}_{n+1}\|}\right)^{1/3}\right\}\right\}$; `// adjust step`

3. Error estimation by Refactorization. In this section, we present another adaptive algorithm, via the refactorization of the DLN method with parameter $\delta \in (0,1)$, and the midpoint rule on the interval $[t_n^{\text{old}}, t_{n+1}]$, where $t_{n+1}^{\text{new}} = \frac{t_n^{\text{old}} + t_{n+1}}{2}$. Since

$$t_{n+1}^{\text{new}} - t_n^{\text{old}} = t_{n+1} - t_{n+1}^{\text{new}} = (\Delta t)_n^{\text{BE}},$$

we can use the midpoint rule to obtain another solution (denoted $\widetilde{y}_{n+1}$) at t_{n+1}, as an extrapolation of the y_{n+1}^{new} and y_n^{old} values [9, 42]

$$\widetilde{y}_{n+1} = 2y_{n+1}^{\text{new}} - y_n^{\text{old}}. \tag{3.1}$$

In Proposition 3.1 we will show that $\widetilde{y}_{n+1}$ is a first-order approximation to $y(t_{n+1})$, and thus the difference between y_{n+1}^{DLN} (a second-order approximation) and $\widetilde{y}_{n+1}$ works as an estimator for the DLN LTE:

$$\textbf{Estimator 3:}\quad \widehat{T}_{n+1} = y_{n+1}^{\text{DLN}} - \widetilde{y}_{n+1}. \tag{3.2}$$

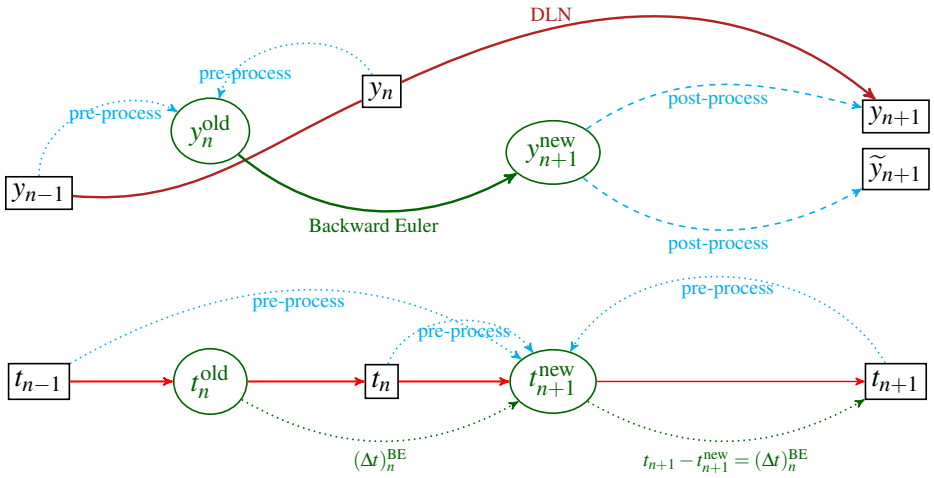


Fig. 3.1: **Estimator of LTE by the Refactorization Algorithm**

We summarize the refactorization process of the two solutions y_{n+1}^{DLN} (the DLN solution) and $\widetilde{y}_{n+1}$ in Algorithm 2 and Figure 3.1.

Algorithm 2: Estimator of LTE by the Refactorization Algorithm

Input: y_n, y_{n-1} and t_{n-1}, t_n, t_{n+1} ;

// Pre-process

$(\Delta t)_n^{\text{BE}} \Leftarrow b^{(n)} \widehat{k}_n$; // time-step for BE

$t_{n+1}^{\text{new}} \Leftarrow \beta_2^{(n)} t_{n+1} + \beta_1^{(n)} t_n + \beta_0^{(n)} t_{n-1}$; // $[t_{n+1}^{\text{new}} - (\Delta t)_n^{\text{BE}}, t_{n+1}^{\text{new}}]$ BE interval

$y_n^{\text{old}} \Leftarrow a_1^{(n)} y_n + a_0^{(n)} y_{n-1}$;

// backward Euler

Solve for y_{n+1}^{new}: $\dfrac{y_{n+1}^{\text{new}} - y_n^{\text{old}}}{(\Delta t)_n^{\text{BE}}} = f\left(t_{n+1}^{\text{new}}, y_{n+1}^{\text{new}}\right)$

// Post-process : extrapolation

$y_{n+1}^{\text{DLN}} \Leftarrow c_2^{(n)} y_{n+1}^{\text{new}} + c_1^{(n)} y_n + c_0^{(n)} y_{n-1}$; // the DLN solution

$\widetilde{y}_{n+1} \Leftarrow 2 y_{n+1}^{\text{new}} - y_n^{\text{old}}$; // first order solution for adaptivity

$\widehat{T}_{n+1} \Leftarrow \| y_{n+1}^{\text{DLN}} - \widetilde{y}_{n+1} \|$; // Estimator of LTE

The step size controller for Algorithm 2 becomes

$$k_{n+1} = k_n \cdot \min\left\{1.5, \max\left\{0.2, \kappa\left(\text{Tol}/\|\widehat{T}_{n+1}\|\right)^{1/2}\right\}\right\}, \tag{3.3}$$

since $\widetilde{y}_{n+1}$ is first order accurate. Then we have the following adaptive DLN Algorithm 3, using the (3.3) step size controller.

Next we prove the consistency error in evaluating the $\widetilde{y}_{n+1}$ solution by the midpoint rule in Algorithm 2.

PROPOSITION 3.1. *The numerical solution $\widetilde{y}_{n+1}$ in Algorithm 2 is a first-order approxi-*

Algorithm 3: Adaptivity with refactorization (Algorithm 2) and step size controller in (3.3)

Input: tolerance Tol, initial value y_1, initial stepsize k_1, safety factor κ, time interval $[T_1, T_2]$;

$n \Leftarrow 1$; $t_n \Leftarrow T_1$;

$t_{n+1} \Leftarrow t_n + k_n$; `// update the current time`

compute y_{n+1} by one-step method (e.g. backward Euler);

$n \Leftarrow n+1$, $k_n \Leftarrow k_{n-1}$;

while $t_n + k_n < T_2$ **do**

  $t_{n+1} = t_n + k_n$;

  Input y_n, y_{n-1} and t_{n-1}, t_n, t_{n+1} in Algorithm 2 and obtain y_{n+1}^{DLN} and $\widehat{T}_{n+1}$;

  if $\widehat{T}_{n+1} < \mathrm{Tol}$ **then**

    $k_{n+1} \Leftarrow k_n \cdot \min\left\{1.5, \max\left\{0.2, \kappa\left(\frac{\mathrm{Tol}}{\|\widehat{T}_{n+1}\|}\right)^{1/2}\right\}\right\}$; `// adjust step by (2.9)`

    $y_{n+1} \Leftarrow y_{n+1}^{\mathrm{DLN}}$; `// accept result`

    $n \Leftarrow n+1$; `// come to next time step`

  else

    $k_n \Leftarrow k_n \cdot \min\left\{1.5, \max\left\{0.2, \kappa\left(\frac{\mathrm{Tol}}{\|\widehat{T}_{n+1}\|}\right)^{1/2}\right\}\right\}$; `// adjust step`

mation of $y(t_{n+1})$, with the local truncation error

$$\mathscr{L} \approx \frac{(1+\varepsilon_n)}{2b^{(n)}\left[(\alpha_2-\alpha_0)+\varepsilon_n(\alpha_2+\alpha_0)\right]}\mathscr{F}(\delta,\varepsilon_n)\,y''(t_{n+1}^{\mathrm{new}})k_n, \tag{3.4}$$

where

$$\mathscr{F}(\delta,\varepsilon_n) := \left(1-2\beta_2^{(n)}\right)+2\left(\beta_0^{(n)}-a_0^{(n)}\beta_2^{(n)}\right)\frac{1-\varepsilon_n}{1+\varepsilon_n}+a_0^{(n)}\left(2\beta_0^{(n)}-1\right)\left(\frac{1-\varepsilon_n}{1+\varepsilon_n}\right)^2.$$

Proof. The LTE of the midpoint method on the interval $[t_n^{\mathrm{old}}, t_{n+1}]$

$$\frac{\widetilde{y}_{n+1}-y_n^{\mathrm{old}}}{t_{n+1}-t_n^{\mathrm{old}}} = f\left(t_{n+1}^{\mathrm{new}}, y_{n+1}^{\mathrm{new}}\right)$$

is, by definition,

$$\mathscr{L} = \frac{1}{t_{n+1}-t_n^{\mathrm{old}}}\left\{y(t_{n+1})-\left(a_1^{(n)}y(t_n)+a_0^{(n)}y(t_{n-1})\right)\right\}-f\left(t_{n+1}^{\mathrm{new}}, y(t_{n+1}^{\mathrm{new}})\right).$$

Then using Taylor expansions for all the terms, at time t_{n+1}^{new}, yields (3.4). In order to complete the proof and show that $\widetilde{y}_{n+1}$ is only a first-order approximation, it only remains to show that $\mathscr{F}(\delta,\varepsilon_n)$ is strictly negative, since

$$\frac{1+\varepsilon_n}{(\alpha_2-\alpha_0)+\varepsilon_n(\alpha_2+\alpha_0)} > 0.$$

Indeed, for fixed $\delta \in (0,1)$, and for all $-1 < \varepsilon_n < 1$, we have

$$\lim_{\varepsilon_n \to -1}\mathscr{F}(\delta,\varepsilon_n) = -\infty, \quad \mathscr{F}(\delta,1) = 0,$$

and that $\mathscr{F}(\cdot,\varepsilon_n)$ is non-decreasing $\frac{d\mathscr{F}(\delta,\varepsilon_n)}{d\varepsilon_n} > 0$. □

4. Numerical Tests. We organize our results in two categories. In Section 4.1 we present two tests, using the constant time step DLN methods. We verify the second-order convergence, and also compare the stability and performance of the DLN methods versus other popular second-order methods. In Section 4.2 we compare our adaptive DLN algorithms with some of the well-established time-adaptive solvers.

4.1. Constant Step Tests. In this section, we implement the constant time-stepping DLN algorithm with three particular values of the parameter $\delta = 2/3, 2/\sqrt{5}, 1$. The value $\delta = 2/3$ was suggested in [25] in order to minimize the error constant and preserve good stability properties. The value $\delta = 2/\sqrt{5}$ was proposed in [53, 54] to ensure the best stability at infinity. In the case $\delta = 1$, the constant time step DLN method reduces to the symplectic midpoint rule, having the smallest error constant (see Table 4.3) and conserving all quadratic Hamiltonians.

4.1.1. Quasi-periodic oscillations. In order to test the second-order convergence of the DLN methods, we consider the following quasi-periodic oscillations [43]

$$\begin{aligned} &y^{(4)} + (\pi^2+1)y'' + \pi^2 y = 0, \quad 0 \le t \le 20, \\ &y(0) = 2,\ y'(0) = 0,\ y''(0) = -(1+\pi^2),\ y'''(0) = 0, \end{aligned}$$

with has the exact solution $y(t) = \cos(t) + \cos(\pi t)$. Let $e_n = y(t_n) - y_n$ denote the error at time t_n, and

$$\|e\|_{2,\infty} = \max_{1\le n\le N}\{\|e_n\|_2\}, \qquad \|e\|_{2,2} = \left(\Delta t \sum_{n=1}^{N} \|e_n\|_2^2\right)^{1/2}.$$

be the notations for the $L^\infty(0,T)$ and $L^2(0,T)$ discrete norms. In Tables 4.1 and 4.2 we summarize the errors and the rates of convergence for the constant step DLN methods, with the specified parameters $\delta = 2/3, 2/\sqrt{5}$, and 1. The values from the Tables 4.1, 4.2, and

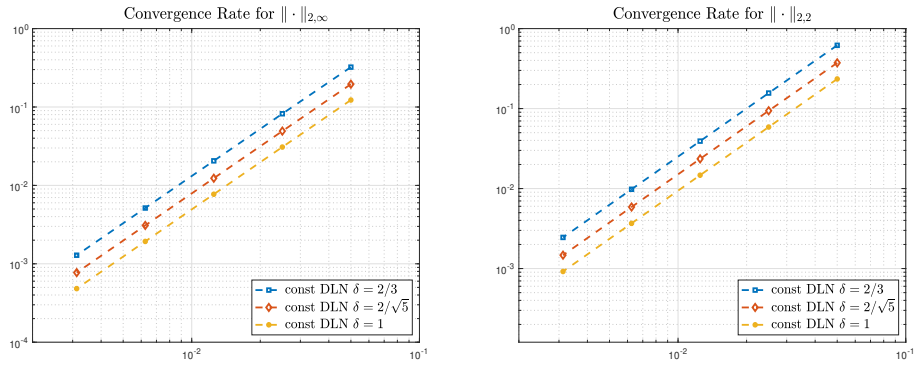
Step size	$\delta = \frac{2}{3}$	Rate	$\delta = \frac{2}{\sqrt{5}}$	Rate	$\delta = 1$	Rate
0.05	0.32233672	-	0.19537687	-	0.12271718	-
0.025	0.08202388	1.9745	0.04926517	1.9876	0.03084194	1.9924
0.0125	0.02056438	1.9959	0.01234158	1.9970	0.00771706	1.9988
0.00625	0.00514472	1.9990	0.00308709	1.9992	0.00192962	1.9997
0.003125	0.00128642	1.9997	0.00077188	1.9998	0.00048244	1.9999

Table 4.1: The predicted second-order convergence $\|\cdot\|_{2,\infty}$ is observed in the test.

Step size	$\delta = \frac{2}{3}$	Rate	$\delta = \frac{2}{\sqrt{5}}$	Rate	$\delta = 1$	Rate
0.05	0.61799316	-	0.37320014	-	0.23460108	-
0.025	0.15634451	1.9829	0.09391299	1.9906	0.05876962	1.9971
0.0125	0.03917128	1.9969	0.02350951	1.9981	0.01469880	1.9994
0.00625	0.00979800	1.9992	0.00587936	1.9995	0.00367508	1.9998
0.003125	0.00244989	1.9998	0.00146999	1.9999	0.00091879	2.0000

Table 4.2: The predicted second-order convergence $\|\cdot\|_{2,2}$ is observed in the test.

the plots in Figure 4.1 confirm the second-order convergence for all three constant time-step



(a) (b)

Fig. 4.1: log-log plot of Δt vs. errors for constant step DLN algorithms, Tables 4.1 and 4.2.

DLN methods. We also note that in all cases, both errors are decreasing as the value of δ is increasing.

In Section A, we shall use this test problem to also test the DLN adaptive algorithms described in Section 2 and Section 3.

4.1.2. Boosted harmonic motion. We compare the errors of DLN with BE, BE plus time filter, and BDF2 for the problem

$$x' = \mu x + \frac{1}{\mu} y, \quad y' = -\frac{1}{\mu} x + \mu y, \quad x(0) = 1, \quad y(0) = 0, \quad \mu = 1.\text{e}{-2}, \tag{4.1}$$

on the time interval $[0, 20]$. The exact solution of (4.1) has a sinusoidal form $x(t) = e^{\mu t}\cos(t/\mu)$, $y(t) = e^{\mu t}\sin(t/\mu)$, with an amplitude $e^{\mu t}$, i.e., it represents a growing spiral in the phase plane. This is apparent when we write the system as a second order equation

$$x'' - 2\mu x' + \big(\mu^2 + \frac{1}{\mu^2}\big)x = 0, \qquad x(0) = 1, \quad x'(0) = 0,$$

and note that it has a boosting ('negative' damping) term. We recall that (4.1) is related to 'Dahlquist's stability test' $u' = \lambda u$, where the eigenvalue $\lambda = \mu + i\frac{1}{\mu}$. We also mention that unlike the common case, here λ lies on the right side of the complex plane, with a positive small real part μ and a much larger imaginary part $\frac{1}{\mu}$.

We shall compare the DLN methods versus the original BE, and two other commonly used second-order methods, namely the 'BE plus time filter' (with parameter $\nu = 2/3$) [43] and BDF2. From backward error analysis, looking at the modified equation ($z' + \frac{\Delta t}{2} z'' = f(z)$), it is expected that BE to have a very dissipative effect. (Indeed, the second order derivative turns the high frequency into dissipation.) The numerical solutions are influenced both by the shape of the methods' stability regions, and by the consistency order and the size of error constants. All the methods are A-stable, see e.g., Figure 4.2, but in this case, the eigenvalue λ is not supposed to be damped. The regions of absolute stability for the DLN methods (corresponding to $\delta = 2/3$ and $\delta = 2/\sqrt{5}$) do not differ much from those of the BE, BE plus time filter and BDF2 methods. The exception is the midpoint rule, (DLN with $\delta = 1$), which has as its 'stability region' exactly the left side of the complex plane. In particular, the 'instability region' being the right half-plane, makes the midpoint method ideal for both

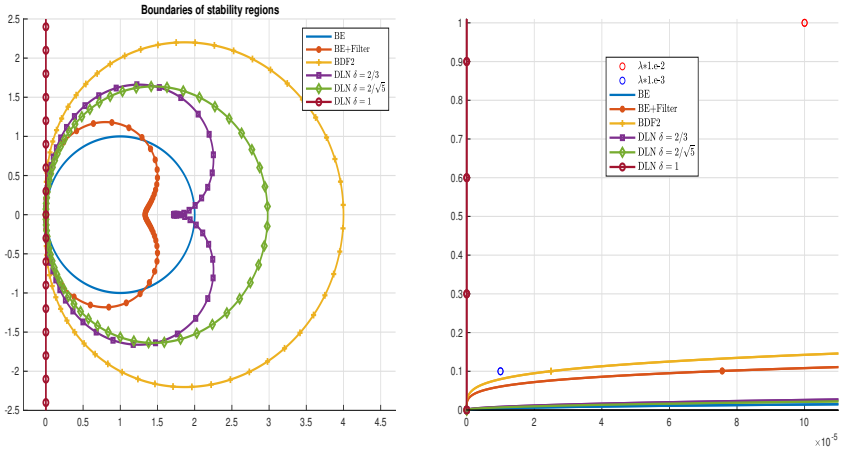
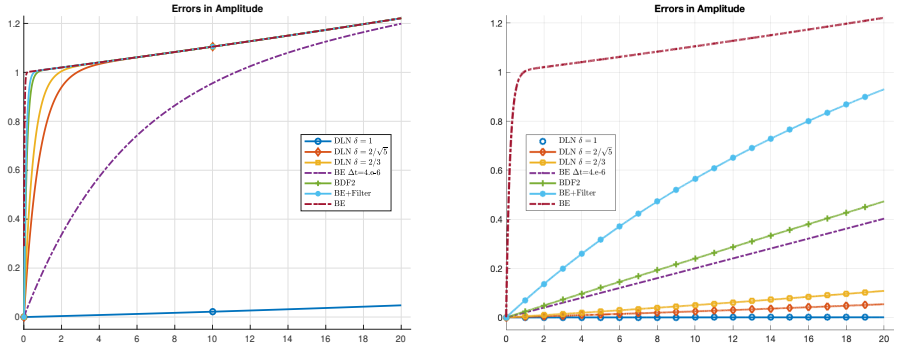


Fig. 4.2: The boundaries of the stability regions of the BE, BE Plus Time Filter, BDF2 and the DLN methods with $\delta = 2/3, 2/\sqrt{5}$ and $\delta = 1$. The plot on the right represents a zoomed frame, showing that for both time steps $\Delta t = 1.\text{e}-2$ and $\Delta t = 1.\text{e}-3$, the values of $\lambda * \Delta t$ are inside of the stability regions for all the methods, except the midpoint rule. This explains the large errors in Figure 4.3 (a).

stable and unstable problems. This is epitomized by the stiff and unstable problem proposed by Lindberg, presented in Section 4.2.3. The error constants for the BE+Filter, BDF2 and



(a) (b)

Fig. 4.3: The amplitudes of the errors of the boosted harmonic motion (4.1) for the DLN methods ($\delta = 2/3, 2/\sqrt{5}, 1$), BE, BE + time filter', BDF2, with $\mu = 1.\text{e}-2$. The left plot corresponds to the constant time step $\Delta t = 1.\text{e}-2$, while the right plot has $\Delta t = 1.\text{e}-3$. Note that the first-order method BE, even with the smaller time step $\Delta t = 4.\text{e}-6$ incorrectly damps the amplitude.

the constant step DLN methods are summarized in Table 4.3. We recall that the midpoint method's error constant is the smallest among all variable-step G-stable one-leg second-order

accurate multistep methods [58]. We note that there is an order of magnitude difference between midpoint's error constant $(1/24)$ and the error constants of the 'BE + filter' $(5/6)$ or BDF2 $(2/9)$ methods. The 'BE + time filter' error constant is 20 times larger (while BDF2 is about 5.3 times larger) than the midpoint error constant. In Figure 4.3 we report

BE+Filter ($\nu = 2/3$)	BDF2	$\delta = 2/3$	$\delta = 2/\sqrt{5}$	$\delta = 1$
$5/6 \approx 0.8333$	$\frac{2}{9} \approx 0.2222$	$\frac{2}{15} \approx 0.1333$	0.070382	$\frac{1}{24} \approx 0.0417$

Table 4.3: The errors constants in LTEs for the second-order methods BE+Filter, BDF2, and the constant step DLN methods $\delta = \frac{2}{3}, \frac{2}{\sqrt{5}}$ and $\delta = 1$ (the midpoint rule).

the results for the amplitudes of the solutions corresponding to two values the constant time step $\Delta t = 1.\mathrm{e}-2$ (left), and respectively with $\Delta t = 2.\mathrm{e}-3$ (right). For completeness, we also included the result corresponding to the BE method with the time step $\Delta t = 4.\mathrm{e}-6$. We note that for $\mu = 1.\mathrm{e}-2$ and $\Delta t = 1.\mathrm{e}-2$, all the methods (except for the midpoint rule) are highly dissipative, driving the solution to zero. For $\Delta t = 1.\mathrm{e}-3$, the DLN methods have an improved performance over BDF2 and BE plus time filter. The BE method, even with a smaller time step $\Delta t = 4.\mathrm{e}-6$, is over dissipating. Finally, we note that for the considered values of the time step, with the given μ, the values of $\lambda * \Delta t$ are close to the imaginary axis, on the right side of the complex plane, and all lie inside the stability regions shown in Figure 4.2, except for the midpoint method.

4.2. Variable Step Tests. In this subsection, we evaluate the behavior of the adaptive variable time-stepping DLN methods, with $\delta = 2/3, 2/\sqrt{5}$ and 1, on several test problems. In all cases we use Algorithm 1 (with Milne's device and AB2-like for the estimator of LTE) and Algorithm 3 *without* setting a minimum step size.

4.2.1. Lotka-Volterra Equations. We test conservation of a Hamiltonian for DLN compared with other methods. There is no exact solution, and therefore non-conservation indicates a clear error. We apply the adaptive variable time-stepping DLN methods via Algorithm 1 and Algorithm 3 to the Lotka-Volterra equations [62, 63, 66, 81]

$$\begin{aligned} &x' = 2x - xy, \quad y' = -y + xy, \\ &x(0) = 4, \quad y(0) = 2, \quad \text{with Hamiltonian} \\ &H(x,y) = x - \ln x + y - 2\ln y. \end{aligned} \tag{4.2}$$

We set the time interval to $[0, 500]$, and the initial time step $k_1 = 1.\mathrm{e}-4$, *without* restrictions on the time steps. We use two tolerances Tol=1.e$-$6 and Tol=1.e$-$8 for Algorithm 1, and Tol=1.e$-$6 for Algorithm 3. Then we compare results of the adaptive DLN methods versus the results of the constant step DLN algorithms, and also those of the MATLAB ode15s, ode23 and ode45 functions (with relative tolerance 1.e$-$6 and absolute tolerance 1.e$-$6). Within these settings, all the solutions are indistinguishable in the phase plane. Therefore we monitor the conservation of the Hamiltonian function. The results are displayed in Figure 4.4 and number of steps are summarized in Table 4.4. The adaptive DLN algorithm 1 with Tol = 1.e-8 is *more accurate* than algorithm 1 with Tol = 1.e-6, but it requires (an order of magnitude) more time steps. The adaptive DLN algorithm 3 with Tol = 1.e-6 has a similar behavior to algorithm 1 with Tol = 1.e-8, however it entails an even larger number of time steps. The accuracy of each algorithm improves as the parameter δ decreases. The Matlab `ode45` function performs better than the `ode15s` and `ode23` functions. All three Matlab algorithms require a smaller number of time steps than either the adaptive DLN algorithms

1 and 3. The constant time step DLN methods have a relatively good performance, but the Hamiltonian has larger oscillations, especially for small values of δ.

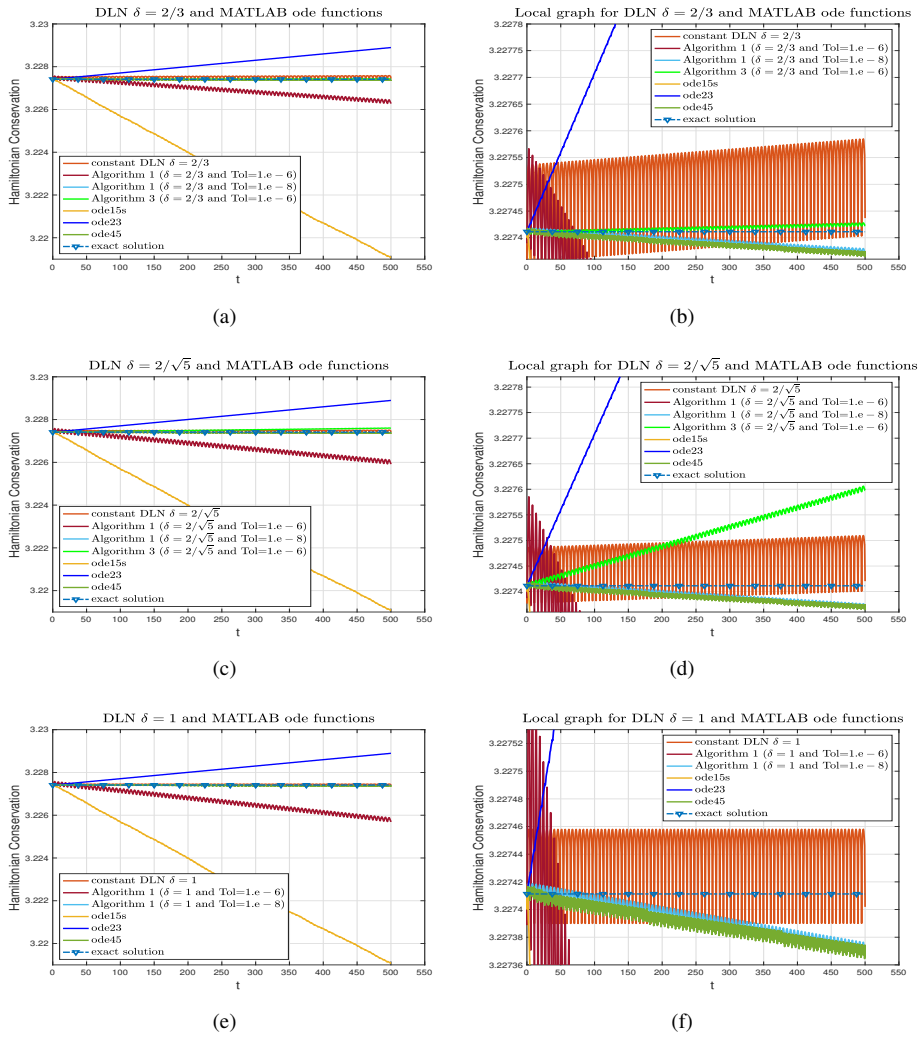


Fig. 4.4: Testing conservation of Hamiltonian for (4.2). The Adaptive DLN approximates the Hamiltonian better than the `ode45`, `ode15s` and `ode23`. The constant step DLN methods have a relatively good performance, but the Hamiltonian has larger oscillations, especially for small values of δ.

4.2.2. Van der Pol's equation. We test the phase error of DLN constant versus adaptive step for the Van der Pol's equation [2, 15, 80]

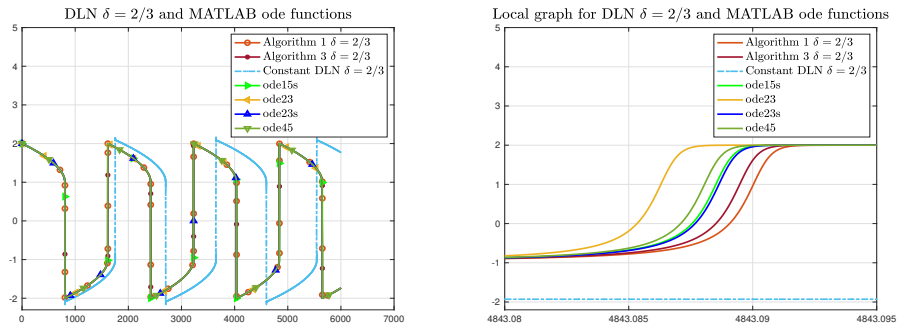
$$x'' - \mu(1-x^2)x' + x = 0, \quad x(0) = 2, \quad x'(0) = 0, \tag{4.3}$$

with the parameter $\mu = 1000$. The problem (4.3) is considered to be very stiff when the parameter μ is large, and is commonly used to test adaptive solvers [9, 44, 46, 54]. We examine the solutions of (4.3) computed by the constant step DLN methods with $\delta = 2/3$

	Const. Δt DLN	Alg.1 (Tol=1.e-6)	Alg.3 (1.e-6)
$\delta = 2/3$	100,000	79,364	900,497
$\delta = 2/\sqrt{5}$	100,000	58,122	924,047
$\delta = 1$	100,000	46,619	-
	`ode15s`	`ode23`	`ode45`
	11,879	29,599	14,149

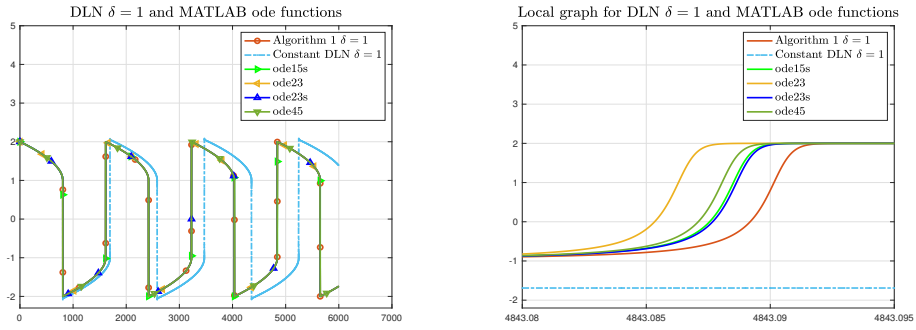
Table 4.4: Number of steps of DLN algorithms and MATLAB's ode solvers, for Lotka-Volterra system (4.2).

and $\delta = 1$, the adaptive DLN algorithms (Algorithm 1 and Algorithm 3), and the MATLAB `ode15s`, `ode23`, `ode23s` and `ode45` functions over $0 \leq t \leq 6000$. Due to stiffness, in order to avoid pre-setting the minimum step size, we use the tolerance Tol = 1.3e-6 and the safety factor $\kappa = 0.65$ for the DLN algorithms. The initial time step is set to be k_1=1.e-4. For the `ode23` and `ode45` functions we set both the relative tolerance and absolute tolerance to AbsTol = RelTol = 1.e-6. We set the relative tolerance RelTol = 1.e-10 and the absolute tolerance AbsTol = 1.e-15 for the `ode15s` and `ode23s` functions, and used the corresponding solutions as reference. The Figures 4.5 and 4.6 show the solutions and zoomed in frames of the adaptive DLN and the constant time step DLN methods, corresponding to $\delta = 2/3$ and $\delta = 1$, versus the MATLAB `ode23`, `ode45` and the reference solutions by MATLAB `ode15s`, `ode23s`. The Table 4.5 summarizes the number of time steps required by each method. The Figures 4.5(a) and 4.6(a) show that the constant step DLN solutions lag behind the other solutions, despite of using a large number of time steps. The details in Figures 4.5(b) and 4.6(b) confirm that *the adaptive DLN Algorithms 1 and 3 perform well for this stiff problem,* while from Table 4.5 we infer that the adaptive Algorithms 1 *(based on the midpoint rule and Milne's device with AB2-like) is the most efficient by far.*



(a) (b)

Fig. 4.5: The solutions and a zoomed in frame for the Van der Pol (4.3) solutions of the adaptive DLN and the constant time step DLN methods with $\delta = 2/3$, versus the MATLAB `ode23`, `ode45` and the reference solutions by MATLAB `ode15s`, `ode23s` (with RelTol = 1.e-10 and AbsTol = 1.e-15). All methods capture the limit cycle, while the constant step DLN exhibits a phase error. The Adaptive DLN outperforms the MATLAB ode solvers, see e.g. Table 4.5.



(a) (b)

Fig. 4.6: Nearly identical results were found for DLN with $\delta = 1$. The solutions and a zoomed in frame for the Van der Pol (4.3) solutions of the adaptive DLN and the constant time step DLN methods with $\delta = 1$, versus the MATLAB `ode23`, `ode45` and the reference solutions by MATLAB `ode15s`, `ode23s` (with RelTol = 1.e-10 and AbsTol = 1.e-15).

	Constant step DLN	Algorithm 1	Algorithm 3	
$\delta = 2/3$	6,000,000	62,806	769,319	
$\delta = 1$	6,000,000	32,379	-	
	`ode15s`	`ode23`	`ode23s`	`ode45`
	19,714	4,377,590	1,593,283	13,242,893

Table 4.5: The number of time steps used by the DLN and the MATLAB `ode` solvers for (4.3). The Adaptive DLN is by far the most efficient.

4.2.3. The Lindberg Example. Finally we examine the behavior of the adaptive DLN algorithms on a stiff problem, where normal methods fail by both underflow and overflow errors, and for which the eigenvalues of the Jacobian evolve from large negative to large positive values. The example

$$\begin{cases} y_1' = 10^4 y_1 y_3 + 10^4 y_2 y_4, \\ y_2' = -10^4 y_1 y_4 + 10^4 y_2 y_3, \\ y_3' = 1 - y_3, \\ y_4' = -0.5 y_3 - y_4 + 0.5, \end{cases} \qquad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} (0) = \begin{bmatrix} 1 \\ 1 \\ -1 \\ 0 \end{bmatrix}.$$

was proposed by Lindberg [61], and was used by Watanabe and Sheikh [82] to test the performance of DIFSUB and DIFSOL solvers [73] and their effectiveness in detecting unstable problems. This system is exactly solvable, by first solving for the y_3, y_4 unknowns, and then regarding the first two equations as a 2×2 linear system

$$\begin{bmatrix} y_1' \\ y_2' \end{bmatrix} = 10^4 \begin{bmatrix} y_3 & y_4 \\ -y_4 & y_3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}.$$

The exact solution is (see Figure 4.7)

$$\begin{cases} y_1 = e^{g_1(t)}\big[\cos(g_2(t)) + \sin(g_2(t))\big], \\ y_2 = e^{g_1(t)}\big[\cos(g_2(t)) - \sin(g_2(t))\big], \\ y_3 = 1 - 2e^{-t}, \\ y_4 = te^{-t}, \end{cases} \qquad \text{where} \begin{cases} g_1(t) = 10^4(t + 2e^{-t} - 2) \\ g_2(t) = 10^4(1 - e^{-t} - te^{-t}) \end{cases}.$$

This example is interesting due to the fact that the real part of eigenvalues

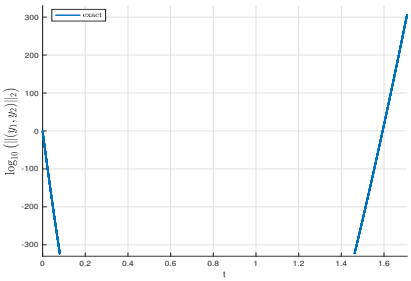


Fig. 4.7: The base 10 logarithmic plot of the Euclidean norm of the (y_1, y_2) components of the solution to the Lindberg example 4.2.3. The exact solution on the time interval $[0, 1.7085]$ exhibits both underflow and overflow. The plot spotlights the unstable character of the problem, the solution growing quickly from 10^{-322} to 10^{307} on the subinterval $[1.4622, 1.7085]$.

$$\lambda_{1,2} = 10^4(1 - 2e^{-t} \pm ite^{-t}),$$

of the matrix solving for y_1, y_2

$$10^4 \begin{bmatrix} 1 - 2e^{-t} & te^{-t} \\ -te^{-t} & 1 - 2e^{-t} \end{bmatrix},$$

varies from -10^4 to 10^4 in the time interval $t \in [0, 4]$. The exact solution (y_1, y_2) becomes very small fast, namely $(y_1, y_2)(0.0807) \approx (-9.8813, 9.8813) * 10^{-324}$, which is below the smallest normalized number in double precision [24], see Figure 4.7. Moreover, within a short time interval $t \in [1.5, 1.7]$, the (y_1, y_2) exact solution grows from $(-5.7788, -1.7968) * 10^{-234}$ to $(-6.3883, 8.7874) * 10^{283}$.

We now test the ability of the adaptive DLN Algorithm 1 (with Milne's device for the estimator of the LTE and the AB2-like solution) and Algorithm 3 to detect the change in the sign and size of the eigenvalues, and therefore the change in the behavior of the problem, from a stable to an unstable regime.

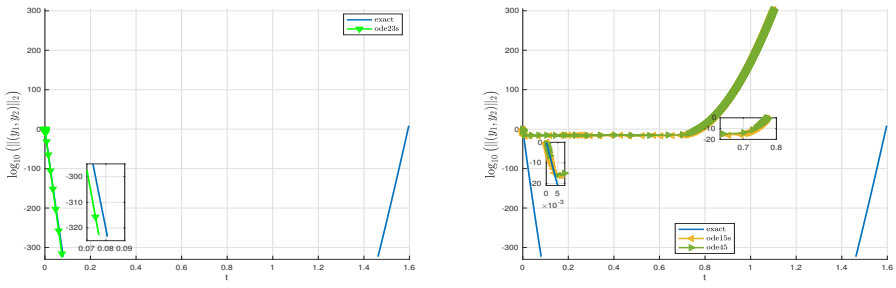
(1) Taking into consideration its stiffness, and in order to avoid overflow, we simulate the Lindberg example on the time interval $[0, 1.597]$.

(i) For comparison we also use the MATLAB `ode45`, `ode15s` and `ode23s` functions with the relative and absolute tolerances RelTol=1.e-11, AbsTol=1.e-15 [60]. *Despite these values for the tolerances, the MATLAB* `ode45`, `ode15s` *and* `ode23s` *solvers fail to approximate well the exact solution.* Namely, `ode23s` approximates the problem correctly up to the time $t = 1.102216$, when it fails due to an integration tolerance error, caused by using a step size below the smallest value allowed 3.552714e-15, as seen in Figure 4.8(a). On the other

hand, `ode15s` and `ode45` both decrease in the Euclidean norm only to 10^{-15} at around $t = 0.004$, and then stay at this value up to almost $t = 0.8$, when they blow up, see Figure 4.8(b).

	$\delta = 2/3$	$\delta = 2/\sqrt{5}$	$\delta = 1$
Algorithm 1	0.79 e-15	0.719 e-15	1.01 e-14
Number of time steps	1,672,599	1,565,431	1,478,057

Table 4.6: The absolute tolerances used for Algorithm 1 in the Lindberg example, on $t \in [0.0, 1.597]$ and with minimal step size 10^{-8}.



(a) (b)

Fig. 4.8: The base 10 logarithmic plot of the Euclidean norm of the (y_1, y_2) components of the solution to the Lindberg example 4.2.3. In the left plot 4.8(a), the `ode23s` function (with AbsTol =1.0e-15, and RelTol = 1.0e-11) fails at time $t = 1.102216$, due to an integration tolerance error, caused by using a step size below the smallest value allowed 3.552714e-15. In the right plot 4.8(b), the solutions obtained by `ode15s` and `ode45` (with AbsTol =1.0e-15, and RelTol = 1.0e-11) both decrease in the Euclidean norm to 10^{-15} around $t = 0.004$, and stay at this value up to around $t = 0.8$, when they blow up.

(ii) The DLN methods via Algorithm 1, with the tolerances from Table 4.6, and the initial and minimal step sizes equal to 10^{-8}, all approximate well the exact solution. The values of the steps sizes are shown in Figure 4.10(a). As seen in Figure 4.9, for all three values of $\delta = 2/3, 2/\sqrt{5}, 1$, the approximate solutions decay fast at the beginning of the time interval, stay at a value close to underflow, and sense at the 'right' time the change in the sign of the eigenvalue. There is a difference in the size of the solutions at the final time $t = 1.597$. Namely, the $\|(y_1, y_2)\|_2$ of the exact solution is around 10^8, while the DLN solutions are approximately at 10^{30}.

(iii) We mention that the DLN solutions ($\delta = 2/3, 2/\sqrt{5}$, and 1) corresponding to *the constant step size methods*, with the step sizes $\Delta t \approx$9.5480e-07, $\Delta t \approx$1.0202e-06 and $\Delta t \approx$1.0805e-06 respectively, *do not approximate well the exact solution* at the end of the interval. Indeed, as it can be seen in Figure 4.10(b), the constant time step solutions do not observe the instability towards the end of the interval, remaining at a value around zero. More precisely, the log10 of the Euclidian norms of the (y_1, y_2) components of the solutions corresponding

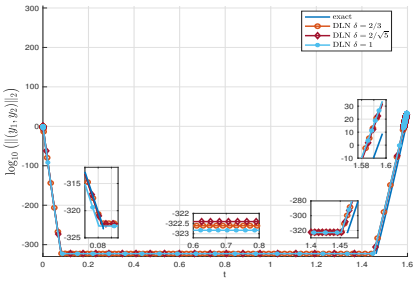
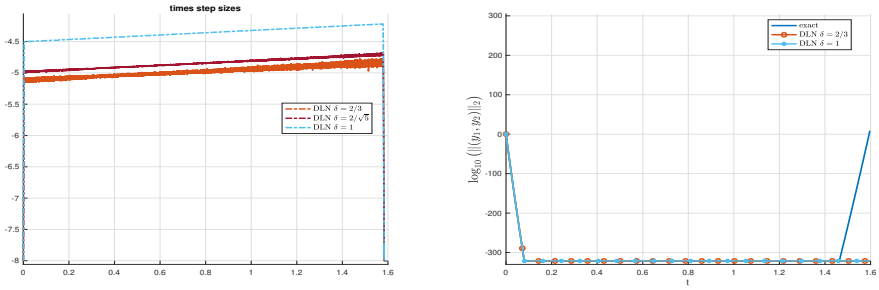


Fig. 4.9: The base 10 logarithmic plot of the Euclidean norm of the (y_1, y_2) components of the solution to the Lindberg example 4.2.3. The solutions obtained by the DLN methods ($\delta = 2/3, 2/\sqrt{5}, 1$, implemented with Algorithm 1, using Milne's device and (AB2-like)) with absolute tolerances around Tol =1.0e-15 (see Table 4.7) and minimum step size 1.e-8 correctly approximate the exact solution.

to the DLN methods with $\delta = 2/3$ and $\delta = 1$ are 5.6818e-322, 6.7687e-322, respectively.



(a) (b)

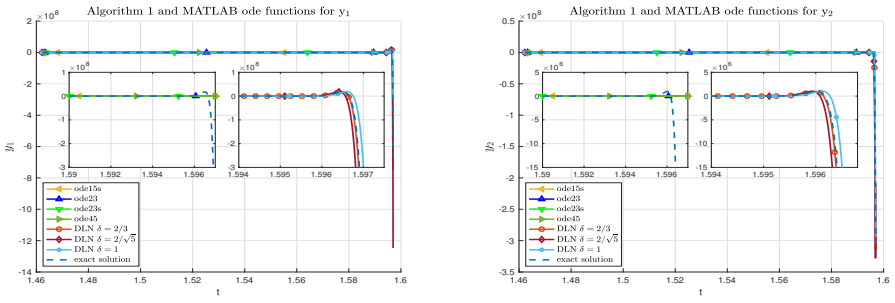
Fig. 4.10: (a) The `log10` of the time step sizes for the adaptive DLN methods, for the Lindberg example, via Algorithm 1, with minimal step size 1.e-08. (b) The `log10` of the Euclidean norm of the constant step solutions for $\delta = 2/3, \Delta t \approx 9.5480\text{e}-07$, $\delta = 1, \Delta t \approx 1.0805\text{e}-06$. The constant step solutions do not observe the instability at the end of the time interval, unlike their adaptive counterparts in Figure 4.9.

(2) In order to focus on the ability of the estimators of Matlab's `ode` functions to detect the change in the stability regime, we shall next simulate the Lindberg example on the time interval $[1.4622, 1.597]$. We will test Algorithm 1 on the time interval $[1.4622, 1.597]$, while Algorithm 3 we use $t \in [1.4633, 1.597]$. We take both the minimum step size and the initial time step 1.e-8. The tolerances of the two adaptive DLN algorithms are given in Table 4.7.
The graphs of first and second components by Algorithm 1 are given in Figure 4.11. From Figure 4.11(a) and 4.11(b), the Algorithm 1 with $\delta = 2/3, 2/\sqrt{5}, 1$ simulate relative well while all MATLAB ode functions fail even under small tolerance. From Figure 4.12(a) and 4.12(b), we can see that the simulations by Algo-

	$\delta = 2/3$	$\delta = 2/\sqrt{5}$	$\delta = 1$
Algorithm 1	$0.635 * (1.e-14)$	$5.268 * (1.e-15)$	$4.627 * (1.e-15)$
Algorithm 3	$1.e-14$	$1.e-14$	-

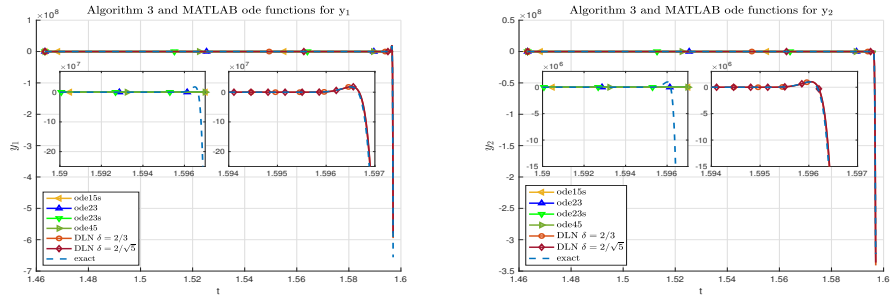
Table 4.7: The absolute tolerances used for the DLN Algorithms, in the Lindberg example, on the smaller time interval $t \in [1.4622, 1.597]$, and with minimal step size 10^{-8}.

rithm 2 works much better but with more steps and later initial time. The obvious difference between the adaptive DLN and MATLAB ode functions can be seen by $\log_{10}(\|(y_1, y_2)\|)$ in Figure 4.13(a) and 4.13(b).



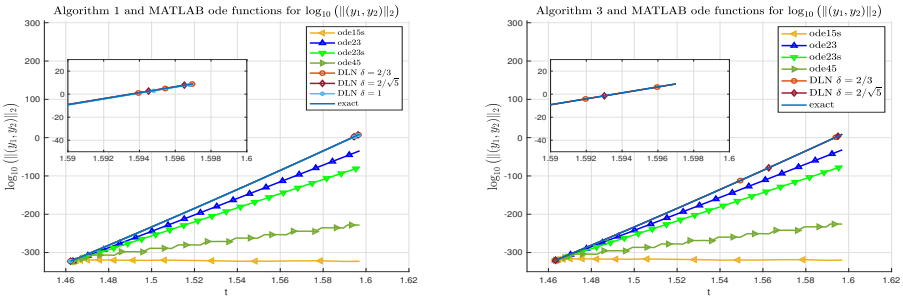
(a) (b)

Fig. 4.11: The first (4.11(a)) and second (4.11(b)) components (y_1, y_2) of the solution for the Lindberg example, by Algorithm 1, and several MATLAB `ode` functions, on the time interval $t \in [1.4622, 1.597]$. All MATLAB `ode` functions fail at $t \approx 1.596$ (the solutions remain relatively small). The solutions given by the adaptive DLN Algorithm 1 are accurate, changing from the stable to the unstable regime.



(a) (b)

Fig. 4.12: The first (4.12(a)) and second (4.12(b)) components (y_1, y_2) of the solution for the Lindberg example, by Algorithm 3, and several MATLAB `ode` functions, on the time interval $t \in [1.4633, 1.597]$. All MATLAB `ode` functions fail at time $t \approx 1.596$ (the solutions remain relatively small). The solutions given by the adaptive DLN Algorithm 3 are accurate, changing from the stable to the unstable regime.



(a) (b)

Fig. 4.13: The $\log_{10}$ of the Euclidean norm of the (y_1, y_2) components of the solution to the Lindberg example, by Algorithm 1, Algorithm 3, and several MATLAB `ode` functions, on the time interval $t \in [1.4633, 1.597]$. The solutions given by the MATLAB `ode` functions remain relatively small. The solutions given by the adaptive DLN Algorithm 1 and Algorithm 3 approximate well the exact solution, and change from the stable to the unstable regime.

5. Conclusions. We designed and analyzed two time-adaptive algorithms for the DLN method. The first algorithm is based on Milne's device, and it uses the differentiation defect [58] and the second-order (AB2-like) explicit to estimate the local truncation error. The second algorithm estimates the DLN's local truncation error by the difference between the first- and second-order accurate DLN solutions, which are provided by the DLN refactorization process in [58]. We found adaptivity based on Milne's device is efficient and reliable. The numerical tests in Section 4.1 employ the constant step DLN method, with three values of the free parameter $\delta = 2/3$, $\delta = 2/\sqrt{5}$ and $\delta = 1$ (the midpoint method), to verify the rates of convergence, and to compare the stability properties of the DLN method to other popular second-order methods. The tests confirmed the predicted rates of convergence and stability. The examples in Section 4.2 use the Lotka-Volterra and the Kepler two-body problem to compare the DLN adaptive algorithms' conservation properties with the constant step DLN methods, and with some of the Matlab's adaptive algorithms. The results from the Van der Pol (with $\mu = 1000$) example, a stiff problem, show that *the adaptive DLN methods are more accurate than the corresponding constant step methods, when using the same number of time steps, and adaptivity controls both amplitude (Hamiltonian) and phase error.*

The Lindberg example is a challenging numerical test, even among stiff problems, for two reasons. The first reason is that the exact solution decays fast to very small values, which due to underflow (in double precision) would make the solution zero, and stay zero from then onward. The second and paramount reason is that the exact solution, later on, has a swift behavioral change, from the stable to an unstable regime. *The adaptive Matlab functions*: `ode45`, and the stiff `ode15s`, `ode23s` solvers, all *fail to approximate the exact solution, even when using extreme values for the absolute and relative errors. The adaptive DLN methods*, with all three values of the free parameter δ, *approximate reasonably well the exact solution*. The approximate DLN solutions decay to small values, right above the underflow, and then sense the change from the stable to the unstable regime.

The *constant time step DLN methods work well for problems with smooth solutions.* In particular, the symplectic midpoint method has the smallest errors, also conserving all quadratic Hamiltonians. *For stiff problems, the adaptive DLN methods outperform the constant time step algorithms*, with minimal computational costs. In our experience, the most successful *step size estimator* was the one based on the Milne's device, using the AB2-like approximate solution to estimate the local truncation error.

Acknowledgement. The authors thank Dr. John Burkardt (University of Pittsburgh) for very helpful discussions.

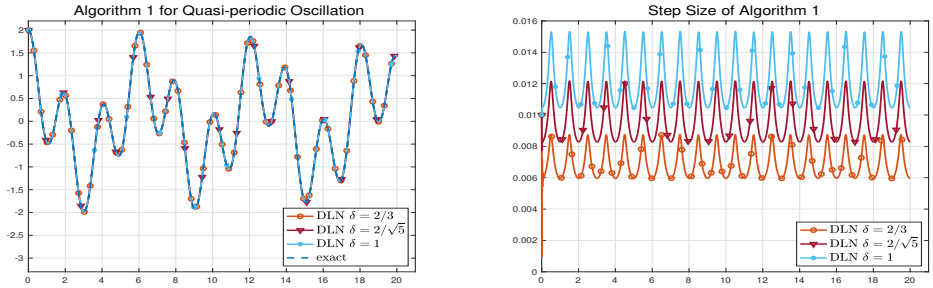
Declarations.

- Funding: The work of WL and WP was partially supported by the NSF under grant DMS 2110379. The work of CT was partially supported by the NSF under grant DMS 2208219.
- Conflict of interest: The authors declare no competing interests.
- Ethics approval: Not applicable
- Consent to participate: Not applicable
- Consent for publication: Not applicable
- Availability of data and materials: Data sharing is not applicable to this article as no data sets were generated or analyzed during the current study.
- Code availability: upon request
- Authors' contributions: The authors contributed equally.

Appendices

A. Quasi-periodic oscillations. In here we test the adaptive DLN Algorithm 1 (based on the AB2 estimator) and Algorithm 3 (based on refactorization), on the quasi-periodic oscillations problem from Section 4.1.1. We use again the time interval $t \in [0,20]$, and set the initial time step $k_1 = 1.e-2$, and the tolerance $\text{Tol} = 1.e-4$. The Figures A.1(a) and A.1(b) display the solutions corresponding to the DLN methods with Algorithm 1. For comparable relative $\|\cdot\|_{2,\infty}$ and $\|\cdot\|_{2,2}$ errors, shown in Table A.1, the number of time steps is rapidly decreasing as the value of δ increases: $2,948$ ($\delta = 2/3$), $2,118$ ($\delta = 2/\sqrt{5}$), and $1,678$ ($\delta = 1$), respectively. We recall that a similar behavior was reported in Section 4.1.1, where for the same constant time step, the $\|\cdot\|_{2,\infty}$ and $\|\cdot\|_{2,2}$ errors were decreasing, as the value of δ was increased. The solutions obtained by the adaptive DLN Algorithm 3 also approximate well the exact solution (Table A.2), but the number of time steps are considerably larger: $24,880$ ($\delta = 2/3$), and $25,649$ ($\delta = 2/\sqrt{5}$).

We also compare the errors of the adaptive DLN algorithms versus the constant DLN methods (with the same number of steps for the respective values of δ) in Table A.1 and Table A.2. Using both the $\|\cdot\|_{2,2}$ and $\|\cdot\|_{2,\infty}$ error norms, the constant time step DLN method perform slightly better than the adaptive DLN algorithms, under the same number of time steps. This is an expected observation, since the quasi-periodic oscillations problem is non-stiff and smooth. However, as seen in stiff test problems (Van der Pol's equation 4.2.2 and Lindberg's example 4.2.3), the adaptive DLN algorithms is more advantageous than the constant step DLN algorithms.



(a) (b)

Fig. A.1: The DLN solutions of the quasi-periodic oscillations Section A, using the adaptive Algorithm 1 with Milne's device and (AB2-like) for the estimation of LTE (left). The evolution of the time step sizes is displayed on the right. For comparable relative errors (Table A.1), the DLN ($\delta = 2/3$) method uses almost twice the number of time steps employed by DLN ($\delta = 1$).

	$\delta=\frac{2}{3}$	$\delta=\frac{2}{\sqrt{5}}$	$\delta=1$	($\delta=\frac{2}{3}$)	($\delta=\frac{2}{\sqrt{5}}$)	($\delta=1$)
# time steps	2,948	2,118	1,678			
time step Δt				0.0068	0.0094	0.0119
$|e|_{2,\infty}$1.e+3	6.3813	7.4051	7.3755	6.0617	7.0454	7.0163
$|e|_{2,2}$1.e+3	12.1539	14.1052	14.0467	11.5443	13.4188	13.3642

Table A.1: The errors corresponding to the quasi-periodic oscillations in Example A, on the time interval $[0,20]$. We compare the adaptive DLN Algorithm 1 (using Milne's device and (AB2-like)) versus the DLN methods with constant time step, and the same number of time steps as the adaptive algorithms, respectively (the last three columns). The second line indicates the number of time steps used in the adaptive algorithms, while the third line displays the sizes of the corresponding time steps used for the constant case methods. It was shown in Section 4.1.1 that when all DLN methods ($\delta=2/3, 2/\sqrt{5}, 1$) are using the same constant time step, the errors are smaller for the larger values of δ (see Tables 4.1,4.2).

	$\delta=\frac{2}{3}$	$\delta=\frac{2}{\sqrt{5}}$	Const ($\delta=\frac{2}{3}$)	Const ($\delta=\frac{2}{\sqrt{5}}$)
# time steps	24,880	25,649		
time step Δt			8.0386e$-$04	7.7976e$-$04
$\|e\|_{2,\infty}$	0.00038190	0.00061367	0.00008513	0.00004806
$\|e\|_{2,2}$	0.00072598	0.00116607	0.00016212	0.00009153

Table A.2: The errors corresponding to the quasi-periodic oscillations in Example A, on the time interval $[0,20]$. We compare the adaptive DLN Algorithm 3 (using refactorization) versus the DLN methods with constant time step, and the same number of time steps as the adaptive algorithms, respectively (the last two columns). The second line indicates the number of time steps used in the adaptive algorithms, while the third line displays the size of the corresponding time step used for the constant case methods.

B. The Kepler Problem. Consider the two-body Kepler problem [44,51]

$$\begin{cases} q_1' = p_1, \\ q_2' = p_2, \\ p_1' = -\frac{q_1}{\sqrt{(q_1^2+q_2^2)^3}}, \\ p_2' = -\frac{q_2}{\sqrt{(q_1^2+q_2^2)^3}}, \end{cases} \qquad \begin{bmatrix} q_1(0) \\ q_2(0) \\ p_1(0) \\ p_2(0) \end{bmatrix} = \begin{bmatrix} 1-0.6 \\ 0 \\ 0 \\ \sqrt{\frac{1+0.6}{1-0.6}} \end{bmatrix},$$

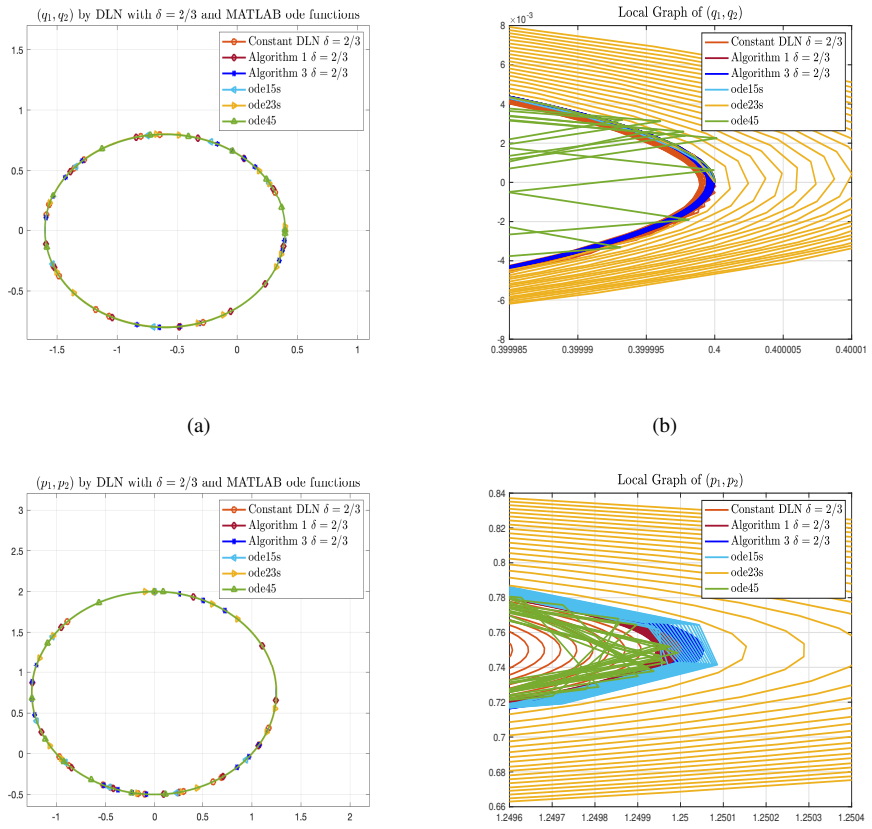
with Hamiltonian

$$H(q_1,q_2,p_1,p_2) = \frac{1}{2}\left(p_1^2+p_2^2\right) - \left(q_1^2+q_2^2\right)^{-1/2}.$$

We evaluate the behavior of the adaptive DLN Algorithm 1, Algorithm 3, versus the constant time step DLN methods, and some of the MATLAB `ode` functions, on the Kepler problem, over the time interval $[0,120]$. We set the tolerance Tol=1.e$-$8 in Algorithm 1, Tol=1.e$-$6 in Algorithm 3, while for the MATLAB `ode` functions, we set the relative tolerance RelTol=1.e$-$6 and the absolute tolerance AbsTol=1.e$-$8. The initial step size is 1.e$-$4, and the number of time steps for the constant step DLN algorithm is 10^5.
In Figure B.1 we show the phase planes for the q_1,q_2 and p_1,p_2 solutions, obtained by the DLN algorithms corresponding to $\delta=2/3$, and the MATLAB `ode15s`, `ode23s` and `ode45` functions. The zoomed in plots in Figure B.1(b) and Figure B.1(d), show that *the adaptive*

DLN methods (Algorithm 1 and Algorithm 3) perform better than the constant step DLN method, and also compared with the MATLAB `ode15s`, `ode23s` and `ode45` functions. The same conclusion holds for DLN ($\delta = 2/\sqrt{5}$) and DLN ($\delta = 1$).
In Figure B.2 we explore the conservation of the Hamiltonian by the solutions of the DLN methods $\delta = 2/3, 2/\sqrt{5}, 1$ by Algorithms 1 and 3 versus MATLAB `ode15s`, `ode23s` and `ode45` functions. The Table B.1 reports the number of time steps.

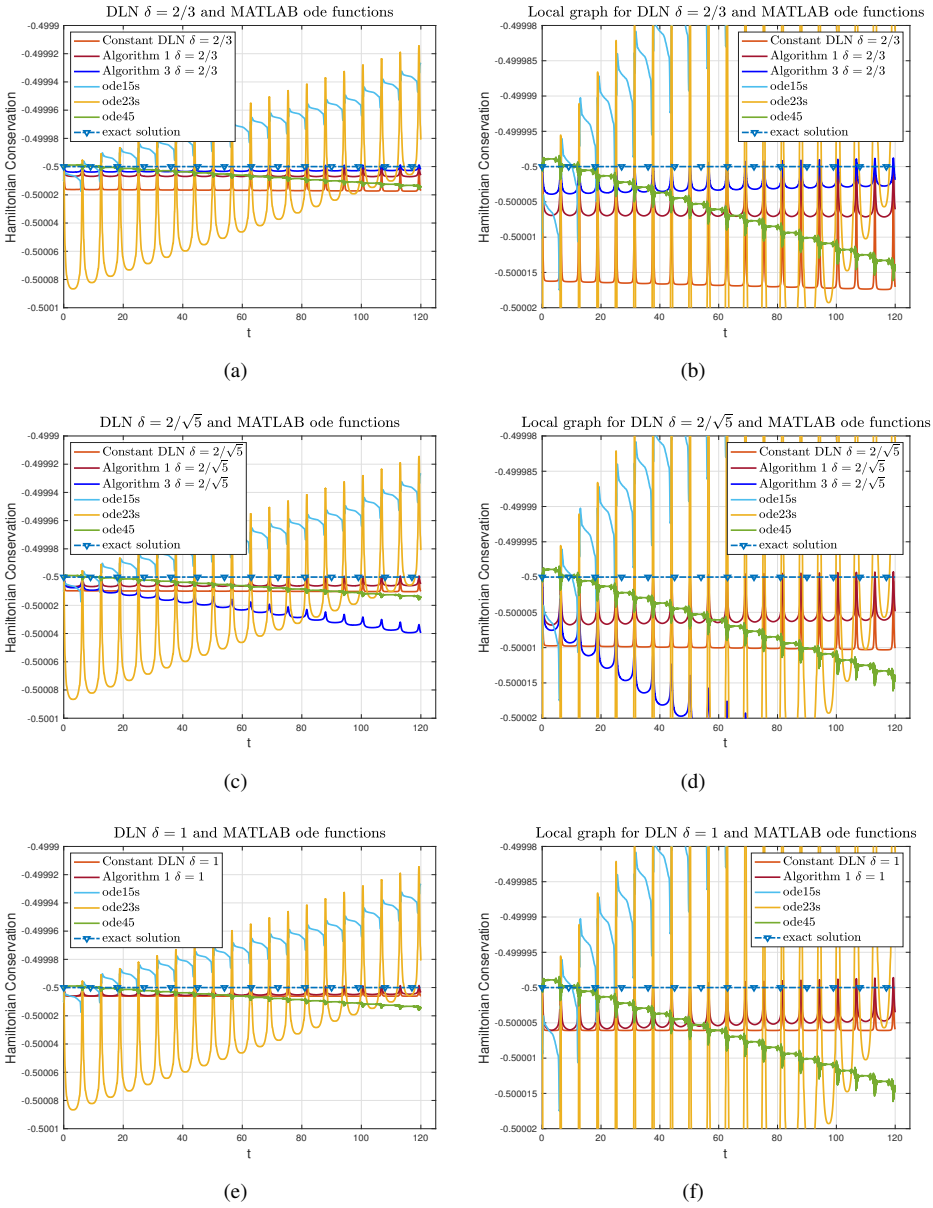
From the information in Figure B.2 and Table B.1, we conclude that *Algorithm 1 performs better than Algorithm 3, the constant time step DLN methods, and the Matlab* `ode15s`, `ode23s`, `ode45` *functions*. The solutions obtained by Algorithm 1 are more accurate than solutions of the constant step DLN algorithms, with fewer number of time steps. Within the used tolerances, the Hamiltonian corresponding to the adaptive DLN algorithms is better conserved than the Hamiltonian corresponding to the MATLAB `ode` functions, but the DLN methods use relatively more time steps.



(a) (b)

(c) (d)

Fig. B.1: The phase plane corresponding to the q_1, q_2 and p_1, p_2 solutions of the Kepler problem B. The plots show the DLN Algorithm 1 and 3 with $\delta = 2/3$, and the MATLAB `ode15s`, `ode23s` and `ode45` functions.



(a) (b)

(c) (d)

(e) (f)

Fig. B.2: The Hamiltonian of the Kepler problem B. The adaptive DLN Algorithm 1 has better conservation properties.

	Constant step DLN	Algorithm 1	Algorithm 3
$\delta = 2/3$	100,000	62,337	154,817
$\delta = 2/\sqrt{5}$	100,000	47,202	157,626
$\delta = 1$	100,000	38,775	-
	ode15s	ode23s	ode45
	3,374	12,235	3,733

Table B.1: The number of time steps used by the DLN algorithms and the MATLAB `ode` functions for the Kepler problem B.

[1] U. M. ASCHER AND L. R. PETZOLD, Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1st ed., 1998.

[2] J. V. D. M. B. VAN DER POL, Frequency demultiplication, Philos. Mag., 120 (1927), p. 363–364.

[3] S. BADIA, F. NOBILE, AND C. VERGARA, Robin–Robin preconditioned Krylov methods for fluid–structure interaction problems, Computer Methods in Applied Mechanics and Engineering, 198 (2009), pp. 2768–2784.

[4] L. C. BERSELLI, S. FAGIOLI, AND S. SPIRITO, Suitable weak solutions of the Navier-Stokes equations constructed by a space-time numerical discretization, J. Math. Pures Appl. (9), 125 (2019), pp. 189–208.

[5] L. BERTAGNA, A. QUAINI, AND A. VENEZIANI, Deconvolution-based nonlinear filtering for incompressible flows at moderately large Reynolds numbers, Internat. J. Numer. Methods Fluids, 81 (2016), pp. 463–488.

[6] M. BUKAČ, G. FU, A. SEBOLDT, AND C. TRENCHEA, Time-adaptive partitioned method for fluid-structure interaction problems with thick structures, J. Comput. Phys., 473 (2023), p. Paper No. 111708.

[7] M. BUKAČ, A. SEBOLDT, AND C. TRENCHEA, Refactorization of Cauchy's Method: A Second-Order Partitioned Method for Fluid-Thick Structure Interaction Problems, J. Math. Fluid Mech., 23 (2021), p. 64.

[8] M. BUKAČ AND C. TRENCHEA, Adaptive, second-order, unconditionally stable partitioned method for fluid-structure interaction, Comput. Methods Appl. Mech. Engrg., 393 (2022), pp. Paper No. 114847, 24.

[9] J. BURKARDT AND C. TRENCHEA, Refactorization of the midpoint rule, Appl. Math. Lett., 107 (2020), p. 106438.

[10] E. BURMAN AND M. A. FERNÁNDEZ, Stabilization of explicit coupling in fluid–structure interaction involving fluid incompressibility, Computer Methods in Applied Mechanics and Engineering, 198 (2009), pp. 766–784.

[11] J. C. BUTCHER, Numerical methods for ordinary differential equations, John Wiley & Sons, Ltd., Chichester, third ed., 2016. With a foreword by J. M. Sanz-Serna.

[12] M. CALVO, T. GRANDE, AND R. D. GRIGORIEFF, On the zero stability of the variable order variable stepsize BDF-formulas, Numer. Math., 57 (1990), pp. 39–50.

[13] M. CALVO, J. I. MONTIJANO, AND L. RÁNDEZ, A_0-stability of variable stepsize BDF methods, J. Comput. Appl. Math., 45 (1993), pp. 29–39.

[14] F. CAPUANO, B. SANDERSE, E. DE ANGELIS, AND G. COPPOLA, A minimum-dissipation time-integration strategy for large-eddy simulation of incompressible turbulent flows, in AIMETA 2017 Proceedings of the XXIII Conference of the Italian Association of Theoretical and Applied Mechanics, sep 2017, pp. 2311–2323.

[15] M. L. CARTWRIGHT, Balthazar van der Pol, J. London Math. Soc., 35 (1960), pp. 367–376.

[16] S. CAUCAO, R. OYARZÚA, S. VILLA-FUENTES, AND I. YOTOV, A three-field Banach spaces-based mixed formulation for the unsteady Brinkman-Forchheimer equations, Comput. Methods Appl. Mech. Engrg., 394 (2022), pp. Paper No. 114895, 32.

[17] K. CHENG, C. WANG, AND S. M. WISE, An energy stable BDF2 Fourier pseudo-spectral numerical scheme for the square phase field crystal equation, Commun. Comput. Phys., 26 (2019), pp. 1335–1364.

[18] J. CRANK AND P. NICOLSON, A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type, Proc. Cambridge Philos. Soc., 43 (1947), pp. 50–67.

[19] M. CROUZEIX AND F. J. LISBONA, The convergence of variable-stepsize, variable-formula, multistep methods, SIAM J. Numer. Anal., 21 (1984), pp. 512–534.

[20] G. G. DAHLQUIST, On the relation of G-stablity to other stability concepts for linear multistep methods, tech. rep., CM-P00069426, 1976.

[21] ———, G-stability is equivalent to A-stability, BIT Numerical Mathematics, 18 (1978), pp. 384–401.

[22] ———, Positive functions and some applications to stability questions for numerical methods. recent advances in numerical analysis, in Proc. Symp., Madison/Wis, 1978.
[23] G. G. DAHLQUIST, On one-leg multistep methods, SIAM J. Numer. Anal., 20 (1983), pp. 1130–1138.
[24] G. G. DAHLQUIST AND Å. BJÖRCK, Numerical methods in scientific computing. Vol. I, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008.
[25] G. G. DAHLQUIST, W. LINIGER, AND O. NEVANLINNA, Stability of two-step methods for variable integration steps, SIAM J. Numer. Anal., 20 (1983), pp. 1071–1085.
[26] P. A. DE SAMPAIO, P. H. HALLAK, A. L. COUTINHO, AND M. S. PFEIL, A stabilized finite element procedure for turbulent fluid–structure interaction using adaptive time–space refinement, International Journal for Numerical Methods in Fluids, 44 (2004), pp. 673–693.
[27] B. L. EHLE, On Padé approximations to the exponential function and $A-$stable methods for the numerical solution of initial value problems, tech. rep., CSRR 2010, Dept. AACS, Univ. of Waterloo, Ontario, Canada, 1969.
[28] E. EMMRICH, Error of the two-step BDF for the incompressible Navier-Stokes problem, M2AN Math. Model. Numer. Anal., 38 (2004), pp. 757–764.
[29] ———, Stability and convergence of the two-step BDF for the incompressible Navier-Stokes problem, Int. J. Nonlinear Sci. Numer. Simul., 5 (2004), pp. 199–209.
[30] ———, Stability and error of the variable two-step BDF for semilinear parabolic problems, J. Appl. Math. Comput., 19 (2005), pp. 33–55.
[31] ———, Convergence of the variable two-step BDF time discretisation of nonlinear evolution problems governed by a monotone potential operator, BIT, 49 (2009), pp. 297–323.
[32] C. W. GEAR, Numerical initial value problems in ordinary differential equations, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1971.
[33] C. W. GEAR AND K. W. TU, The effect of variable mesh size on the stability of multistep methods, SIAM J. Numer. Anal., 11 (1974), pp. 1025–1043.
[34] V. GIRAULT AND P.-A. RAVIART, Finite element approximation of the Navier-Stokes equations, vol. 749 of Lecture Notes in Mathematics, Springer-Verlag, Berlin, 1979.
[35] P. M. GRESHO, D. F. GRIFFITHS, AND D. J. SILVESTER, Adaptive time-stepping for incompressible flow. I. Scalar advection-diffusion, SIAM J. Sci. Comput., 30 (2008), pp. 2018–2054.
[36] P. M. GRESHO, R. L. LEE, S. T. CHAN, AND R. L. SANI, Solution of the time-dependent incompressible Navier-Stokes and Boussinesq equations using the Galerkin finite element method, in Approximation methods for Navier-Stokes problems (Proc. Sympos., Univ. Paderborn, Paderborn, 1979), vol. 771 of Lecture Notes in Math., Springer, Berlin, 1980, pp. 203–222.
[37] P. M. GRESHO, R. L. LEE, R. L. SANI, AND T. STULLICH, Time-dependent FEM solution of the incompressible Navier–Stokes equations in two-and three-dimensions, tech. rep., California Univ., 1978.
[38] P. M. GRESHO AND R. L. SANI, Incompressible flow and the finite element method, Volume 2: Isothermal Laminar Flow, Incompressible Flow & the Finite Element Method, Wiley, 2000.
[39] D. F. GRIFFITHS AND D. J. HIGHAM, Numerical methods for ordinary differential equations, Springer Undergraduate Mathematics Series, Springer-Verlag London, Ltd., London, 2010. Initial value problems.
[40] R. D. GRIGORIEFF, Time discretization of semigroups by the variable two-step BDF method, in Numerical treatment of differential equations (Halle, 1989), vol. 121 of Teubner-Texte Math., Teubner, Stuttgart, 1991, pp. 204–216.
[41] R. D. GRIGORIEFF AND P. J. PAES-LEME, On the zero-stability of the 3-step BDF-formula on nonuniform grids, BIT, 24 (1984), pp. 85–91.
[42] M. D. GUNZBURGER, Finite element methods for viscous incompressible flows, Computer Science and Scientific Computing, Academic Press Inc., Boston, MA, 1989. A guide to theory, practice, and algorithms.
[43] A. GUZEL AND W. LAYTON, Time filters increase accuracy of the fully implicit method, BIT, 58 (2018), pp. 301–315.
[44] E. HAIRER, C. LUBICH, AND G. WANNER, Geometric numerical integration, vol. 31 of Springer Series in Computational Mathematics, Springer, Heidelberg, 2010. Structure-preserving algorithms for ordinary differential equations, Reprint of the second (2006) edition.
[45] E. HAIRER, S. P. NØRSETT, AND G. WANNER, Solving ordinary differential equations. I, vol. 8 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin, second ed., 1993. Nonstiff problems.
[46] E. HAIRER AND G. WANNER, Solving ordinary differential equations. II, vol. 14 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin, 2010. Stiff and differential-algebraic problems, Second revised edition.
[47] P. HENRICI, Discrete variable methods in ordinary differential equations, John Wiley & Sons, Inc., New York-London, 1962.
[48] J. G. HEYWOOD AND R. RANNACHER, Finite-element approximation of the nonstationary Navier-Stokes problem. IV. Error analysis for second-order time discretization, SIAM J. Numer. Anal., 27 (1990), pp. 353–384.
[49] V. JOHN, Finite element methods for incompressible flow problems, vol. 51 of Springer Series in Computa-

tional Mathematics, Springer, Cham, 2016.
[50] D. A. KAY, P. M. GRESHO, D. F. GRIFFITHS, AND D. J. SILVESTER, Adaptive time-stepping for incompressible flow. II. Navier-Stokes equations, SIAM J. Sci. Comput., 32 (2010), pp. 111–128.
[51] J. KEPLER, Astronomia nova, seu Physica celestis, traditia commentariis de motibus stellae Martis, ex observationibus G. V. Tychonis Brahe, Prague, 1609.
[52] G. Y. KULIKOV AND S. K. SHINDIN, On multistep interpolation-type methods with automatic control of global error, Comput. Math. Math. Phys., 44 (2004), pp. 1314–1333.
[53] G. Y. KULIKOV AND S. K. SHINDIN, On stable integration of stiff ordinary differential equations with global error control, in Computational Science – ICCS 2005, V. S. Sunderam, G. D. van Albada, P. M. A. Sloot, and J. J. Dongarra, eds., Berlin, Heidelberg, 2005, Springer Berlin Heidelberg, pp. 42–49.
[54] G. Y. KULIKOV AND S. K. SHINDIN, One-leg integration of ordinary differential equations with global error control, Comput. Methods Appl. Math., 5 (2005), pp. 86–96.
[55] ———, One-leg variable-coefficient formulas for ordinary differential equations and local-global step size control, Numer. Algorithms, 43 (2006), pp. 99–121.
[56] J. D. LAMBERT, Numerical methods for ordinary differential systems, John Wiley & Sons, Ltd., Chichester, 1991. The initial value problem.
[57] W. LAYTON, W. PEI, Y. QIN, AND C. TRENCHEA, Analysis of the variable step method of Dahlquist, Liniger and Nevanlinna for fluid flow, Numer. Methods Partial Differential Equations, 38 (2022), pp. 1713–1737.
[58] W. LAYTON, W. PEI, AND C. TRENCHEA, Refactorization of a variable step, unconditionally stable method of Dahlquist, Liniger and Nevanlinna, Appl. Math. Lett., 125 (2022), p. Paper No. 107789.
[59] W. LAYTON, L. G. REBHOLZ, AND C. TRENCHEA, Modular nonlinear filter stabilization of methods for higher Reynolds numbers flow, J. Math. Fluid Mech., 14 (2012), pp. 325–354.
[60] Y. LI AND C. TRENCHEA, A higher-order Robert–Asselin type time filter, J. Comput. Phys., 259 (2014), pp. 23–32.
[61] B. LINDBERG, On a dangerous property of methods for stiff differential equations, Nordisk Tidskr. Informationsbehandling (BIT), 14 (1974), pp. 430–436.
[62] A. J. LOTKA, Undamped oscillations derived from the law of mass action., Journal of the American Chemical Society, 42 (1920), pp. 1595–1599.
[63] ———, Elements of physical biology, Williams & Wilkins company, Baltimore, 1925.
[64] M. MAYR, W. WALL, AND M. GEE, Adaptive time stepping for fluid-structure interaction solvers, Finite Elements in Analysis and Design, 141 (2018), pp. 55–69.
[65] W. E. MILNE, Numerical Integration of Ordinary Differential Equations, Amer. Math. Monthly, 33 (1926), pp. 455–460.
[66] J. D. MURRAY, Mathematical biology. I, vol. 17 of Interdisciplinary Applied Mathematics, Springer-Verlag, New York, third ed., 2002. An introduction.
[67] O. NEVANLINNA AND W. LINIGER, Contractive methods for stiff differential equations. II, BIT, 19 (1979), pp. 53–72.
[68] O. OYEKOLE, C. TRENCHEA, AND M. BUKAČ, A Second-Order in Time Approximation of Fluid-Structure Interaction Problem, SIAM J. Numer. Anal., 56 (2018), pp. 590–613.
[69] J.-H. PARK, A. J. SALGADO, AND S. M. WISE, Benchmark computations of the phase field crystal and functionalized Cahn-Hilliard equations via fully implicit, Nesterov accelerated schemes, https://arxiv.org/abs/2204.07247, (2022).
[70] Y. QIN, Y. HOU, W. PEI, AND J. LI, A variable time-stepping algorithm for the unsteady Stokes/Darcy model, Journal of Computational and Applied Mathematics, (2021), p. 113521.
[71] A. SEBOLDT AND M. BUKAČ, A non-iterative domain decomposition method for the interaction between a fluid and a thick structure, Numer. Methods Partial Differential Equations, 37 (2021), pp. 2803–2832.
[72] L. F. SHAMPINE AND M. K. GORDON, Computer solution of ordinary differential equations, W. H. Freeman and Co., San Francisco, Calif., 1975. The initial value problem.
[73] R. D. SKEEL AND A. K. KONG, Blended linear multistep methods, ACM Trans. Math. Software, 3 (1977), pp. 326–345.
[74] G. SÖDERLIND, Digital filters in adaptive time-stepping, ACM Transactions on Mathematical Software (TOMS), 29 (2003), pp. 1–26.
[75] G. SÖDERLIND, I. FEKETE, AND I. FARAGÓ, On the zero-stability of multistep methods on smooth nonuniform grids, BIT, 58 (2018), pp. 1125–1143.
[76] G. SÖDERLIND AND L. WANG, Adaptive time-stepping and computational stability, J. Comput. Appl. Math., 185 (2006), pp. 225–243.
[77] H. J. STETTER, Analysis of discretization methods for ordinary differential equations, Springer-Verlag, New York-Heidelberg, 1973. Springer Tracts in Natural Philosophy, Vol. 23.
[78] A. TAKHIROV AND C. TRENCHEA, Efficient nonlinear filter stabilization of the Leray-α model, J. Comput. Phys., 471 (2022), p. Paper No. 111668.
[79] A. TAKHIROV, C. TRENCHEA, AND J. WATERS, Second-order efficient nonlinear filter stabilization for high Reynolds number flows, Numer. Methods Partial Differential Equations, 39 (2023), pp. 90–107.

[80] B. VAN DER POL, A theory of the amplitude of free and forced triode vibrations, 1 (1920) 701–710., Radiol. Rev., 1 (1920), pp. 701–710.

[81] V. VOLTERRA, Variazionie fluttuazioni del numero d'individui in specie animali conviventi, Mem. Acad. Lincei., 2 (1926), p. 31–113. Variations and fluctuations of a number of individuals in animal species living together. Translation by R.N. Chapman. In: Animal Ecology. pp. 409-448. McGraw Hill, New York, 1931.

[82] D. S. WATANABE AND Q. M. SHEIKH, One-leg formulas for stiff ordinary differential equations, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 489–496.

[83] H. XU, D. BAROLI, F. DI MASSIMO, A. QUAINI, AND A. VENEZIANI, Backflow stabilization by deconvolution-based large eddy simulation modeling, J. Comput. Phys., 404 (2020), pp. 109103, 21.