Generalizing Global Error Estimation for Ordinary Differential Equations by Using Coupled Time-Stepping Methods

Emil M. Constantinescu

Abstract

This study introduces new time-stepping strategies with built-in global error estimators. The new methods propagate the defect along with the numerical solution much like solving for the correction or Zadunaisky's procedure; however, the proposed approach allows for overlapped internal computations and, therefore, represents a generalization of the classical numerical schemes for solving differential equations with global error estimation. The resulting algorithms can be effectively represented as general linear methods. Several explicit self-starting schemes akin to Runge-Kutta methods with global error estimation are introduced, and the theoretical considerations are illustrated on several examples.

Keywords: Ordinary differential equations, Time integration, Local and global error estimation

1. Introduction

The global error or a posteriori error represents the actual numerical error resulting after applying a time-stepping algorithm. Calculating this error and controlling it by adapting the step size are generally viewed as expensive processes, and therefore in practice only local error or the error from one step to the next is used for step size control or as a proxy for error estimation [1, 2, 3, 4]. In general, however, local error estimates cannot predict how those local errors will propagate through the simulation, and for some problems these local errors can grow to be larger than intended. Therefore, from the end-user perspective, local error estimation (LEE) is not always suitable, especially for problems with unstable modes or long integration times [5, 6] because the solution may end up having unexpectedly large numerical errors. This aspect prompts us to revisit global error estimation (GEE) in order to make it more transparent and practical, ultimately leading to better error control and reliable accuracy.

We consider the first-order system of ordinary differential equations

\[ y(t)' = f(y(t)) \quad ; \quad y(t_0) = y_0, \quad t_0 < t \leq T, \quad y \in \mathbb{R}^m, \quad f : \mathbb{R}^m \to \mathbb{R}^m, \]

(1.1)
of size \( m \) with \( y_0 \) given. In this case, the global error at every time step \( n \) is defined by

\[ \varepsilon(t_n) = y(t_n) - y_n, \quad n = 1, 2, \ldots, N_T, \]

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research Program under contract DE-AC02-06CH11357, FWP #56706 and #57K87.

Email address: emconsta@mcs.anl.gov (Emil M. Constantinescu)

1Mathematics and Computer Science Division, Argonne National Laboratory, 9700 S. Cass Avenue, Lemont, IL 60439.

Preprint submitted to Journal of Computational and Applied Mathematics May 9, 2017
that is, the difference between the exact solution \( y(t_n) \) at time step \( n \) from a sequence of \( N_T \) steps and a numerical approximation \( y_n \). A priori and a posteriori error bounds under appropriate smoothness assumptions are well known \([7, 6]\). Here we focus on a posteriori estimates of \( \varepsilon(t_n) \).

One of the most comprehensive surveys for global error estimation is by Skeel \([8]\). Global error estimation in time stepping has a long history \([9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]\). A posteriori global error estimation has been recently discussed in \([21, 22, 23, 24]\). Step-size control with multimethod Runge-Kutta (RK) is analyzed in \([25, 26, 13, 27]\). Global error estimation for stiff problems is discussed in \([28, 29, 30, 31]\). Adjoint methods for global error estimation for PDEs \([32, 33]\) are analyzed in \([34, 35]\). These studies cover most of the types of strategy that have been proposed to address global error estimation. The Zadunaisky procedure \([36]\) and the related procedure for solving for the correction \([8]\) have arguably been the most popular global-error estimation strategies \([10, 9]\). The work of Dormand et al. \([37, 38]\) relies on this procedure and is extended to a composition of RK methods in \([39]\). Further extensions are introduced by Murua and Makazaga \([40, 15]\). Shampine \([13]\) proposes using multiple methods to estimate global errors.

We introduce and analyze efficient strategies for estimating global errors for time-stepping algorithms. We present a general approach that includes most of the classical strategies as particular cases, and we develop new algorithms that fall under general linear time-stepping schemes. Therefore, they generalize many of the existing strategies. Our work builds on similar ideas introduced by Shampine \([13]\) and Zadunaisky \([36]\) and the followups in the sense that the strategy evolves the defect along with the solution; however, in our strategy the internal calculations of the two quantities can be overlapped by using a single scheme to evolve them simultaneously. Therefore, the new method automatically integrates the local truncation error or defect. Previous strategies can be cast as particular cases of the one introduced in this study when the overlapping part is omitted. This leads to new types of schemes that are naturally represented as general linear (GL) methods, which are perfectly suited for this strategy, as we demonstrate.

The GL methods introduced in this study have built-in asymptotically correct global and local error estimators. These methods propagate at least two quantities; one of them is the solution, and the other one can be either the global error or equivalently another solution that can be used to determine the global errors. We show that two conditions are required for GL methods to propagate global errors: (i) a particular relation between the truncation error of the two quantities and (ii) a decoupling property between the errors of its two outputs. The GL framework encapsulates all linear time-stepping algorithms and provides a platform for robust algorithms with built-in error estimates. Moreover, this encapsulated treatment simplifies the analysis of compound schemes used for global error analysis; for instance, stability analysis turns out to be much simpler in this representation. The algorithms introduced in this study work well with variable time steps; however, we do not discuss global error control here. This is a topic for a future study. Nevertheless, local error control can be used as before with global error estimates being a diagnostic of the output.

Recent work on global error control by Kulikov, Weiner, et al. \([41, 42, 43]\) extends the quasi-consistency...
property introduced by Skeel [44] and recently advanced by Kulikov [45]. Moreover, Weiner and Kulikov [46] extend these ideas to peer methods by using a peer-triplets strategy. These strategies give good results in terms of global error control on prototypical problems. The GL-based algorithm proposed in this study bears a more general representation of the methods discussed above. We will also show how Runge-Kutta triplets [38] and by proxy the peer triplets [46] are naturally represented as GL methods. Furthermore, we illustrate the connection between the strategy proposed here and other strategies that are briefly discussed in Sec. 5.

Methods with global error estimation provide a posteriori error estimates, whereas traditional implementations that use local error control do not — and in some cases, which are not known a priori — this aspect can be critical. The rest of the manuscript is organized in the following way. Section 2 introduces the background for the theoretical developments and discusses different strategies for estimating the global errors, which include developments that form the basis of the proposed approach. In Sec. 3 we discuss the GL methods that are used to represent practical algorithms. The analysis of these schemes and examples are provided in Sec. 4. In Sec. 5 we discuss the relationship between the approach introduced here and related strategies and show how the latter are particular instantiations of the former. Several numerical experiments are discussed in Sec. 6, and concluding remarks are presented in Sec. 7.

2. Global error estimation

Let us consider a one-step linear numerical discretization method for non-autonomous (1.1),

\[ y_{n+1} = y_n + \Delta t \Phi(t_n, y_n, \Delta t_n), \quad y_0 = y(t_0), \quad n = 1, 2, \ldots, N_T, \]  

(2.1)

where \( \Phi \) is called the Taylor increment function with \( \Phi(t_n, y_n, 0) = f(t_n, y_n) \). We denote the time series obtained via (2.1) with step \( \Delta t \) by \( \{y_{\Delta t}\} \). A method of order \( p \) for a sufficiently smooth function \( f \) satisfies

\[ \|y(t_n + \Delta t) - y_{n+1}\| \leq C_1 \Delta t^{p+1}, y_n = y(t_n), \]

for a constant \( C_1 \). The local error \( d_{p+1}(t) \Delta t^{p+1} \) then satisfies

\[ y(t + \Delta t) - y(t) - \Delta t \Phi(t, y(t), \Delta t) = d_{p+1}(t) \Delta t^{p+1} + O(\Delta t^{p+2}). \]  

(2.2)

Under sufficient smoothness assumptions [7, 47], convergent numerical methods have an asymptotic expansion of the global error of the form

\[ y(t) - y_{\Delta t}(t) = e_p(t) \Delta t^p + \cdots + e_N(t) \Delta t^N + E_{\Delta t}(t) \Delta t^{N+1}, \]  

(2.3)

where \( E_{\Delta t}(t) \) is bounded on \( t_0 < t \leq T \) and \( 0 \leq \Delta t \leq \Delta T \) for some \( \Delta T \) [7, 6], so that

\[ y(t) - y_{\Delta t}(t) = e_p(t) \Delta t^p + O(\Delta t^{p+1}). \]  

(2.4)

Moreover, one can show that \( e_p(t) \) satisfies

\[ e_p'(t) = \frac{\partial f}{\partial y}(t, y) \cdot e_p(t) + d_{p+1}(t), \quad e_p(t_0) = 0. \]  

(2.5)

The other \( e_j(t), j > p \) terms satisfy similar equations.
B-series notation. To alleviate the analysis difficulties that come with large $p$, we use B-series \([48, 49]\). We now establish briefly the standard notation \([50]\). Let $\mathcal{T}$ be a set of ordered indexes $\mathcal{T}_q = \{j_1 < j_2 < j_3 < \cdots < j_q\}$ with cardinality $q$. A labeled tree of order $q$ is a mapping $\tau: \mathcal{T}_q \setminus \{j_1\} \rightarrow \mathcal{T}_q$ such that $\tau(j) < j$, $\forall j \in \mathcal{T}_q \setminus \{j_1\}$. The set of all labeled trees of order $q$ is denoted by $LT_q$. The order of a tree is denoted by $\rho(\tau) = q$. An equivalence class of order $q$ is defined as the permutation $\sigma: \mathcal{T}_q \rightarrow \mathcal{T}_q$ such that $\sigma(j) = j$, $\forall j \in \mathcal{T}_q$. These unlabeled trees of order $q$ are denoted by $T_q$, and the number of different monotonic labelings of $\tau \in T_q$ is denoted by $\alpha(\tau)$. Also, $T_q^\# = T_q \cup \emptyset$, where $\emptyset$ is the empty tree and the only one with $\rho(\emptyset) = 0$. We will also use graphical notation to represent derivatives discussed in \([50, 6]\).

For example, the tree $Y$ corresponds to $f'f''(f, f)$. The trees of order 4 are $T_4 = \{\Psi, \Psi, Y, i\}$.

$\alpha(\tau) = 1$ for $\tau \in T_q \setminus \{\Psi\}$, $\alpha(\Psi) = 3$. In this context, a numerical method applied to (1.1) with $f$ $p$-times continuous differentiable is of order $p$ if the asymptotic expansion of the numerical solution satisfies the B-series expansion up to order $p$, where the B-series is a proxy for the exact solution expressed as

$$\sum_{\tau \in T_p} \frac{\Delta t^{p(\tau)}}{\rho(\tau)!} \alpha(\tau) F(\tau)(y),$$

where $\gamma = \gamma_p$.

Estimating global errors using two methods. We now introduce the general global error estimation strategy used in this study. This approach relies on propagating two solutions through a linear time-stepping process that has the property of maintaining a fixed ratio, denoted by $\gamma$, between their truncation error terms. This result will play a crucial role in constructing the new methods discussed in Sec. 4 and can be stated as follows.

**Theorem 2.1** (Global error estimation with two methods). Consider numerical solutions $\{y_n\}$ and $\{\tilde{y}_n\}$ of (1.1) obtained by two time-stepping methods that satisfy (2.3), which are started from the same exact initial condition. If the local errors of the two methods with increments $\Phi$ and $\tilde{\Phi}$ satisfy

$$y(t + \Delta t) - y(t) = \Delta t \Phi(t, y(t), \Delta t) = \gamma d_{p+1}(t_n) \Delta t^{p+1} + O(\Delta t^{p+2}) \text{ and}$$

$$y(t + \Delta t) - y(t) = \Delta t \tilde{\Phi}(t, y(t), \Delta t) = \gamma d_{p+1}(t_n) \Delta t^{p+1} + O(\Delta t^{p+2}),$$

where $d_{p+1}(t_n) = \frac{1}{(p+1)!} \sum_{\tau \in T_{p+1}} \alpha(\tau) a(\tau) F(\tau)(y(t))$ with constant $\gamma \neq 1$, then the global error can be estimated as

$$\varepsilon_p(n\Delta t) = \frac{1}{1 - \gamma} (\tilde{y}_n - y_n) = \varepsilon_p(t_n) \Delta t^p + O(\Delta t^{p+1}) = \varepsilon(t_n) + O(\Delta t^{p+1}),$$

when $y_0 = \tilde{y}_0 = y(t_0)$, where $y$ is the solution of (2.6a) and $\tilde{y}$ of (2.6b).

**Proof.** We will use (2.3) and (2.5) to write the global error equations for the two methods with nearby solutions obtained by methods (2.6). Therefore, by using (2.5) directly for the two methods in (2.6) we have that the global errors evolve as

$$e'_p(t) = \frac{\partial f}{\partial y}(t, y) \cdot e_p(t) + d_{p+1}(t), \quad e_p(t_0) = 0,$$

$$\tilde{e}'_p(t) = \frac{\partial f}{\partial y}(t, \tilde{y}_p(t)) \cdot \tilde{e}_p(t) + \gamma d_{p+1}(t), \quad \tilde{e}_p(t_0) = 0.$$
It follows that the solutions of the two ordinary differential equations satisfy \( \gamma e_p(t) = \tilde{e}_p(t) \). We can then verify (2.7) by inserting (2.4):

\[
\varepsilon_p(t_n) = \frac{1}{1-\gamma} (\tilde{y}_n - y_n)
= \frac{1}{1-\gamma} (y(t_n) - \dot{e}_p(t_n)\Delta t^p - y(t_n) + e_p(t_n)\Delta t^p + O(\Delta t^{p+1}))
= e_p(t_n)\Delta t^p + O(\Delta t^{p+1})
\]

for \( n = 1, 2, \ldots \). ∎

We will refer to \( \varepsilon \) as the exact global error and to \( \varepsilon_p \) as the order-\( p \) global error estimate. A particular case is \( \gamma = 0 \).

**Corollary 2.1.** If \( \gamma = 0 \) in Theorem 2.1, then we revert to the case of using two methods of different orders, \( p \) and \( p+1 \), to estimate the global errors for the method of order \( p \).

Moreover, under the assumptions of Theorem 2.1, one can always compute a higher-order approximation by combining the two solutions.

**Corollary 2.2.** A method of order \( p+1 \) can be obtained with conditions of Theorem 2.1 by

\[
\hat{y}_n = y_n + \varepsilon_p(t_n) = \frac{1}{1-\gamma} \tilde{y}_n - \frac{\gamma}{1-\gamma} y_n.
\]  

We note that a related analysis has been carried out in [13] with an emphasis on reusing standard codes for solving ODEs with global error estimation. Moreover, Murua and Makazaga take a similar approach to identify the global error from two related numerical solutions [40].

The result presented above is the basis for the developments in this study. We introduce a new type of method that provides a posteriori error estimates, and we show that this procedure generalizes all strategies that compute global errors by propagating multiple solutions or integrating related problems.

We next discuss the validity of this approach when variable time steps are used.

**Global errors with variable time steps.** Following [6], for variable time stepping we consider \( t_{n+1} - t_n = \nu(t_n)\Delta t, n = 1, 2, \ldots \). Instead of (2.5), the global error \( e_p(t) \) satisfies the following equation:

\[
e_p'(t) = \frac{\partial f}{\partial y}(t, y) \cdot e_p(t) + \nu(t)^pd_{p+1}(t), \quad e_p(0) = 0.
\]  

The results introduced in this study and summarized by Theorem 2.1 carry over to variable time stepping with \( \Delta t \) replaced by \( \Delta t_{\text{max}} = \max(\nu(t)\Delta t) \) and, therefore, allow the application of such strategies in practical contexts that involve adaptive time stepping.

In this study we do not address the problem of adapting the time step based on global error estimates; however, we discuss adaptivity based on local error estimates that are provided directly by the methods proposed here.
3. General linear methods

The methods introduced in this study are represented by GL schemes. In this context, we take advantage of the existing theory underlying GL methods and augment it with global error estimation capabilities. Two key elements are required. The first is a consequence of Theorem 2.1, which imposes a restriction on the truncation error. The second has to do with restricting the interaction between the solutions of the two methods. Both will be addressed in Sec. 4. The advantage of representing existing global error strategies in a more general framework is that it allows for the development of more robust methods. For this we need GL methods that carry at least two quantities as discussed above. In this section we present the GL methods theory without built-in error estimates.

General linear methods were introduced by Burrage and Butcher [51]; however, many GL-type schemes have been proposed to extend either Runge-Kutta methods [52] to linear multistep (LM) or vice versa [53, 54], as well as other extensions [55, 56, 57, 44] or directly as peer methods [58, 59]. GL methods are thus a generalization of both RK and LM methods, and we use the GL formalism to introduce new methods that provide asymptotically correct global error estimates.

We denote the solution at the current step \((n-1)\) by an \(r\)-component vector \(y^{[n-1]} = [y^{[n-1]}_1, y^{[n-1]}_2, \ldots, y^{[n-1]}_r]^T\), which in general contains the available information in the form of numerical approximations to the ODE solutions and their derivatives. To increase clarity, we henceforth denote the time index inside square brackets. The stage values (at step \(n\)) are denoted by \(Y_{(i)}\) and stage derivatives by \(f_{(i)} = f(Y_{(i)})\), \(i = 1, 2, \ldots, s\), and can be compactly represented as \(Y = \begin{bmatrix} Y^{T}_{(1)} & Y^{T}_{(2)} & \cdots & Y^{T}_{(s)} \end{bmatrix}^T\) and \(f = \begin{bmatrix} f^{T}_{(1)} & f^{T}_{(2)} & \cdots & f^{T}_{(s)} \end{bmatrix}^T\).

The \(r\)-value \(s\)-stage GL method is described by

\[
\begin{align*}
Y_{(i)} &= \Delta t \sum_{j=1}^{s} A_{ij} f_{(j)} + \sum_{j=1}^{r} U_{ij} y^{[n-1]}_{(j)}, \quad i = 1, 2, \ldots, s, \\
y^{[n]}_{(i)} &= \Delta t \sum_{j=1}^{s} B_{ij} f_{(j)} + \sum_{j=1}^{r} V_{ij} y^{[n-1]}_{(j)}, \quad i = 1, 2, \ldots, r,
\end{align*}
\]

(3.1)

where \((A, U, B, V)\) are the coefficients that define each method and can be grouped further into the GL matrix \(M:\)

\[
\begin{bmatrix}
Y \\
y^{[n]}
\end{bmatrix} = \begin{bmatrix} A \otimes I_m & U \otimes I_m \\
B \otimes I_m & V \otimes I_m
\end{bmatrix} \begin{bmatrix}
\Delta tf \\
y^{[n-1]}
\end{bmatrix} = M \begin{bmatrix}
\Delta tf \\
y^{[n-1]}
\end{bmatrix}.
\]

Expression (3.1) is the most generic representation of GL methods [6, p. 434] and encompasses both RK methods \((r = 1, s > 1)\) and LM methods \((r > 1, s = 1)\) as particular cases. In this work we consider methods with \(r = 2\), where the first component represents the primary solution of the problem (2.6a) and the second component can represent either the defect (2.7) or the secondary component (2.6b). Only multistage-like methods are considered; however, multistep-multistage methods \((r > 2, s > 1)\) are also possible.

In this study we use self-starting methods that do not require a solution history and specialized starting procedures. In general the initial input vector \(y^{[0]}\) can be generated through a starting procedure,
$S = \{ S_i : \mathbb{R}^m \rightarrow \mathbb{R}^m \}_{i=1, \ldots, r}$, represented by generalized RK methods; see [50, Chap. 53] and [60]. The final solution is typically obtained by applying a “finishing procedure,” $F = \{ F_i : \mathbb{R}^m \rightarrow \mathbb{R}^m \}_{i=1, \ldots, r}$, to the last output vector; in our case this is also trivial.

### 3.1. Order conditions for GL methods

The order conditions rely on the theory outlined by Butcher et al. [50, 61, 60]. The derivatives of the numerical and exact solution are represented by rooted trees and expressed as a B-series [62, 63]. We use an algebraic criterion to characterize the order conditions for GL methods as follows. Let $\tau \in T$ and $E^{(\theta)} : T \rightarrow \mathbb{R}$, the “exact solution operator” of the differential equation (1.1), which represents the elementary weights for the exact solution at $\theta \Delta t$. If $\theta = 1$, then $E^{(1)}(\tau) = E(\tau) = (\sigma(\tau) \alpha(\tau))/\rho(\tau)!$ and $E \left( \begin{array}{c} \cdots, \cdot, \vdots, \cdot, \vdots, \cdot \end{array} \right) = \{ 1, 1/2, 1/3, 1/6 \}$ for $\rho(\tau) \leq 3$. We consider a mapping $\xi_i : T \rightarrow \mathbb{R}$, which represents the elementary weights of the input of the time-stepping process. In general, $\xi_i(\tau) = \Phi(i)(\tau)$, where $\Phi(i)(\tau)$, $i = 1, \ldots, r$, results from the starting procedure. Then for the general linear method $(A, U, B, V)$, one has

$$
\eta(\tau) = A \eta D(\tau) + U \xi(\tau), \quad \tilde{\xi}(\tau) = B \eta D(\tau) + V \xi(\tau), \quad \tau \in T,
$$

where $\eta, \eta D$ are mappings from $T$ to scalars that correspond to the internal stages and stage derivatives, respectively, and $\tilde{\xi}$ represents the elementary weights of the output vector. The exact weights are obtained from $[E \xi](\tau)$. The order of the GL method and its truncation error can be determined by a direct comparison between $\tilde{\xi}(\tau)$ and $[E \xi](\tau)$. More details can be found in [50], where a criterion for order $p$ is given for a GL method described by $M$ and the starting procedure. Therefore, in general, an order $p$ GL method results from the direct comparison of elementary weights of $[M^n](\tau) = [E^n \xi](\tau) \forall \tau_j$, $\rho(\tau_j) \leq p$. This criterion is a direct consequence of [61, Def. 3 and Prop. 1]. In our case, methods satisfying Theorem 2.1 can be developed by enforcing (2.6) on the corresponding solution vector. We further discuss the order conditions in our case in Sec. 4.1.

### 3.2. Linear stability of GL methods

The linear stability analysis of method (3.1) is performed on a linear scalar test problem: $y'(t) = ay(t)$, $a \in \mathbb{C}$. Applying (3.1) to the test problem yields a solution of the form $y^{[n+1]} = R(z) y^{[n]}$,

$$
R(z) = V + zB \left( I_s - zA \right)^{-1} U,
$$

$$
\Phi(w, z) = \det(wI_r - R(z)),
$$

where $z = a \Delta t$, $R(z)$ is referred to as the stability matrix of the scheme and $\Phi(w, z)$ is the stability function.

For given $z$, method (3.1) is linearly stable if $\rho(R(z))$, the spectral radius of $R(z)$, is contained by the complex unit disk. The stability region is defined as the set $S = \{ z \in \mathbb{C} : \rho(R(z)) \leq 1 \}$. The linear stability region provides valuable insight into the method’s behavior with nonlinear systems. Additional details can be found in [50].
4. Methods with global error estimation (GEE)

We now introduce GL methods with global and local error estimation. We focus on Runge-Kutta-like schemes in the sense that the resulting GL methods are self-starting multistage schemes. We therefore restrict our exposition to methods that carry two solutions explicitly and thus \( r = 2 \). Generalizations are possible but not addressed here. The methods are given in two forms that use different input and output quantities. The first form is denoted by GL\( y \tilde{y} \) and evolves two solutions of the ODE problem \( y \) and \( \tilde{y} \).

Methods GL\( y \tilde{y} \) with \( y(1) = y \) and \( y(2) = \tilde{y} \) take the following form:

\[
Y(i) = \Delta t \sum_{j=1}^{s} A_{ij} f(Y(j)) + U_{i,1} y^{[n-1]} + U_{i,2} y^{[n-1]}, \quad i = 1, 2, \ldots, s,
\]

\[
y^{[1]} = \Delta t \sum_{j=1}^{s} B_{1,j} f(Y(j)) + V_{1,1} y^{[n-1]} + V_{1,2} y^{[n-1]}, \tag{4.1}
\]

\[
y^{[n]} = \Delta t \sum_{j=1}^{s} B_{2,j} f(Y(j)) + V_{2,1} y^{[n-1]} + V_{2,2} y^{[n-1]}.
\]

We will consider \( V = I_r \), although more general forms can also be considered. The second form, denoted by GL\( y \varepsilon \), is given as a method that evolves the solution of the base method and the error explicitly, \( y \) and \( \varepsilon \), as \( \{y^{[n]}, \varepsilon^{[n]}\} = GLy\varepsilon(\{y^{[n-1]}, \varepsilon^{[n-1]}\}) \), and has a more practical flavor. In this case we take \( y(1) = y \) and \( y(2) = \varepsilon \) in (4.1). Both forms can be expressed as GL methods with tableaux \((A_{y\tilde{y}}, U_{y\tilde{y}}, B_{y\tilde{y}}, V_{y\tilde{y}})\) and \((A_{y\varepsilon}, U_{y\varepsilon}, B_{y\varepsilon}, V_{y\varepsilon})\), respectively; and one can switch between the forms as explained below.

**Lemma 4.1.** GL methods of form (4.1) that satisfy the conditions of Theorem 2.1 with coefficients \((A_{y\tilde{y}}, U_{y\tilde{y}}, B_{y\tilde{y}}, V_{y\tilde{y}})\), where \( y^{[n]} = [(y^{[n]}), (\tilde{y}^{[n]}), (\varepsilon^{[n]}), y^{[n]}] \), and \((A_{y\varepsilon}, U_{y\varepsilon}, B_{y\varepsilon}, V_{y\varepsilon})\), where \( y^{[n]} = [(y^{[n]}), (\varepsilon^{[n]}), (\varepsilon^{[n]}), y^{[n]}] \), are related by

\[
A_{y\tilde{y}} = A_{y\varepsilon}, \quad V_{y\tilde{y}} = V_{y\varepsilon}, \quad U_{y\tilde{y}} = U_{y\varepsilon} T_{y\varepsilon}^{-1}, \quad B_{y\tilde{y}}(1,:) = T_{y\varepsilon} B_{y\varepsilon}, \tag{4.2}
\]

where \( T_{y\varepsilon} = \begin{bmatrix} 1 & 0 \\ 1 & 1 - \gamma \end{bmatrix} \).

**Proof.** We start with a GLy\( y \varepsilon \) method defined by \((A_{y\varepsilon}, U_{y\varepsilon}, B_{y\varepsilon}, V_{y\varepsilon})\) and write the resulting expression by applying (4.1) with \( y^{[1]} = y^{[n]} \) and \( y^{[2]} = \varepsilon^{[n]} \). We then replace \( \varepsilon^{[n]} \) with \( \frac{1}{1 - \gamma} (\tilde{y}^{[n]} - y^{[n]} \varepsilon) \) as in Theorem 2.1, (2.7). The resulting expression can then be written as a GLy\( \tilde{y} \) scheme with \( y^{[1]} = y^{[n]} \) and \( y^{[2]} = \tilde{y}^{[n]} \).

This calculation leads to (4.2). This transformation is unique as long as \( \gamma \neq 1 \).

The following algorithm is proposed.

<table>
<thead>
<tr>
<th>Algorithm: General linear methods with global error estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize: ( y^{[0]} = y(t_0) = y_0, \varepsilon^{[0]} = \varepsilon(t_0) = 0 ).</td>
</tr>
<tr>
<td>Solve: ( y' = f(t, y) ) using</td>
</tr>
<tr>
<td>( {y^{[n]}, \varepsilon^{[n]}} = GLy\varepsilon({y^{[n-1]}, \varepsilon^{[n-1]}}), ) [solution, GEE] (4.3a)</td>
</tr>
<tr>
<td>( \varepsilon_{\text{loc}} = \varepsilon^{[n]} - \varepsilon^{[n-1]} ), [local error] (4.3b)</td>
</tr>
<tr>
<td>( \tilde{y}^{[n]} = y^{[n]} + \varepsilon^{[n]} ), [high order] (4.3c)</td>
</tr>
</tbody>
</table>

8
If GLy is used, then (4.3c) is estimated as \( \tilde{y}^{[n]} = \frac{1}{\gamma} \tilde{y}^{[n]} - \frac{\gamma}{\gamma} \tilde{y}^{[n]} \). We note that this algorithm is suitable for self-starting GL methods using fixed and adaptive time steps and results in Eq. 8. For methods that are not self-starting, a finalizing procedure might be necessary after each step in order to extract the error components in (4.3a); however, this aspect is not addressed in this study. Moreover, as expected, the cumulative sum of the local errors yields an asymptotically correct approximation to the global error as suggested by (4.3b) and illustrated through results in Eq. 8. This aspect has been used in [29] for lower-order approximations of the local errors.

Adjust Consistency and preconsistency analysis, and non-autonomous systems. One can show that the GLy methods are consistent and preconsistent with abscissa vector \( c = A \mathbb{1} \) if the solution is initialized with the solution and no derivatives, as is the case in this study. Therefore, when solving non-autonomous systems, one needs to evaluate the function \( f \) in (4.1) for stage \( j \) at time \( t_{n-1} + c_j \Delta t \).

4.1. Order conditions for GEE methods

The order conditions are based on the algebraic representation of the propagation of the B-series through the GL process as discussed in Sec. 3.1. For form GLy, we need to set \( \xi_{(1,2)}(0) = 1 \) and \( \xi_{(1,2)}(\tau_q) = 0, q = 1, 2, \ldots \), resulting in Runge-Kutta like conditions such as \( \hat{\xi}_{(i)}(\bullet) = \sum_j b_{i,j} \), \( |E\xi|_{(i)}(\bullet) = 1 \), and \( \tilde{\xi}_{(i)}(\bullet) = B_i . A \mathbb{1} \), \( |E\xi|_{(i)}(\bullet) = 1/2 \), where \( \hat{\xi} \) represents the numerical output as introduced in (3.2), and \( E\xi \) corresponds to the elementary weights of the exact solution.

Additional constraints are imposed so that Theorem 2.1 applies directly as a result of the GL process. In particular, we need to enforce relations (2.6). To this end, we consider an order \( p \) GLy method by setting \( E\xi_1(\tau) = \hat{\xi}_1(\tau) = \hat{\xi}_2(\tau) \), for all \( \tau \in T_p \), and

\[
\gamma \left( E\xi_1(\tau) - \hat{\xi}_1(\tau) \right) = E\xi_2(\tau) - \hat{\xi}_2(\tau), \quad \tau \in T_{p+1}, \gamma \neq 1,
\]

assuming that the inputs of the GL process \( \gamma \xi_1(\tau) = \xi_2(\tau), \tau \in T_{p+1} \). For instance, if \( p = 2 \) and \( r = 2 \), then (4.4) yields: \( \gamma \left( \frac{1}{3} - B_{1,2} A \mathbb{1} \right) = \left( \frac{1}{3} - B_{2,2} A \mathbb{1} \right) \) for \( \tau = \frac{1}{3} \) and \( \gamma \left( \frac{1}{3} - B_{1,2} (A \mathbb{1})^2 \right) = \left( \frac{1}{3} - B_{2,2} (A \mathbb{1})^2 \right) \) for \( \nabla \), where the exponent is applied componentwise. Then the error of the base method satisfies

\[
\varepsilon_p = \frac{\Delta t^p}{p!} \sum_{\tau \in T_{p+1}} \left( E\xi_1(\tau) - \hat{\xi}_1(\tau) \right) F(\tau)(y) + \mathcal{O}(\Delta t^{p+1}) .
\]

Expression (4.4) is equivalent to imposing (2.6). We also impose stability order [64] \( \hat{p} = p + 3 \): \( \Phi(\exp(z), z) = \mathcal{O}(\Delta t^{\hat{p}}) \) defined in (3.4), to obtain robust methods.

The two solutions that evolve through the GL process are connected internally, and therefore the error estimation may be hindered in the case of unstable dynamics as discussed in [16]. In Fig. 5 we illustrate such a behavior, and in Fig. 6 we show how the convergence is affected if the decoupling conditions (see Proposition 4.1) are satisfied in turn. To this end, we require that the elementary differentials of the two methods resulting from applying the GL method be independent from each other’s entries for all trees of order \( p + 1 \) and \( p + 2 \). According to (3.2), the output weights depend on the method coefficients and the
input weights. This requirement therefore can be expressed as

\[
\tilde{\xi}_{(\ell)}(\tau_j)(\xi_{(1)}(\tau_k), \xi_{(2)}(\tau_k), \xi_{(1,2)}(\tau_q)) = \tilde{\xi}_{(\ell)}(\tau_j)(\xi_{(1)}(\tau_k), \xi_{(1,2)}(\tau_q)),
\]

for all \(j, k, \rho(\tau_k), \rho(\tau_j) \in \{p + 1, p + 2\}, \rho(\tau_q) \in \{1, \ldots, p\},\)

where \(\xi_{(\ell)}(\tau_j)\) is the coefficient of input \(\ell\) corresponding to tree index \(j\) and \(\tilde{\xi}_{(\ell)}(\tau_k)\) is the coefficient of GL output \(i\) corresponding to tree index \(k\). In other words, output 1 that corresponds to tree index \(j\) does not depend on input 2 of tree index \(k\), and the same for output 2 and input 1.

We recapitulate a lemma that states that elementary weights in the expression of the B-series are independent. The following classical result captures this aspect.

**Lemma 4.2** (Independence of elementary differentials [50]). The elementary differentials are independent. Moreover, the values of the distinct elementary differentials for \(y^{(j)} \cdot y = \prod_{j=1}^{k}(y^{(j)})^{m_j}/m_j!\), \(y^{(j)}(t_0) = 0\)

are given by \(F(\tau)(y(t_0)) = e_i\), where \(k\) is the number of resulting trees when the root is removed and \(m_j\)

is the number of copies of \(\tau_j\), where \(y^{(j)}\) is the \(j\)th component.

We now establish a mechanism by which conditions for the method’s coefficients are set so that (4.5) is satisfied.

**Lemma 4.3.** The elementary weights of a GL method (4.1) with \(V = I\) satisfy

\[
\tilde{\xi}(\tau_j) = K + \xi(\tau_j) + BU\xi(\tau_{p-1}) + G(\tau_{[1,2,\ldots,p-2]}), \quad \rho(\tau_j) = p, \quad j = 1, 2, \ldots,
\]

where \(K\) is a constant that depends on the tree index, \(\tau_{[\ell]}\) are all the trees that correspond to order \(\ell\) \((\rho(\tau_i) = \ell, \forall i)\), and \(G\) is a function of \(\tau\) of order 1 to \(p - 2\), and \(\xi(\tau_q) = [\xi_1(\tau_q), \xi_2(\tau_q)]^T\).

**Proof.** For the first tree \(\tau_0\) we have \(\eta D(\tau_0) = 0\). The next tree is \(\tau_1 = \bullet\), for which \(\eta D(\bullet) = 1\). Relation (3.2) gives

\[
\eta(\tau_1) = A \eta D(\tau_1) + U \xi(\tau_1) = A I + U \xi(\tau_1).
\]

This is allowed by Lemma 4.2. Next, for order 2, we have \(\eta D(\tau_2) = \eta D(\bullet) = \eta(\tau_1)\)

and

\[
\eta(\tau_2) = A \eta D(\tau_2) + U \xi(\tau_2) = A \cdot (A I + U \xi(\tau_1)) + U \xi(\tau_2).
\]

For the next tree we have \(\eta D(\tau_3) = \eta D(\tau_2) = (\eta(\tau_1))^2\) and

\[
\eta(\tau_3) = A (A I + U \xi(\tau_1))^2 + U \xi(\tau_3),
\]

where the power is taken componentwise. The last third-order tree gives \(\eta D(\tau_4) = \eta D(\tau_2) = \eta(\tau_2)\) and

\[
\eta(\tau_4) = A (A \cdot (A \eta D(\tau_1) + U \xi(\tau_1)) + U \xi(\tau_2) + U \xi(\tau_4))
= A^3 I + A^2 U \xi(\tau_1) + AU \xi(\tau_2) + U \xi(\tau_4).
\]
We then arrive at the following recurrence formula:

$$
\eta D(\tau[p]) = W(\eta(\tau\{1,2,\ldots,p-2\}))+U\xi(\tau[p-1]),
$$

(4.6)

where $W(\eta(\tau\{1,2,\ldots,p-2\}))$ is a function of trees that are of order $1,2,\ldots,p-2$ and of the method’s coefficients $A$ and $U$. Similarly, one can verify that the recurrence for the output quantities satisfies

$$
\hat{\xi}(\tau[p]) = BW(\eta(\tau\{1,2,\ldots,p-2\}))+BU\xi(\tau[p-1]) + \xi(\tau[p]).
$$

(4.7)

For $p=3$ we have trees with index 3 and 4, and the output is obtained again from (3.2) and using the above derivations as

$$
\begin{align*}
\hat{\xi}(\tau_3) &= B((A\mathbb{I}_s)^2 + (U\xi(\tau_1))^2) + \xi(\tau_3) = BW(\xi(\tau_1)) + \xi(\tau_3), \\
\hat{\xi}(\tau_4) &= BA(\mathbb{I} + U\xi(\tau_1)) + BU\xi(\tau_2) + \xi(\tau_4) = BW(\xi(\tau_1)) + BU\xi(\tau_2) + \xi(\tau_4),
\end{align*}
$$

which gives off the recurrence. For arbitrary $p$ an inductive argument can be made. One needs to repeat the steps above by using (3.2) and computing the derivatives (4.6), which will yield $BU\xi(\tau_j)$ terms for $\rho(\tau_j) = p-1$, $\xi(\tau_j)$ for $\rho(\tau_j) = p$, and more complicated expressions for $\rho(\tau_j) = 1,2,\ldots,p-2$ as in (4.7).

Proposition 4.1 provides sufficient conditions for the independence assumptions (4.5).

**Proposition 4.1** (Output independence of GEE method). A GEE method (4.1) with $r=2$ for which the off-diagonal elements of matrix $BU$ are zero satisfies the independence assumption (4.5).

**Proof.** Using the results of Lemma 4.3, we compute the output $i$ for trees of order $p+1$ and assume that the input is consistent of order $p$, that is, $\xi_i(\tau\{1,2,\ldots,p\}) = 0$. We obtain

$$
\hat{\xi}_i(\tau[p+1]) = K + \xi_i(\tau[p+1]) + (BU)_i: \xi(\tau[p]) + G(\tau\{1,2,\ldots,p-1\}) = K + \xi_i(\tau[p+1]), \quad i = 1,2.
$$

For $p+2$ and the fact that $BU$ is a diagonal matrix, we obtain

$$
\begin{align*}
\hat{\xi}_i(\tau[p+2]) &= K + \xi_i(\tau[p+2]) + (BU)_i: \xi(\tau[p+1]) + G(\tau[k\in\{1,2,\ldots,p\}]) \\
&= K + \xi_i(\tau[p+2]) + (BU)_i: \xi(\tau[p+1]) \\
&= K + \xi_i(\tau[p+2]) + (BU)_{i,i} \xi_i(\tau[p+1]), \quad i = 1,2.
\end{align*}
$$

A similar calculation for $p+3$ reveals that matrices $BUA$ and $Bdiag(A\mathbb{I})U$ need to have only diagonal entries. These conditions are necessary when dealing with mildly stiff systems and for long-time simulations (see Figure 3). Collectively, we call these decoupling conditions. We note that such types of requirements were discussed both by Murua and Makazaga [40, 15] and by Shampine [13]. The work presented here generalizes this concept and applies it to a wider class of methods.
4.2. Order barriers

The order barriers of an \((r,s)\)-GL method apply to the methods with built-in error estimates (GEEs) of one order lower, and this is rather an upper bound. For example, a GEE method of order 2 has the same restrictions on \(r\) and \(s\), as does a GL method of order 3. These restrictions are reasonable because otherwise the underlying higher-order approximation discussed in Sec. 5.3 would violate the GL order barriers. In other words, the highest-order approximation in GEE cannot be done with fewer stages or steps than in the classical GL approach for the same order.

4.3. Variable steps

The self-starting methods introduced in this study are amenable to variable time steps. This fact results from the discussion at the end of Sec. 2 and will be illustrated numerically in Sec. 6.2 (see Fig. 8). However, the construction of GEE based on GL methods that propagate quantities from past steps (i.e., not self-starting) with adaptive time steps will necessarily have coefficients that depend on the time steps or be based on a rescaling strategy [65] and will have to satisfy the properties listed above. Moreover, a finalizing procedure might be necessary at every step in order to extract the error components. Constructing such a method can be involved, but it is nonetheless possible, as illustrated by particular instances in [10].

4.4. Second-order explicit GEE Runge-Kutta like methods

We now introduce a few methods of type (4.3). We begin with a detailed inspection of second-order methods. Schemes with \(s = 2\) are not possible because that would imply that one can have an explicit third-order method by (2.9) with only two stages, a statement that is easy to disprove.

A method with \(s = 3\) and \(\gamma = 0\) in GL\(\varepsilon\) form is given by the following tableaux,

\[
\mathbb{M}_{y\varepsilon} = \begin{bmatrix}
0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 10 \\
1/4 & 1/4 & 0 & 1 & -1 \\
1/12 & 1/12 & 5/6 & 1 & 0 \\
1/12 & 1/12 & -1/6 & 0 & 1
\end{bmatrix}, \tag{4.8}
\]

where the four blocks represent \((A_{y\varepsilon}, U_{y\varepsilon}, B_{y\varepsilon}, V_{y\varepsilon})\) as discussed above. The diagonal \(BU\) condition is satisfied ([BU]); however, the \(BAU\) one is not ([BAU]). Method (4.8) can then be expressed as follows:

\[
Y_1 = y^{\lfloor n-1\rfloor}, \tag{4.9a}
\]

\[
Y_2 = y^{\lfloor n-1\rfloor} + 10\varepsilon^{\lfloor n-1\rfloor} + \Delta t f(Y_1), \tag{4.9b}
\]

\[
Y_3 = y^{\lfloor n-1\rfloor} - \varepsilon^{\lfloor n-1\rfloor} + \Delta t \left( \frac{1}{4} f(Y_1) + \frac{1}{4} f(Y_2) \right), \tag{4.9c}
\]

\[
y^{\lfloor n\rfloor} = y^{\lfloor n-1\rfloor} + \Delta t \left( \frac{1}{12} f(Y_1) + \frac{1}{12} f(Y_2) + \frac{5}{6} f(Y_3) \right), \tag{4.9d}
\]

\[
\varepsilon^{\lfloor n\rfloor} = \varepsilon^{\lfloor n-1\rfloor} + \Delta t \left( \frac{1}{12} f(Y_1) + \frac{1}{12} f(Y_2) - \frac{1}{6} f(Y_3) \right). \tag{4.9e}
\]
In (4.9) we note the Runge-Kutta structure; however, we see that the defect takes an active role in the stage calculations. Using (4.2), we obtain the GLỹ form as

\[
M_{y\tilde{y}} = \begin{bmatrix}
0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & -9 & 10 \\
\frac{1}{4} & \frac{1}{4} & 0 & 2 & -1 \\
\frac{1}{12} & \frac{1}{12} & \frac{5}{6} & 1 & 0 \\
\frac{1}{6} & \frac{1}{6} & \frac{2}{3} & 0 & 1
\end{bmatrix}.
\] (4.10)

In particular, (4.10) is expressed as

\[
\begin{align*}
Y_1 &= y^{[n-1]}, \\
Y_2 &= -9y^{[n-1]} + 10\tilde{y}^{[n-1]} + \Delta t f(Y_1), \\
Y_3 &= 2y^{[n-1]} - \tilde{y}^{[n-1]} + \Delta t \left( \frac{1}{4} f(Y_1) + \frac{1}{4} f(Y_2) \right), \\
y^{[n]} &= y^{[n-1]} + \Delta t \left( \frac{1}{12} f(Y_1) + \frac{1}{12} f(Y_2) + \frac{5}{6} f(Y_3) \right), \\
\tilde{y}^{[n]} &= \tilde{y}^{[n-1]} + \Delta t \left( \frac{1}{6} f(Y_1) + \frac{1}{6} f(Y_2) + \frac{2}{3} f(Y_3) \right), \\
e^{[n]} &= \tilde{y}^{[n]} - y^{[n]}.
\end{align*}
\] (4.11)

Here we note the explicit contribution of two solutions. A solution of order 3 is obtained according to (2.9) by \( \tilde{y}^{[n]} = \tilde{y}^{[n]} \) because \( \gamma = 0 \). Moreover, a local error estimate for \( y^{[n]} \) in (4.11d) corresponds to

\[
e_{\text{loc}} = e^{[n]} - e^{[n-1]} = \Delta t \left( \frac{1}{12} f(Y_1) + \frac{1}{12} f(Y_2) - \frac{1}{6} f(Y_3) \right),
\] (4.12)

which is an obvious statement. This is also obtained by replacing \( \tilde{y}^{[n-1]} \) by \( y^{[n]} \) in the right-hand sides of (4.11) and taking the differences between the two solutions or setting \( e^{[n-1]} = 0 \) in (4.9). We remark that methods solving for the correction require at least four stages. Although this seems like a marginal improvement, we expect to reap more benefits when considering higher-order methods.

Next we introduce three additional second-order methods that we used in our experiments. A second-order method with \( s = 3 \), \([\text{BU, BAU}]\), and \( \gamma = 0 \) in GLyε format is given by

\[
M_{y\varepsilon} = \begin{bmatrix}
0 & 0 & 0 & 1 & 4 \\
1 & 0 & 0 & 1 & 0 \\
4/9 & 2/9 & 0 & 1 & 0 \\
0 & -1/2 & 3/2 & 1 & 0 \\
1/4 & 1/2 & -3/4 & 0 & 1
\end{bmatrix}.
\] (4.13)

Another second-order method with \( s = 3 \) and \([\text{BU, BAU}]\) that is based on two second-order approxi-
mations \((\gamma = 1/2)\) in GLy\(\varepsilon\) format is given by

\[
M_{y\varepsilon} = \begin{bmatrix}
0 & 0 & 0 & 1 & -11/10 \\
1 & 0 & 0 & 1 & 13/30 \\
4/9 & 2/9 & 0 & 1 & 5/3 \\
5/12 & 5/12 & 1/6 & 1 & 0 \\
-1/4 & -1/4 & 1/2 & 0 & 1
\end{bmatrix}.
\]

A second-order method that results in having both \(\text{BU}\) and \(\text{BAU}\) diagonal matrices ([\(\text{BU}, \text{BAU}\)]) is a four-stage method with the following coefficients in GLy\(\bar{y}\) form:

\[
\begin{align*}
A_{2,1} &= 3/4, & A_{3,1} &= 1/4, & A_{3,2} &= 29/60, \\
A_{4,1} &= -21/44, & A_{4,2} &= 145/44, & A_{4,3} &= -20/11, \\
B_{1,1} &= 109/275, & B_{1,2} &= 58/75, & B_{1,3} &= -37/110, & B_{1,4} &= 1/6, \\
B_{2,1} &= 3/11, & B_{2,2} &= 75/88, & B_{2,4} &= -1/8,
\end{align*}
\]

where the rest of the coefficients are zero.

4.5. Third-order explicit GEE Runge-Kutta like methods

Closed-form solutions were difficult to obtain for methods of order 3. We therefore explored the space of such methods using a numerical optimization such as in [60]. One method of order 3 with \(\gamma = 0, s = 5\) stages, \([\text{BU, BAU}]\), and having significant negative real axis stability was found to have the following coefficients up to 40 digits of accuracy:

\[
\begin{align*}
a_{2,1} &= 236960494743637972313, & a_{3,1} &= 46526546497697123895, & a_{3,2} &= 10297879294026594958, \\
a_{4,1} &= 23634789958549583299, & a_{4,2} &= 79205144337496116638, & a_{4,3} &= 400518895910743782, \\
a_{5,1} &= 14755475729852471739, & a_{5,2} &= 13792728665289746282, & a_{5,3} &= 61274785225371016, \\
a_{5,4} &= 19873934796698722958, & b_{1,1} &= 61546696837458703723, & b_{1,2} &= 55810897929026623355, \\
b_{1,3} &= 3783262186984159168, & b_{1,4} &= 579977240678451339, & b_{1,5} &= 5944983295780563947, \\
b_{2,1} &= 37952002461487740355, & b_{2,2} &= 327797206874531339, & b_{2,3} &= 428051886861226631, \\
b_{2,4} &= 2280857693166989693, & b_{2,5} &= 546558094687445865, & b_{2,6} &= 17440018880147234390, \\
u_{1,1} &= 100943411362049826, & u_{1,2} &= 1061876656429106311, & u_{1,3} &= 550571213782304513, \\
u_{2,1} &= 7848690464456624568, & u_{2,2} &= 568438625167558413, & u_{2,3} &= 8457083484046479351, \\
u_{3,1} &= 88170308697388281, & u_{3,2} &= 88170308697388281, & u_{3,3} &= 88170308697388281, \\
u_{4,1} &= 860728277018754108, & u_{4,2} &= 10865804643649025911, & u_{4,3} &= 10090756366631371803,
\end{align*}
\]

We note that this is not an optimal method. It is just an example that was relatively easy to obtain and will be used in the numerical experiments. We note that method RK3(2)G1 (5.2) as introduced in [37] requires 8 stages (that number reduces to 7 because of its first same as last property).
5. Relationships with other global error estimation strategies and optimal methods

Here we discuss the relationship between our approach and existing strategies. Our purpose is to draw parallels with the strategy we introduce and emphasize some unique aspects regarding the asymptotics and necessary conditions. Specifically, we consider the following approaches.

Methods satisfying the exact principal error equation \[66, 67\]. This strategy was indirectly introduced by Butcher \[66\] in an attempt to break the order barriers of multistage methods under the alias “effective order.” Stetter \[67\] observed the relationship between the local and global error expansions and repurposed them for global error estimation. This strategy requires a triplet of methods. Examples of such triplets can be found in many studies \[66, 67, 68, 11, 12, 16, 17\]. Although this concept is attractive in terms of efficiency, Prince and Wright \[16\] noted a severe problem with using it for global error estimation: If the system has unstable components, then the error approximation becomes unreliable, as can be seen in Fig. 5. This is a severe limitation because having unstable components makes the local error estimates unreliable, and this is precisely the case when one would need to use global error estimation.

Solving the error equation \[18\]. If the Jacobian \(J\), of \(f\) is available, then (2.5) can be solved directly:
\[
\varepsilon'(t) = J \varepsilon(t) + [d_{p+1}(t_n) \Delta t].
\]
It is argued that the error equation can be solved with a cheaper, lower-order method. In this case, however, the bulk of the work resides in determining \(d_{p+1}\).

The Zadunaisky procedure \[36\]. This method is based on using Lagrange interpolation (denoted by \(P(t)\)) over several steps and then solving a perturbed system \(z' = f(t, P(t)) - P'(t) - f(t, z)\).

Solving for the correction \[36, 8\]. This approach follows the original development in \[36\] and further refined in \[37, 38, 14, 39\]. This is a generalization of the Zadunaisky procedure by which the error equation is slightly modified: \(z' = f(t, P(t) + \varepsilon) - P'(t)\). We will show that the equations being solved in this approach can be represented by using a general linear method representation (5.1).

The extrapolation approach \[69\]. This procedure is based on using the same method to propagate two independent solutions \(y_{\Delta t,n}\) and \(y_{\frac{\Delta t}{2},n}\), one with \(\Delta t\) and one with \(\Delta t/2\), respectively. The global error is computed by \(\varepsilon = \frac{p}{1-2^p}(y_{\Delta t,n} - y_{\frac{\Delta t}{2},n})\), where \(p\) is the order of the method. The factor used above is equivalent to setting \(\gamma = 1/2^p\) in the GEE methods (see Sec. 5.2 and \[70\]).

These strategies are particular instantiations of the GL method introduced here. They are based on propagating two solutions or propagating one solution and its defect, and we have seen that they can be related trivially through Lemma 4.1. More details are given in \[70\].

5.1. Approach for solving for the correction

We discuss in some detail the relationship between the approach of solving for the correction and the method proposed here. The Zadunaisky procedure follows along the same lines. Let us consider the Runge-Kutta methods that integrate the global errors introduced by \[37, 38, 39, 40, 15\]. The RK
The stacked stages in (5.1) is defined by the triplet \((A, B, C)\), and the interpolation operators are defined by \((B^*, D^*)\), where \(B^* \cdot [\theta^0, \theta^1, \ldots, \theta^s]^T\) yields the interpolant weight vector and \(D^* \cdot [\theta^0, \theta^1, \ldots, \theta^s]^T\) yields its derivative. In particular, \(D^*_{ij} = B^*_i \cdot j\), \(j = 1, \ldots, s\). We denote \(b^*_i(\theta) = \sum_{j=1}^s B^*_i \theta^j\), \(d^*_i(\theta) = \sum_{j=1}^s D^*_i \theta^j\), and consider the dense output formula given by

\[
P(t + \theta \Delta t) = y_n + \theta \Delta t \sum_{i=1}^s b^*_i f_i \quad \text{and} \quad P'(t + \theta \Delta t) = \theta \Delta t \sum_{i=1}^s d^*_i f_i.
\]

The error equation that is being solved is \(\varepsilon'(t) = f(t, P(t) + \varepsilon(t)) - P'(t)\). We denote \(\overline{B}^* = \text{diag}\{C\} \cdot B^* \cdot W(C)^T\), where \(W(C)\) is the Vandermonde matrix with entries \(C\); that is, \(\{W(C)\}_{ij} = C_i^{j-1}\); and \(\overline{D}^* = D^* \cdot W(C)^T\). The resulting method cast in GLy form (4.1) is

\[
\begin{bmatrix}
Y_1 \\
Y_2 \\
y_n+1 \\
\varepsilon_n+1
\end{bmatrix} =
\begin{bmatrix}
A & 0 & I_s & 0 \\
\overline{B}^* - AD^* & A & I_s & I_s \\
B^T & 0 & 1 & 0 \\
-B^T \overline{D}^* & B^T & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\Delta tf(Y_1) \\
\Delta tf(Y_2) \\
y_n \\
\varepsilon_n
\end{bmatrix}.
\]

\tag{5.1}
\]

Here we express the method for a scalar problem, in order to avoid the tensor products, and we represent the stacked stages in \(Y_{1,2}\). For example, method RK3(2)G1 [37] is given by the following Butcher tableau:

\[
\begin{array}{c|cccc}
0 & 0 \\
\frac{1}{2} & \frac{1}{2} & 0 \\
1 & -1 & 2 & 0 \\
1 & 1 & 2 & 0 & 0
\end{array}
\]

\[
B^* =
\begin{bmatrix}
1 & -\frac{3}{2} & 2 \\
0 & 2 & -\frac{4}{3} \\
0 & \frac{3}{6} & -\frac{2}{6} \\
0 & -1 & 1
\end{bmatrix}.
\]

\tag{5.2}
\]

The equations to be solved when using the procedure for solving for the correction are then equivalent to solving (5.1) when using the strategy (4.1) introduced here. The explicit coefficients corresponding to RK3(2)G1 (5.2) results in the following tableau in GLy\(\varepsilon\) form:

\[
M_{\varepsilon} =
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
-1 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1/6 & 2/3 & 1/6 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
-7/24 & 1/3 & 1/12 & -1/8 & 1/2 & 0 & 0 & 0 & 1 & 1 \\
7/6 & -4/3 & -1/3 & 1/2 & -1 & 2 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1/6 & 2/3 & 1/6 & 0 & 1 & 1 \\
1/6 & 2/3 & 1/6 & 0 & 0 & 0 & 0 & 1 & 0 \\
-1/6 & -2/3 & -1/6 & 0 & 1/6 & 2/3 & 1/6 & 0 & 0 & 1
\end{bmatrix}.
\]

\tag{5.3}
\]

This strategy of estimating the global errors is called Runge-Kutta triplets; a similar discussion can be extended for the peer-triplets strategy [46]. The Zadunaisky procedure can be shown to have a similar...
5.2. Global error extrapolation

Let us consider again the Runge-Kutta methods defined by the triplet \((A, B, C)\) of order \(p\). By applying the extrapolation algorithm, we obtain the method in the GL format,

\[
\begin{bmatrix}
Y_1 \\
Y_2 \\
Y_3 \\
y_{n+1} \\
\varepsilon_{n+1}
\end{bmatrix}
= \begin{bmatrix}
A & 0 & 0 & I_s & 0 \\
0 & 1/2A & 0 & I_s & \beta^{-1}I_s \\
0 & 1/2B^T \otimes I_s & 1/2A & I_s & \beta^{-1}I_s \\
B^T & 0 & 0 & 1 & 0 \\
-\beta B^T & \beta B^T & \beta B^T & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\Delta tf(Y_1) \\
\Delta tf(Y_2) \\
\Delta tf(Y_3) \\
y_n \\
\varepsilon_n
\end{bmatrix},
\]  

(5.4)

where \(\beta = 1/2^p\), \(\gamma = 1/2^p\), and \(Y_{1,2,3}\) are the \(s\)-stage vectors corresponding to the original method stacked on top of each other. This is a method of type (4.1).

5.3. Underlying higher-order method

All the methods described in this study, including the one introduced in Cor. 2.2, use implicitly an underlying higher-order method to estimate the global error. In the case of methods satisfying the exact principal error equation and for solving the error equation, we find that the actual equation being solved is modified to include the truncation error term. By adding the two together one obtains a higher-order estimate (see [70]). In the case of the Zadunaisky algorithm and solving for the correction, one can recover the underlying higher-order method by replacing the error term in the perturbed equation with \(\hat{y}\) from (2.9) and using the conditions imposed on \(P\) (see [37]). We showed an example in Sec. 5.1. The extrapolation algorithm reveals the higher-order estimate directly in its derivation.

5.4. Optimal methods

We now discuss the need to balance the local truncation errors, which we would like to be as small as possible, with the ability to capture the global errors. Solving for the correction procedure is attractive because it allows the reuse of methods with well-established properties. In particular, one may consider methods that minimize the truncation errors. When such optimal methods are used in the context of global error estimation, however, one must verify that the errors are still quantifiable. For instance, if some of the truncation error terms are zero, then special care needs to be exercised because some problems may render the global error estimation “blind” to local error accumulation.

To illustrate this rather subtle point, we use the procedure for solving for the correction with method RK3(2)G1 (5.2) as introduced in [37]. This is a third-order scheme. Moreover, it has no errors that correspond to fourth-order trees \(\lambda\) and \(\bar{\lambda}\) but does not resolve exactly \(\lambda\) and \(\bar{\lambda}\); otherwise it would have been a fourth-order method. With the aid of Lemma 4.2 we construct a simple problem: \(y'_1 = 1\), \(y'_2 = \kappa_2 y_1^3\), \(y'_3 = \kappa_3 y_1^4\), where \(\kappa_i\) are some constants (e.g., \(\kappa_2 = 1/6\) and \(\kappa_3 = 4\)) and \(y(t_0) = [0, 0, 0]^T\). For
this problem, the RK3(2)G1 is a fourth-order method because the tall tree that would have affected the
third component is matched exactly by this method. Thus, the base method \( y \) has the same order as does
the higher-order companion, \( \hat{y} \). Therefore, the third component can cause the results to be unreliable. In
Fig. 1 we show the third component, which confirms the inadequacy in the error estimation procedure.
We note that this analysis also affects the schemes proposed in [40, 15].

![Figure 1: Failure to capture the global errors correctly for system \( y_1' = 1, y_2' = \kappa_2 y_1^3, y_3' = \kappa_3 y_1^2 \) solved with RK3(2)G1 (5.2) [37].]

6. Numerical results

In this section we present numerical results with a detailed set of test problems.

6.1. Test problems

We consider a set of simple but comprehensive test problems.

Problem [Prince42] is defined in [16] (4.2) by

\[
\begin{align*}
y' &= y - \sin(t) + \cos(t) , \quad y(0) = \kappa \\
y(t) &= \kappa \exp(t) + \sin(t) .
\end{align*}
\]

(6.1a)

(6.1b)

Here we take \( \kappa = 0 \). As a direct consequence of (6.1b), we see that any perturbation of the solution \( y \),
such as numerical errors, leads to exponential growth. Therefore we have an unstable dynamical system;
and even if we start with \( \kappa = 0 \), numerical errors will lead to an exponential solution growth. This is a
classical example that is used to show the failure of local error estimation in general and of global error
estimation by using the algorithm satisfying the exact principal error equation, [16] in particular.

A similar problem [Kulikov2013I] is defined by Kulikov [29] by

\[
\begin{align*}
y_1' &= 2t y_2^{1/5} y_4 , \quad y_2' = 10t \exp(5(y_3 - 1)) y_4 , \quad y_3' = 2t y_4 , \quad y_4' = -2t \ln(y_1) ,
\end{align*}
\]

(6.2)

so that \( y_1(t) = \exp(\sin(t^2)), y_2(t) = \exp(5 \sin(t^2)), y_3(t) = \sin(t^2) + 1, y_4(t) = \cos(t^2) \). This problem is
non-autonomous and exhibits unstable modes after some time.
To illustrate the error behavior over long-time integration, we consider problem \([Hull1972B4]\), which is a nonlinear ODE defined in \([71]\) (B4) by

\[
y'_1 = -y_2 - \frac{y_1 y_3}{\sqrt{y_1^2 + y_2^2}}, \quad y'_2 = y_1 - \frac{y_2 y_3}{\sqrt{y_1^2 + y_2^2}}, \quad y'_3 = \frac{y_1}{\sqrt{y_1^2 + y_2^2}},
\]

(6.3)

with \(y_0 = [3, 0, 0]^T\).

The next problem \([LStab2]\) is used to assess linear stability properties of the proposed numerical methods:

\[
y' = Ay, \quad y(0) = [y_1(0), y_2(0)]^T, \quad A = \begin{bmatrix} a & -b \\ b & a \end{bmatrix}, \quad \Lambda(A) = \{a + ib, a - ib\},
\]

(6.4a)

\[
\begin{align*}
y_1(t) &= \exp(at) \left( y_2(0) \cos(bt) - y_1(0) \sin(bt) \right), \\
y_2(t) &= \exp(at) \left( y_1(0) \cos(bt) + y_2(0) \sin(bt) \right).
\end{align*}
\]

(6.4b)

Here we take \(y_1(0) = 10\) and \(y_2(0) = 10\). This problem allows one to choose the position of the eigenvalues of the Jacobian, \(\Lambda(A)\), in order to simulate problems with different spectral properties.

6.2. Numerical experiments

We begin with simple numerical experiments that show when local error control without global error estimates is not suitable. We therefore compare the result of well-tuned numerical integrators that use local error control with the global error estimates for the same problem. We use Matlab’s ode45 integrator with different tolerances whenever we refer to methods with local error estimation.

In Fig. 2 we show the errors over time for problems \([Prince42]\) (6.1) and \([Kulikov2013I]\) (6.2). These problems are solved by using LEE methods, Figs. 2(a)-2(b), and GEE methods, (4.13) and (4.15), Figs. 2(c)-2(d); furthermore, the other GEE methods introduced above give similar results. The absolute error tolerance for LEE control is set to 1e-02. The methods with LEE systematically underestimate the global error levels as expected, whereas the methods with GEE capture the errors relatively well. Moreover, the global errors are captured well across components, as shown in Fig. 2(d). The LEE results are intended to show unreliability in accuracy; however, this setting is not suitable for comparing efficiency between LEE and GEE because LEE methods are not designed to account for or estimate the global error. The point of GEE methods is that they provide global error estimates whereas LEE methods do not; however, in this case the GEE methods do not perform step-size adaptivity to improve efficiency, as is done with the LEE methods.

In Fig. 3 we show the error behavior for problem \([Hull1972B4]\) (6.3) when long integration windows are considered. For LEE we set the absolute tolerance to 1e-05. In this case we observe an error drift to levels of 1e-03 over 1,000 time units. The method with GEE (4.15) can follow closely the error in time. We note that methods (4.8) and (4.13) with [BAU] do not perform well. Therefore, the built-in error estimator in the GEE method to maintain its accuracy over time, we need to decouple the two embedded schemes further by ensuring that BAU is a diagonal matrix; moreover, both GEE methods that satisfy
this property perform well on this test case. This aspect is observed for all methods introduced here, discussed further below, and illustrated in Fig. 6.

Convergence analysis. We next analyze the convergence properties of the methods discussed here. In Fig. 4 we show the convergence of the solution and of the error estimate. Here we illustrate the convergence of GEE methods of orders 2 (4.14) and 3 (4.16) for problem [Prince42] (6.1). All GEE methods introduced in this study converge with their prescribed order; moreover, the error estimate also converges with order $p+1$, as expected from (2.9). In Fig. 5 we show the behavior of global error estimation when using methods satisfying the exact principal error equation. Here we use method [16, (3.11)], which fails to capture the error magnitude as discussed in [16] because the estimated error is several orders of magnitude smaller than the true global error.

In Fig. 6 we illustrate the effect of having different coupling conditions satisfied. In particular, if BU
Figure 3: Errors when solving [Hull1972B4] (6.3) with LEE and GEE. For LEE we set the absolute tolerance to 1e-05. (a) During short integration times LEE satisfies the error tolerance well. (b) For longer times, however, we see an expected drift to error levels of 1e-03. (c) GEE method with ∆t = 0.005 and diagonal BAU (4.15) gives accurate error estimates even over long times, while GEE with non-diagonal BAU (4.8) and (4.13) do not.

Figure 4: Convergence of GEE methods for problem [Prince42] (6.1). In (a) we show the error in the solution obtained by the second-order GEE method (4.14) and the estimated global error, which follows it closely, as well as the difference between the true error and the error estimate. An asymptotic guide is provided by the red dashed lines. (b) This is the same as (a) but using the third-order method (4.16).

is not diagonal, the GEE may not converge at all (see Fig. 6(a)). This method is obtained by replacing the first row of $B_{y\bar{y}}$ by $[0,0,1]$ and $U_{y\bar{y}} = \begin{bmatrix} -4 & 1 & 3/4 \\ 5 & 0 & 1/4 \end{bmatrix}^T$ in (4.10). If BU is diagonal but BAU is not, the method may converge more slowly to the asymptotic regime as the high-order error coupling terms become negligible (see Fig. 6(b)). In Fig. 6(c), we show the convergence for the same problem when both BU and BAU are diagonal. Methods with diagonal BAU and non-diagonal BU are possible but of little interest because they do not satisfy consistency requirement (4.5) and would likely lead to results similar to those in Fig. 6(a).

Linear stability. We next look at the linear stability properties of the methods introduced in this study. In Fig. 7(a) we delineate the stability regions according to (3.3). In Fig. 7(b) we show numerical results
(a) Error estimation for methods with exact principal error equation for problem [Prince42]

(b) Method [16, (3.11)]

Figure 5: (a) Failure to capture the global errors correctly for problem [Prince42] (6.1) when using a method with exact error equation such as [16, (3.11)] with $\Delta t = 3e-02$ and (b) its convergence analysis.

(a) similar to (4.10), BU, BAU
(b) (4.13), BU, BAU
(c) (4.15), BU, BAU

Figure 6: Convergence of second-order GEE methods with different decoupling properties for [Kulikov2013I]. (a) Not satisfying BU leads to a lack of convergence. (b) Satisfying just BU but not BAU may lead to slower convergence. (c) Satisfying both BU or BAU leads to consistent convergence properties.

for problem [LStab2] (6.4) with $\lambda \Delta t = \{\frac{1}{7}, \frac{1}{2}, \frac{3}{4}, 1\} \times (-1 \pm \sqrt{1})$ when using method (4.13). As expected, all solutions except the one corresponding to $\lambda \Delta t = -1 \pm \sqrt{1}$ are stable, as can be interpreted from Fig. 7(a).

Adaptive time stepping and practical considerations. Two practical situations are illustrated next: (i) a situation in which local error control is used to adapt the time steps and the global error estimate is used to validate the solution accuracy and (ii) a situation in which a prescribed level of accuracy is needed, which requires recomputing the solution with a different time step. In the first case we consider a simple local error control using local error estimates (4.3b) provided by GEE (4.16). The local error control is achieved by using standard approaches [6, II.4]. In Fig. 8 we show the error evolution [Kulikov2013I] while attempting to restrict the local error to be around 1e-05. The time-step length is allowed to vary between 1e-03 and 1e-05. Note that the global error estimate remains faithful under the time-step change
Figure 7: Linear stability regions for four methods introduced in this study (a); and the solution of method (4.13) for problem [LStab2] (6.4) in (b) with different time steps. Spectral radius of (4.13) is indicated in (a) with marker * of the same color. Solutions in (b) are stable except the one (cyan line) for which the eigenvalues are outside the corresponding stability region (red line) in (a).

as expected based on the discussion at the end of Sec. 2 following (2.10). We also show that summing the local errors via cumulative sum (cumsum) yields precisely the global error as suggested by (4.3b). Similar conditions are used in the second case in which the target is to satisfy a prescribed tolerance. In a first run the time step is fixed to 0.0001, and an integration is carried out to the final time. The global error and the asymptotic are used to determine the time step that would guarantee a global error of 1e-04. A simple asymptotic analysis indicates that a time step of 0.00013 needs to be used in order to achieve that target for GEE (4.16). The solution is recomputed by using this smaller fixed time-step, and the global errors shown in Fig. 9 satisfy the prescribed global accuracy. We note that this case is just an illustration of the theoretical properties of the methods introduced here. It is likely a suboptimal strategy for achieving a prescribed global tolerance.

7. Discussion

In this study we introduce a new strategy for global error estimation in time-stepping methods. This strategy is based on advancing in time the solution along with the defect or, equivalently, two solutions that have a fixed relation between their truncation errors. The main idea is summarized in Theorem 2.1, and practical considerations are brought up by Proposition 4.1. This translates into a particular relation among the truncation errors of GL outputs and into a decoupling constraint. We note that this strategy can be seen as a generalization of the procedure for solving for the correction and of several procedures from the same class. We provide equivalent representation of these methods in the proposed GL form (4.1).

We briefly discuss several algorithms in this study. The methods with exact principal error equation
Figure 8: Time-step adaptivity using a local error controller. The GEE method (4.16) is used to integrate problem [Kulikov2013I] by restricting local error to be less than 1e-05. The error evolution is shown on the left, and the time-step length is shown on the right panel. The time-step length is allowed to vary between 1e-03 and 1e-05.

Figure 9: Error evolution for [Kulikov2013I] when using the GEE method (4.16) with $\Delta t = 0.001$ (left) and $\Delta t = 0.00013$ (right). The smaller time step has been calculated based on the GEE global error estimate to keep the global errors below 1e-04.

[67] are attractive because they offer global error estimates extremely cheaply; however, they were shown in [16] to be unreliable, as illustrated in Fig. 5. Strategies that directly solve the error equation need a reliable way of estimating the local errors and the availability of the Jacobian. We found these methods to be robust, especially the strategy proposed in [18] for low-order methods. The procedure for solving for the correction is arguably one of the most popular approaches for global error estimation. It is related to the Zadunaisky procedure and solving for the correction, as discussed, and a particular case of this approach is introduced in this study. The extrapolation algorithm is the most robust; however, it is also the most expensive and also a particular case of the methods introduced here. With respect to global error control, Kulikov and Weiner [29, 46] report promising results.

The methods introduced here are based on a general linear representation. The particular case under
study is given by form (4.1); however, the analysis is not restricted to that situation. We introduce particular instances of second- and third-order GL methods with global error estimation. The error estimates can be used for error control; however, in this study we do not address this issue. The GL representation for global error estimation introduced in this study allows stages to be reused or shared among the solutions that are propagated within. This leads to methods having lower costs. Moreover, the stability analysis is simplified when compared with global error strategies that use multiple formulas or equations. We consider only GL methods with two solutions, but this concept can be extended to having multiple values propagated in time (e.g., multistep-multistage or peer methods).

We provide several numerical experiments in which we illustrate the behavior of the global error estimation procedures introduced here, their convergence behavior, and their stability properties.

We investigate nonstiff differential equations. Additional constraints are necessary in order to preserve error estimation and avoid order reduction necessary for stiff ODEs [72, 73].

Global error estimation is typically not used in practice because of its computational expense. This study targets strategies that would make it cheaper to estimate the global errors and therefore make them more practical.

References


URL http://www.mcs.anl.gov/petsc


URL http://dx.doi.org/10.1007/BFb0093147

URL http://imajna.oxfordjournals.org/content/33/1/136.abstract


URL http://dx.doi.org/10.1007/BFb0064894


URL http://dx.doi.org/10.1137/110845100


