A Rooted-Tree Based Derivation of ROW-Type Methods with Non-Exact Jacobian Entries for Index-One DAEs

Tim Jax

Dissertation submitted to
the Faculty of Mathematics and Natural Sciences
at the University of Wuppertal
in partial fulfillment of the requirements for the degree of a

Doctor of Natural Sciences
(Dr. rer. nat.)
in
Applied Numerical Mathematics

Supervisors: Prof. Dr. Michael Günther (BUW)
Prof. Dr. Gerd Steinebach (HBRS)

November 2019

The PhD thesis can be quoted as follows:

urn:nbn:de:hbz:468-20200612-103531-4
[http://nbn-resolving.de/urn/resolver.pl?urn=urn%3Anbn%3Ade%3Ahbz%3A468-20200612-103531-4]

DOI: 10.25926/e36a-vq51
[https://doi.org/10.25926/e36a-vq51]

»I first met Gerhard Wanner in 1970 and, that year, I also met Arne Wolfbrandt. Further developments of Rosenbrock and ROW methods and the possibility of applications to DAEs could not even be dreamed about at that time.«

J. C. Butcher, 26th January 2018
(via personal e-mail communication)

Declaration

I declare that the given thesis was completed independently by myself using no other than the aids and tools specified. All notions that were directly or indirectly adopted from external references are identified as such. I assure that there is no match of the given contents with other theses realized during my previous studies. I have not applied for a doctor's degree in the given doctoral subject at other university departments and do not hold a corresponding doctor's degree.

Bad Honnef, 25th November 2019, Tim Jax

Abstract

Solving differential-algebraic equations (DAEs) efficiently by means of appropriate numerical schemes for time-integration is an ongoing topic in applied mathematics. In this context, especially when considering large systems that occur with respect to many fields of practical application effective computation becomes relevant. In particular, corresponding examples are given when having to simulate network structures that consider transport of fluid and gas or electrical circuits.

Due to the stiffness properties of DAEs, time-integration of such problems generally demands for implicit strategies. Among the schemes that prove to be an adequate choice are linearly implicit Rung-Kutta methods in the form of Rosenbrock-Wanner (ROW) schemes. Compared to fully implicit methods, they are easy to implement and avoid the solution of non-linear equations by including Jacobian information within their formulation. However, Jacobian calculations are a costly operation. Hence, necessity of having to compute the exact Jacobian with every successful time-step proves to be a considerable drawback.

To overcome this drawback, a ROW-type method is introduced that allows for non-exact Jacobian entries when solving semi-explicit DAEs of index one. The resulting scheme thus enables to exploit several strategies for saving computational effort. Examples include using partial explicit integration of non-stiff components, utilizing more advantageous sparse Jacobian structures or making use of time-lagged Jacobian information. In fact, due to the property of allowing for non-exact Jacobian expressions, the given scheme can be interpreted as a generalized ROW-type method for DAEs. This is because it covers many different ROW-type schemes known from literature.

To derive the order conditions of the ROW-type method introduced, a theory is developed that allows to identify occurring differentials and coefficients graphically by means of rooted trees. Rooted trees for describing numerical methods were originally introduced by J.C. Butcher. They significantly simplify the determination and definition of relevant characteristics because they allow for applying straightforward procedures. In fact, the theory presented combines strategies used to represent ROW-type methods with exact Jacobian for DAEs and ROW-type methods with non-exact Jacobian for ODEs. For this purpose, new types of vertices are considered in order to describe occurring non-exact elementary differentials completely. The resulting theory thus automatically comprises relevant approaches known from literature. As a consequence, it allows to recognize order conditions of familiar methods covered and to identify new conditions.

With the theory developed, new sets of coefficients are derived that allow to realize the ROW-type method introduced up to orders two and three. Some of them are constructed based on methods known from literature that satisfy additional conditions for the purpose of avoiding effects of order reduction. It is shown that these methods can be improved by means of the new order conditions derived without having to increase the number of internal stages. Convergence of the resulting methods is analyzed with respect to several academic test problems. Results verify the theory determined and the order conditions found as only schemes satisfying the order conditions predicted preserve their order when using non-exact Jacobian expressions.

Zusammenfassung

Die effiziente Lösung differential-algebraischer Gleichungen (DAEs) mittels geeigneter numerischer Verfahren zur zeitlichen Integration ist ein anhaltendes Thema in der angewandten Mathematik. In diesem Zusammenhang wird eine effektive Berechnung insbesondere im Fall zu betrachtender großer Systeme relevant, die in zahlreichen Feldern der praktischen Anwendung vorkommen. Beispiele hierfür ergeben sich vor allem bezüglich der Simulation von Netzwerk-Strukturen, die den Transport von Fluiden und Gasen oder elektrische Schaltungen betrachten.

Bedingt durch die Steifheits-Eigenschaften von DAEs erfordert die Zeitintegration solcher Probleme grundsätzlich implizite Methoden. Zu den Verfahren die sich als eine geeignete Wahl erweisen zählen linear-implizite Runge-Kutta Methoden in der Form von Rosenbrock-Wanner (ROW) Verfahren. Im Vergleich zu voll-impliziten Methoden sind sie einfach zu implementieren und vermeiden eine Lösung nicht-linearer Gleichungen, indem sie die Jacobi-Matrix in ihrer Formulierung berücksichtigen. Allerdings ist die Berechnung der Jacobi-Matrix eine teure Operation. Daher erweist sich die Notwendigkeit der Ermittlung der exakten Jacobi-Matrix mit jedem erfolgreichen Zeitschritt als ein großer Nachteil.

Um diesem Nachteil entgegen zu wirken wird ein ROW-Typ Verfahren vorgestellt, das für die Berechnung semi-expliziter DAEs vom Index eins die Verwendung nicht-exakter Einträge in der Jacobi-Matrix erlaubt. Das resultierende Verfahren ermöglicht es somit verschiedene Strategien zur Reduzierung des Rechenaufwands auszunutzen. Hierzu zählt unter anderem die Verwendung partieller expliziter Integration nicht-steifer Anteile, der Einsatz vorteilhafterer dünn besetzter Strukturen der Jacobi-Matrix oder die Nutzung zeitverzögerter Informationen. In der Tat kann das beschriebene Verfahren aufgrund der Eigenschaft einer Betrachtung nicht-exakter Jacobi-Matrizen als eine verallgemeinerte ROW-Typ Methode für DAEs interpretiert werden. Dies ist darauf zurück zu führen, dass es zahlreiche verschiedene, aus der Literatur bekannte ROW-Typ Verfahren beinhaltet.

Um die Ordnungsbedingungen der vorgestellten ROW-Typ Methode herzuleiten wird eine Theorie entwickelt, die eine grafische Identifizierung auftretender Differentiale und Koeffizienten mittels Wurzelbäume erlaubt. Wurzelbäume zur Beschreibung numerischer Methoden wurden ursprünglich von J.C. Butcher eingeführt. Sie vereinfachen die Bestimmung und Definition relevanter Eigenschaften erheblich, weil sie die Anwendung unkomplizierter Prozeduren ermöglichen. In der Tat vereint die vorgestellte Theorie Strategien, die zur Darstellung von ROW-Typ Methoden mit exakter Jacobi-Matrix für DAEs und ROW-Typ Methoden mit nicht-exakter Jacobi-Matrix für ODEs geläufig sind. Zu diesem Zweck werden neue Knotentypen berücksichtigt um auftretende nicht-exakte Differentiale vollständig zu beschreiben. Die resultierende Theorie umfasst somit automatisch relevante, aus der Literatur bekannte Ansätze. In der Folge ermöglicht sie es Ordnungsbedingungen enthaltener bekannter Methoden zu erkennen und neue Bedingungen zu ermitteln.

Mit der entwickelten Theorie werden neue Koeffizientensätze hergeleitet, die es erlauben die vorgestellte ROW-Typ Methode bis zur Ordnung zwei und drei zu realisieren. Einige von ihnen sind auf Basis von aus der Literatur bekannten Methoden konstruiert, die Zusatzbedingungen zum Zweck der Vermeidung von Effekten der Ordnungsreduktion erfüllen. Es wird gezeigt, dass diese Methoden mittels der neu hergeleiteten Ordnungsbedingungen verbessert werden können ohne die Anzahl interner Stufen erhöhen zu müssen. Die Konvergenz der resultierenden Methoden wird bezüglich verschiedener akademischer Testprobleme analysiert. Die Ergebnisse bestätigen die ermittelte Theorie und die gefundenen Ordnungsbedingungen, da nur jene Verfahren die Ordnung unter Betrachtung nicht-exakter Jacobi-Matrizen erhalten, welche die prognostizierten Ordnungsbedingungen erfüllen.

Acknowledgement

First of all, I like to thank my supervisor Prof. Dr. Gerd Steinebach. Since the beginning of my studies at the Hochschule Bonn-Rhein-Sieg University of Applied Sciences back in 2007 he taught me in the field of mathematics. In this context, he also introduced me to the fascinating world of applied mathematics and numerical analysis. He guided me during my entire academic career and all corresponding theses. I am grateful for his permanent support and helpful comments regarding my research ideas and scientific interests. His commitment made this PhD thesis possible.

Also, I like to thank my supervisor Prof. Dr. Michael Günther at the University of Wuppertal together with his colleagues Prof. Dr. Andreas Bartel and Prof. Dr. Matthias Ehrhardt. I am thankful that they gave me the opportunity of realizing my PhD thesis in the field of applied mathematics despite my engineering background and that they were willing to support the PhD project proposed in cooperation with the Hochschule Bonn-Rhein-Sieg. Their comments and ideas were always of avail and helped a lot to improve the contents of the PhD thesis permanently.

In addition, I like to thank Prof. Dr. Johannes Geilen and Prof. Dr. Rainer Herpers for their support and giving me the opportunity of realizing my PhD at the Hochschule Bonn-Rhein-Sieg. I like to thank the Graduate Institute of the Hochschule Bonn-Rhein-Sieg, too, as their instructive courses helped me a lot to improve my skills, especially with respect to the process of scientific writing.

Additional thanks go to the members and former members at the department for mechanical engineering, electrical engineering and technical journalism at the Hochschulue Bonn-Rhein-Sieg University of Applied Sciences as well as the members and former members of the research group for applied mathematics and numerical analysis at the University of Wuppertal.

Also, I like to thank the colleagues in the scientific community that showed interest in the numerical methods developed and that came up with several helpful remarks such as Dr. Jens Lang, Dr. Joachim Rang, Dr. Helmut Podhaisky, Dr. Severiano González-Pinto, Dr. John C. Butcher and Dr. Ernst Hairer.

With respect to financial support I like to thank the Graduate Institute of the Hochschule Bonn-Rhein-Sieg for granting me a PhD scholarship for the first three years of my research. Also, I like to thank Dr. Geilen and the department for mechanical engineering, electrical engineering and technical journalism at the Hochschulue Bonn-Rhein-Sieg as well as Dr. Günther and the research group for applied mathematics and numerical analysis at the University of Wuppertal for employing me as research associate. In this context, I like to thank Dr. Edmund Kügeler at the German Aerospace Center (DLR) as well.

Last but not least I thank my family and friends for all their support and patience during the past seven years. In particular, I want to thank my mother Carmen who kept me free of ties in many ways and who had always an open ear regarding my worries and doubts.

Contents

1 Introduction

DAE Systems

Computing systems of differential-algebraic equations (DAEs) is an ongoing topic in many fields of applied mathematics. DAEs occur whenever characteristics in subjects such as physics, chemistry or engineering must be expressed via a combination of differential equations and algebraic constraints. Examples range from describing the behavior of moving objects, such as the movement of a mass in a pendulum, to simulating complex applications, such as the transport of fluids in a network structure. Further examples can be found regarding the description of chemical reactions or the computation of electrical networks. Hence, DAEs have a significant meaning in many applications of our everyday life. For a more general overview regarding the topic of DAEs Simeon's retrospective given in [70] can be recommended. Also, established textbooks such as those by Brenan, Campbell and Petzold [5], Hairer and Wanner [24] as well as Strehmel, Weiner and Podhaisky [81] deal with aspects of DAEs in detail.

Some DAEs are considered to be a limiting case of stiff ordinary differential equations (ODEs) because given stiffness properties become infinite [50]. For that reason, many DAE problems can be computed by numerical schemes that are usually applied for solving stiff ODEs [13, 50]. The original idea of solving DAEs by means of schemes originally constructed for ODEs is generally ascribed to Gear regarding his presentation in [12] (see [5, 50]). In this context, due to stability reasons, an effective time-integration generally requires for implicit or linearly implicit methods. Among the schemes preferred are especially multistep methods such as backward differentiation formulas (BDF) which count to the very first methods applied [5, 40, 50, 51, 52].

A BDF method of order five which is commonly used in engineering applications is Petzold's DASSL [81]. However, a problem of multi-step BDF methods is given by the fact that they lose advantageous stability properties when having to realize and apply schemes of higher order. In this context, the maximum order with respect to A-stability of linear multi-step methods is restricted by two. This is a consequence of the second Dahlquist barrier [81].

Besides BDF methods it is possible to apply implicit Runge-Kutta schemes in order to solve stiff systems [24]. However, a significant drawback of implicit Runge-Kutta methods is the necessity of having to solve a system of non-linear equation with every time-step.

ROW-Type Methods

An alternative to using BDF and implicit RK schemes with respect to stiff differential equations is given by the application of ROW-type methods [33, 69, 34]. ROW methods belong to the class of linearly implicit Runge-Kutta schemes which - contrary to implicit Rung-Kutta methods - require for the solution of linear systems only, i.e. they avoid having to solve non-linear equations. This can be generally achieved by including Jacobian information within the formulation of these schemes directly [81].

As a consequence, ROW-type methods are generally much easier to implement [24]. In fact, ROW methods can be interpreted as diagonally implicit Runge-Kutta methods with a fixed number of Newton iterations (one Newton iteration per stage instead of considering iteration until convergence [24]). This leads usually to a loss of non-linear stability properties (B-stability). However, their linear stability properties (A- and L-stability) can be preserved. Hence, they often prove to be more efficient than implicit Runge-Kutta methods [81].

The concept of ROW methods was introduced by Rosenbrock back in 1962/63 considering the solution of ODEs [65]. The idea of replacing the Jacobian by an expression computed with respect to given initial values for the purpose of having to compute it only once per successful time-step was suggested by Calahan in 1968 (i.e. using $f_y(y_0)$ instead of $f_y(g_i)$ with $g_i = y_0 + \sum(\alpha_{ij} k_j)$) [24]. Further, Wanner introduced further coefficients in 1977 [89, 81].

A derivation of the order conditions required for realizing ROW-type methods can be realized using Butcher's theory of rooted trees that (inspired by the works of Merson [45]) was originally introduced with respect to explicit one-step Runge-Kutta schemes for ODEs [6, 7]. It allows for a convenient graphical representation of occurring elementary differentials and coefficients. A more detailed overview regarding the meaning of rooted trees with respect to deriving Runge-Kutta methods and the important role of Butcher regarding their derivation can be found in [8] and [44]. An extension of ROW methods to DAE problems assumed to be of index one was introduced by Roche in 1988 [64]. In this context, Roche extended the graphical representation via rooted trees by including a second vertex in order to describe components which are related to occurring algebraic components.

ROW-Type Methods for DAEs in Application

With respect to the solution of DAE systems, ROW-type methods prove to be a good choice when having to simulate network structures. In this context, computing the flow behavior of fluids within structures of open channels and closed pipes is among familiar fields of application. For example, ROW methods are used for the purpose of computing river flow in combination with the transport of sediments and soluble substances in [63] and [73]. In [26] ROW-type methods are used in order to analyze the effects of different strategies for space discretization when calculating river hydraulics. In [74] ROW-type methods are considered to examine the influence of different coupling mechanisms on performance and accuracy with respect to river flow simulations. An application of ROW-type methods for the purpose of computing pipe flow problems is described in [75]. Here, a network for water supply including tanks, pumps and valves is regarded.

Modeling equations for computing pipe and channel flow problems are generally given by hyperbolic conservation laws of mass and momentum. In network simulations, corresponding partial differential equations (PDEs) are supplemented by algebraic constraints in order to describe the mutual coupling of given single reaches. Also, depending on the network structure and properties given, additional algebraic equations are required for the purpose of modeling possibly occurring pump and valve components. Thus, the resulting mathematical problem then corresponds to a system of partial differential-algebraic equations (PDAEs). In general, such systems are solved by means of spatial semi-discretization using the method of lines (i.e. occurring partial derivatives in space are approximated via appropriate strategies such as finite differences). This strategy leads to a large DAE problem that, finally, can be solved by appropriate integration schemes. With respect to demands given in order to solve such problems, ROW-type methods proved to be quite effective, robust and stable [73].

A recent example of integrating ROW-type methods into the solution of DAE problems that describe network structures for fluid flow is given by the research project EWave [53]. EWave was funded by the German Federal Ministry of Education and Research (BMBF). Objective of this project was to realize a decision support management system for applications in water supply. In this context, water works should be provided by operating schemes for controllable components given within the network such as pumps and valves. These operation plans were computed via optimization programs that work on the basis of demand forecasts and simulation results given with respect to the network structure considered. However, the network structure of pressure zones considered as prototype region for realizing the EWave project counted up to 15.000 pipe elements [32]. Hence, in order to support the water works with operation plans in time (especially when taking into account the fact that there must be additional time considered after the network simulation for running subsequent optimization tools in addition) flow conditions within these pipes needed to be computed as fast as possible. For this purpose, the ROW-type method RODASP [72, 76] was applied and proved to be very effective.

Aside from computing the behavior of fluids within network structures, the application of ROW-type schemes to DAEs proved to be beneficial in other fields of practical interest. For example, in [20] examples are given where ROW-type methods are applied in order to compute DAEs with respect to electrical circuit simulations. More recently, the application of ROW-type methods with respect to DAE systems that result when solving quasi-static problems in solid mechanics is considered in [25].

Non-Exact Jacobian Expressions

Despite their useful properties, a general drawback of ROW-type methods is given by the necessity of having to compute the exact Jacobian with every successful time-step. This is because determining the Jacobian numerically corresponds to an expensive operation, especially when regarding the solution of large systems [24, 81]. In order to overcome this problem, attempts can be found in literature that allow for ROW-type methods with non-exact Jacobian entries. In this context, Steihaug and Wolfbrandt derived so-called W methods in [71]. W methods correspond to ROW-type schemes for ODEs that allow to replace the original Jacobian $\partial f/\partial y$ by arbitrary expressions. However, the realization of corresponding methods of higher order requires to satisfy large numbers of conditions. Therefore finding sets of coefficients becomes challenging easily. One of the very few publications that deals with high-order W methods is given by [55] where W methods of order four with six internal stages are presented.

To decrease the number of order conditions required, ROW-type schemes that are characterized by special choices of the Jacobian approximations are often considered. In this context, suitable approximations such as representations of the Jacobian close to the exact Jacobian are generally preferred in order to ensure stability [24, 67]. For this purpose, reusing the Jacobian of a previous time-step (i.e. applying time-lagged Jacobian information) is among the strategies preferred. Reusing the Jacobian is generally attributed to approximations of the form $\partial f/\partial y + \mathcal{O}(h)$ [34]. It allows to reduce computational effort as well as the number of order conditions required [81]. Corresponding schemes were analyzed by Verwer et al. in [86, 87], Kaps and Ostermann [34] or by Novati [47].

Note that there have been further realizations of ROW-type methods for ODEs that exploit the possibility of non-exact or non-complete Jacobian representations in order to reduce computational efforts. For example, Krylov techniques were applied to ROW-type methods in [67] and [90]. They allow to realize so-called matrix-free schemes that avoid explicit computation of the full Jacobian and to exploit sparsity more advantageous in order to the solution of occurring linear systems [47, 90]. More recently, ROW-Type schemes combined with Krylov techniques have been studied by Tranquilli in [84] and [85].

Besides the application of Krylov techniques making use of approximate matrix factorization (AMF) is a common practice, especially when considering the solution of (time-dependent) PDEs. The general idea of AMF is to split the system matrix occurring in linear systems to be solved into a sum of matrices. Compared to the original system matrix the matrices resulting after this splitting might have a more advantageous structure. The system matrix then is replaced by an approximation based on this sum [27]. The application of AMF to ROW-type methods has been recently studied especially by González-Pinto et al. [14, 15, 17, 18]. Note that a general comparison of AMF and Krylov techniques is presented in [4].

The idea of AMF can also be applied to ROW-type methods in order to realize a splitting into matrices that refer to stiff and non-stiff components. In this context, by neglecting the matrices that correspond to non-stiff elements the linearly implicit ROW-type method is reduced to an underlying explicit Runge-Kutta scheme. In fact, corresponding realizations can be interpreted as a combination of (linearly) implicit and explicit strategies, i.e. an IMEX approach [28]. IMEX is generally applied in the context of Runge-Kutta methods when having to solve systems characterized by a group of non-stiff and a group of stiff components. A special representation of such strategies are partitioned ROW-type methods such as those presented by Rentrop in [61]. Here, a special choice of equations given within an ODE system is interpreted to be non-stiff and, therefore, solved via an explicit Runge-Kutta formulation while the rest of equations is interpreted to be stiff and, therefore, solved via an linearly implicit ROW formulation.

However, with respect to DAEs, ROW-type schemes that exploit strategies for using non-exact Jacobian expressions are rare. In fact, one of the very few publications in this context is given by Rentrop et al. in [62]. Here, a partitioned scheme is realized that solves given differential equations explicitly via the underlying Runge-Kutta scheme while algebraic constraints are computed linearly implicit. In several works Strehmel and Weiner consider linearly implicit Runge-Kutta schemes and ROW-type methods that can be applied to DAEs and that allow for special choices of given Jacobian entries [77, 79]. Finally, in [56] Rang derives ROW-type methods for DAEs of index one that also include the order conditions introduced by Steihaug and Wolfbrandt with respect to W methods for ODEs [71].

Motivation

Especially when having to compute large DAE systems, given options to save computational efforts via the use of non-exact Jacobian expressions seem restricted compared to the possibilities given by ROW-type methods for ODEs that allow for arbitrary Jacobian approximations. In fact, there seems to be no analysis regarding the requirements that must be considered when using arbitrary Jacobian entries in the context of DAE systems. However, such methods could be promising as they might allow to adapt solution strategies with respect to occurring stiffness properties within the given system more appropriately. An example could be given by hyperbolic PDEs that are characterized by stiff source terms [39]. When such problems are combined with algebraic constraints, the solution after spatial semi-discretization yields a DAE system with stiff components theoretically given just by the source terms and the algebraic equations. Hence, a ROW-type scheme that combines properties of IMEX and partitioning strategies in order to reduce the linearly implicit solution to these components could be advantageous. In fact, ROW-type methods for DAEs that allows for arbitrary Jacobian approximations should be the most adaptive and, therefore, should offer some promising characteristics for the purpose of saving computational efforts effectively.

First attempts of applying a corresponding scheme were presented at the beginning of the given PhD project in [29]. However, back then, the requirements for realizing and applying such a scheme were not clear. It took some time to recognize that the excellent paper by Rang [56] just considered a combination of well-known order conditions while the great paper by Rentrop et al. [62] introduced new (coupling) conditions. As a consequence, realizing a ROW-type method for DAEs that allows for arbitrary Jacobian entries must lead to additional conditions that were not described in literature so far. First results that seem to verify this supposition were published as part of the given thesis in [30] and [31]. Here, order conditions required when taking into account arbitrary Jacobian entries with respect to differential parts of index-one DAE systems were identified. For this purpose, a theory based on rooted trees was considered.

The objective of the given dissertation is to supplement the results presented in [30] by considering additional non-exact Jacobian entries with respect to algebraic parts. In this context, a generalized ROW-type method is introduced that allows for realizing different ROW-type schemes known from literature by choosing appropriate Jacobian approximations. In order to describe characteristics of the resulting method, an extended theory is presented that allows to identify the order conditions required for implementation. For this purpose, rooted trees are constructed that enable to identify occurring differentials and coefficients as well as order conditions graphically. This is not only because rooted trees significantly ease the process of identifying properties and order conditions. It is because a corresponding theory with respect to ROW-type methods for DAEs is missing in literature so far. In fact, the last comprehensive analysis was given by Roche in [64] when defining ROW-type methods with exact Jacobian for the solution of DAEs. In this context, a major objective when realizing a corresponding theory is given by the challenge of preserving all the properties and approaches known for schemes given in literature already. This means, procedures described for example by Hairer and Wanner in [24] should still be recognizable and valid.

2 Preliminaries

This section presents general aspects and properties of a ROW-type method that is characterized by non-exact Jacobian entries when applied to index-one DAE systems. In this context, relevant issues with respect to semi-explicit DAEs assumed to be of index one are repeated first. Afterwards, the formulation of the ROW-type scheme considered for computing this class of problems is introduced. Finally, features of this method and options for saving computational efforts are discussed.

2.1 Problem Formulation

The subsequently developed and analyzed ROW-type scheme is focused on solving semi-explicit DAE systems of index one. Regarding the Kronecker canonical form, the general formulation of such DAE systems reads

$$y'(x) = f(y(x), z(x)), \qquad y(x_0) = y_0 \tag{2.1a}$$

$$0 = g(y(x), z(x)), \qquad z(x_0) = z_0 \tag{2.1b}$$

with initial values y_0, z_0 assumed to be consistent, i.e. $g(y_0, z_0) = 0$. Equations (2.1a) and (2.1b) represent differential and algebraic parts. They are characterized by $y : \mathbb{R} \to \mathbb{R}^{n_y}$, $f : \mathbb{R}^{n_y} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_y}$ and $z : \mathbb{R} \to \mathbb{R}^{n_z}$, $g : \mathbb{R}^{n_y} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_z}$ where given domains are allowed to be subsets of $\mathbb{R}$. Functions f and g are considered to be sufficiently differentiable.

In order to ensure that the system has (differential) index one the inverse of partial derivatives $g_z(y(x), z(x))$ is assumed to be existent and bounded in neighborhood of the exact solution of (2.1). As a consequence, (2.1b) yields a locally unique solution of the form $z(x) = G(y(x))$ due to the implicit function theorem. Hence, by inserting this expression into (2.1a) an ordinary differential system of the form $y'(x) = f(y(x), G(y(x)))$ can finally be generated [24, 64, 81].

Remark 2.1

- *In general, differential function f within (2.1a) might allow for applying some additive splitting into non-stiff components f_N and stiff components f_S such that*

$$f = f_N + f_S. \tag{2.2}$$

This assumption also includes the possibility of partitioning the differential part given into non-stiff and stiff equations by regarding corresponding vectorized components [30, 81].

- *Problem formulation (2.1) corresponds to the autonomous case, i.e. the system is assumed to be independent of explicit expressions x. This restriction is given without loss of generality as explicit dependencies on x can be taken into account by including the equation $x' = 1$ [64].*

2.2 Scheme Formulation

For solving the semi-explicit index-one DAE system (2.1) efficiently, an altered formulation of the original linearly implicit ROW scheme for DAEs introduced by Roche [64] is considered below. The modified ROW-type method reads:

$$y_1 = y_0 + \sum_{i=1}^{s} b_i k_i, \qquad z_1 = z_0 + \sum_{i=1}^{s} b_i k_i^{alg} \tag{2.3a}$$

$$\begin{pmatrix} k_i \\ 0 \end{pmatrix} = h \begin{pmatrix} f(v_i, w_i) \\ g(v_i, w_i) \end{pmatrix} + h \sum_{j=1}^{i} \gamma_{ij} \begin{bmatrix} A_y & A_z \\ B_y & B_z \end{bmatrix} \begin{pmatrix} k_j \\ k_j^{alg} \end{pmatrix} \tag{2.3b}$$

with

$$v_i = y_0 + \sum_{j=1}^{i-1} \alpha_{ij} k_j, \qquad w_i = z_0 + \sum_{j=1}^{i-1} \alpha_{ij} k_j^{alg}. \tag{2.3c}$$

As for classical ROW methods, y_1 and z_1 denote numerical solutions with respect to given differential and algebraic parts at $x_1 = x_0 + h$, i.e. after a single time-step of size h. Terms k_i and k_i^{alg} correspond to increments. The number of internal stages is defined by parameter s. Weights b_i and coefficients α_{ij} as well as γ_{ij} are real numbers for $i = 1, ..., s$ and $j = 1, ..., i$. As usual for ROW-type methods, $\alpha_{ii} = 0$ and $\gamma_{ii} = \gamma$ are used. This way, increments k_i and k_i^{alg} are computed via (2.3b) by solving a linear system for each $i = 1, ..., s$ with equal system matrices. After dividing the second line of (2.3b) by h these system matrices read:

$$\begin{bmatrix} I - \gamma_{ii} h A_y & -\gamma_{ii} h A_z \\ -\gamma_{ii} B_y & -\gamma_{ii} B_z \end{bmatrix}.$$

As a consequence, determining increments requires just one single LU-decomposition within every step [64, 80].

In contrast to the original ROW method for DAEs defined in [64] that is restricted to exact Jacobian representations, formulation (2.3) is characterized by replacing original Jacobian entries $(f_y)_0$, $(f_z)_0$, $(g_y)_0$, $(g_z)_0$ by approximations A_y, A_z, B_y, B_z, respectively. In this context, sub-matrices A_y, A_z and B_y are assumed to be arbitrary. B_z, however, is assumed to be an arbitrary but regular sub-matrix. Regularity of B_z is required to enable resolving for k_i^{alg} when $h \to 0$. Nevertheless, especially for stability reasons A_y, A_z and B_y, B_z should be approximations that are related to the original entries $(f_y)_0$, $(f_z)_0$ and $(g_y)_0$, $(g_z)_0$ rather than arbitrary values.

Remark 2.2 *Note that Strehmel and Weiner studied linearly implicit Runge-Kutta methods for singularly perturbed problems and DAEs that theoretically cover the formulation of ROW-type scheme (2.3) [77, 78, 79, 80]. In this context, method (2.3) is introduced explicitly in [80]. However, there are several distinguishing features regarding contents of these references compared to the analyses presented in subsequent sections. For example, in [80] there is no detailed derivation of order conditions or sets of coefficients required to apply method (2.3). In fact, in their works Strehmel and*

Weiner generally do not consider schemes that are based on arbitrary Jacobian entries. They usually consider the cases

- $A_y = (f_y)_0 + \mathcal{O}(h)$, $A_z = (f_z)_0 + \mathcal{O}(h)$, $B_y = (g_y)_0 + \mathcal{O}(h)$, $B_z = (g_z)_0 + \mathcal{O}(h)$,
- $A_y = 0$, $A_z = 0$, $B_y = (g_y)_0 + \mathcal{O}(h)$, $B_z = (g_z)_0 + \mathcal{O}(h)$,
- $A_y = 0$, $A_z = 0$, $B_y =$ *arbitrary (especially* $B_y = 0$*)*, $B_z = (g_z)_0 + \mathcal{O}(h)$

in order to describe relevant characteristics. For that reason, a detailed description of order conditions that result when including arbitrary components A_y *and* A_z *is generally missing. Also, strategies to identify order conditions by means of graphical representation via rooted trees are not considered. Corresponding aspects, however, are taken into account below.*

2.3 Properties

ROW-type method (2.3) is characterized by allowing for non-exact Jacobian entries when solving semi-explicit index-one DAE systems. Hence, it could be interpreted as a W method for DAEs. W methods as originally introduced by Steihaug and Wolfbrandt [71] correspond to ROW-type methods for ODEs that allow for non-exact, arbitrary Jacobians by satisfying additional order conditions. In fact, Strehmel and Weiner refer to formulation (2.3) as a W method in [80]. However, contrary to W methods for ODEs, the ROW-type method presented is not allowed to consider arbitrary Jacobian entries in general. At least sub-matrix B_z is restricted and needs to be regular. Moreover, realizable versions of the scheme (2.3) seem to require for special choices of B_z that are close to the original Jacobian entry $(g_z)_0$. For that reason, method (2.3) less correlates with a W scheme regarding its original meaning. Instead, it should better be referred to as a generalized ROW-type method for DAEs. This is because given possibilities for choosing its Jacobian entries make it a class that covers a wide range of ROW-type schemes for ODEs and DAEs known from literature. A detailed list of most relevant schemes into which the given method turns automatically by choosing sub-matrices A_y, A_z, B_y and B_z appropriately is presented in Table 2.1 regarding the ODE case and Table 2.2 regarding the DAE case.

Among schemes covered by ROW-type method (2.3) is the standard formulation of explicit Runge-Kutta methods for ODEs and the original ROW method for ODEs as described by Hairer and Wanner [23, 24], the W method for ODEs introduced by Steihaug and Wolfbrand [71] as well as the ROW method for DAEs presented by Roche [64]. Further schemes included are especially characterized by applying partial explicit and partial implicit strategies in order to solve components assumed to be non-stiff and stiff effectively. Corresponding schemes combine explicit Runge-Kutta formulations and linearly implicit ROW formulations after separating components assumed to be non-stiff and stiff via appropriate partitioning and splitting techniques. Among the schemes regarding partitioning strategies for DAEs are the methods introduced by Rentrop, Roche and Steinebach [62] as well as Strehmel and Weiner [77, 79]. They generally assume given differential parts to be completely non-stiff and, thus, integrate

Table 2.1: Schemes covered by the generalized ROW-type method (ODE case).

Problem	A_y	A_z	B_y	B_z	Ref.
$y' = f(y)$	0	/	/	/	[23]
$y' = f(y)$	$(f_y)_0$	/	/	/	[24]
$y' = f(y)$	$f_y + \mathcal{O}(h)$	/	/	/	[34]
$y' = f(y)$	A_y	/	/	/	[71]
$y' = f_N(y) + f_S(y)$	$(f_S)_y$	/	/	/	[28]
$\begin{pmatrix} y'_N \\ y'_S \end{pmatrix} = \begin{pmatrix} f_N(y_N, y_S) \\ f_S(y_N, y_S) \end{pmatrix}$	$\begin{bmatrix} 0 & 0 \\ 0 & (f_S)_{y_S} \end{bmatrix}$	/	/	/	[61]

Table 2.2: Schemes covered by the generalized ROW-type method (DAE case).

Problem	A_y	A_z	B_y	B_z	Ref.
$y' = f(y, z)$ $0 = g(y, z)$	$(f_y)_0$	$(f_z)_0$	$(g_y)_0$	$(g_z)_0$	[64]
$y' = f(y, z)$ $0 = g(y, z)$	0	0	$(g_y)_0$	$(g_z)_0$	[62]
$y' = f(y, z)$ $0 = g(y, z)$	$f_y + \mathcal{O}(h)$	$f_z + \mathcal{O}(h)$	$g_y + \mathcal{O}(h)$	$g_z + \mathcal{O}(h)$	[78, 79]
$y' = f(y, z)$ $0 = g(y, z)$	0	0	$g_y + \mathcal{O}(h)$	$g_z + \mathcal{O}(h)$	[78, 79]
$y' = f(y, z)$ $0 = g(y, z)$	0	0	B_y	$g_z + \mathcal{O}(h)$	[77]
$y' = f(y, z)$ $0 = g(y, z)$	0	0	0	$g_z + \mathcal{O}(h)$	[77]
$y' = f(y, z)$ $0 = g(y, z)$	A_y	A_z	$(g_y)_0$	$(g_z)_0$	[30]
$y' = f_N(y, z) + f_S(y, z)$ $0 = g(y, z)$	$(f_S)_y$	f_z	$(g_y)_0$	$(g_z)_0$	[30]
$y' = f_N(y, z) + f_S(y, z)$ $0 = g(y, z)$	$(f_S)_y$	$(f_S)_z$	$(g_y)_0$	$(g_z)_0$	[29]

them via underlying explicit Runge-Kutta formulations. Corresponding schemes are realized within the generalized method (2.3) by setting Jacobian entries A_y and A_z to zero. This way, the linearly implicit ROW-type formulation automatically turns into an explicit Runge-Kutta formulation for given differential parts.

Most of the methods listed in Table 2.1 and Table 2.2 enable to reduce computational efforts by applying such strategies of partial explicit and partial implicit solution. In this context, extended explicit integration of non-stiff components by underlying Rung-Kutta formulations reduces the extent of costly implicit integration. It minimizes the number of given Jacobian entries and, thus, speeds up computations by exploiting sparser Jacobian structures. However, corresponding strategies can only be applied when stiffness properties of occurring components are known in advance or when there are appropriate applications for automatic stiffness detection implemented. In cases where stiffness cannot be predicted or determined an implicit solution is inevitable. Fortunately, computational efforts going along with required implicit solutions can still be reduced when considering ROW-type methods. By satisfying additional order conditions it is possible to apply time-lagged Jacobians, i.e. the information of a previously computed Jacobian is reused for several time-steps. This strategy can be interpreted as a compromise between having to compute the exact Jacobian with every time-step or regarding arbitrary Jacobian entries (especially for setting most of them equal to zero). In fact, even the number of order conditions that must be satisfied ranges in between the number of conditions required for applying an exact Jacobian and the number of conditions required for applying a non-exact, arbitrary Jacobian. With respect to DAEs, corresponding ROW-type versions have been analyzed by Strehmel and Weiner in [79, 80] assuming $A_y = (f_y)_0 + \mathcal{O}(h)$, $A_z = (f_z)_0 + \mathcal{O}(h)$, $B_y = (g_y)_0 + \mathcal{O}(h)$, $B_z = (g_z)_0 + \mathcal{O}(h)$.

Via generalized method (2.3), all these schemes and their strategies for reducing computational efforts are now combined within one single, unifying ROW-type class. Moreover, they can be easily realized just by choosing given Jacobian entries appropriately. In this context, when solving DAEs, the possibility of regarding arbitrary Jacobian entries A_y and A_z with respect to differential parts enables to exploit advantages of partial explicit and partial implicit integration more efficient than it is the case for classical methods listed in Table 2.2. As classical methods generally assume differential parts to be completely non-stiff or completely stiff, they lack the opportunity of using partial implicit integration when there are just few stiff components present. They usually consider the cases $A_y = 0$, $A_z = 0$ for overall explicit solution and $A_y = (f_y)_0$, $A_z = (f_z)_0$ or $A_y = (f_y)_0 + \mathcal{O}(h)$, $A_z = (f_z)_0 + \mathcal{O}(h)$ for overall implicit solution of the differential parts. However, when assuming arbitrary components A_y and A_z, advantages of much sparser Jacobian structures can be taken into account in order to speed up computations. In fact, it enables to apply processes of partitioning and splitting to all components of the differential parts freely and, thus, more flexible. Corresponding versions of ROW-type methods for DAEs have been studied as part of the given thesis first. Results were published in [30] assuming exact Jacobian entries with respect to given algebraic parts, i.e. $B_y = (g_y)_0$ and $B_z = (g_z)_0$.

Different methods covered according to Table 2.1 and Table 2.2 had been derived in literature just with respect to the special choices of Jacobian entries listed. For that reason, there is a general lack regarding a generalized and unifying theory that clarifies their mutual relationships. In fact, considering the common graphical derivation of order conditions via rooted trees, this disadvantage is shown in several references by repeatedly used types of vertices (usually meager and fat nodes) that have different meanings. As a consequence, it becomes difficult to apply these strategies for deriving order conditions to methods that combine different properties of the schemes defined in literature. For that reason, subsequent sections introduce a new theory for finding order conditions with respect to the generalized ROW-type method (2.3) graphically via rooted trees. The resulting theory includes and unifies correspondingly restricted methods described within most of the references listed in Table 2.1 and Table 2.2.

Remark 2.3 *Note that the theory presented in subsequent sections is restricted to the case using exact Jacobian entries $B_z = (g_z)_0$. An extension to non-exact components of B_z especially by assuming $B_z = (g_z)_0 + \mathcal{O}(h)$ has been derived during the realization of the thesis given. However, its representation proves to be quite extensive and, therefore, will not be detailed below.*

3 Taylor Series via Rooted Trees

To derive a unifying theory that defines order conditions of the generalized ROW-type method (2.3) Taylor expansions of the exact and the numerical solution must be formulated. In general, they can be derived graphically using an algebraic theory based on rooted trees. Within this section, strategies for constructing these trees are introduced. Also, details for identifying corresponding elementary differentials and coefficients are presented. In this context, Section 3.1 repeats the approach for defining Taylor series with respect to the analytical solution. Section 3.2 then introduces the approach for defining Taylor series with respect to the numerical solution.

3.1 Exact Solution

In order to describe Taylor series with respect to the exact solution of DAE system (2.1) the algebraic theory based on rooted trees presented by Roche in [64] and reformulated by Hairer and Wanner in [24] is considered below. For this purpose, derivatives of the exact solution are presented first. Afterwards, rooted trees that represent occurring elementary differentials are constructed. Finally, the Taylor series with respect to the analytical solution is defined. As detailed in Section 3.2, corresponding results form a basis for describing Taylor expansions with respect to the numerical solution by method (2.3).

3.1.1 Derivatives

In order to determine the derivatives with respect to exact solutions of (2.1), algebraic parts according to (2.1b) are differentiated first. This yields $0 = g_y y' + g_z z'$ and thus by considering $y' = f$ [24]:

$$z' = (-g_z^{-1})g_y f. \tag{3.1}$$

As described in [64, 24], derivatives of DAE system (2.1) then result from differentiating (2.1a) and (3.1) successively with respect to x. For this purpose, the condition

$$(-g_z^{-1})'u = (-g_z^{-1})(g_{zy}((-g_z^{-1})u, f) + g_{zz}((-g_z^{-1})u, (-g_z^{-1})g_y f)). \tag{3.2}$$

is used that follows from $(A^{-1}(x))' = -A^{-1}(x)A'(x)A^{-1}(x)$ and the chain rule [24].

Remark 3.1 *Within (3.2) a common notation is applied: To abbreviate partial derivatives of higher order, they are written in the form of multi-linear mappings. For example [24]:*

$$f_{yz}(u, v) \quad \textit{represents} \quad \sum_{j,k} \frac{\partial^2 f_i}{\partial y_j \partial z_k} \cdot u_j v_k.$$

By applying the procedure described, derivatives regarding the differential parts of (2.1) finally read [24]

$$
\begin{aligned}
y' &= f\\
y'' &= f_y f + f_z(-g_z^{-1})g_y f\\
y''' &= f_{yy}(f,f) + f_y f_y f + f_{yz}(f,(-g_z^{-1})g_y f) + f_{zy}((-g_z^{-1})g_y f, f)\\
&\quad + f_z(-g_z^{-1})g_{yy}(f,f) + f_z(-g_z^{-1})g_y f_y f + f_y f_z(-g_z^{-1})g_y f\\
&\quad + f_z(-g_z^{-1})g_y f_z(-g_z^{-1})g_y f + f_{zz}((-g_z^{-1})g_y f,(-g_z^{-1})g_y f)\\
&\quad + f_z(-g_z^{-1})g_{yz}(f,(-g_z^{-1})g_y f) + f_z(-g_z^{-1})g_{zy}((-g_z^{-1})g_y f, f)\\
&\quad + f_z(-g_z^{-1})g_{zz}((-g_z^{-1})g_y f,(-g_z^{-1})g_y f)\\
y'''' &= ...
\end{aligned}
$$

while derivatives regarding the algebraic parts of (2.1) become [24]

$$
\begin{aligned}
z' &= (-g_z^{-1})g_y f\\
z'' &= (-g_z^{-1})g_{zy}((-g_z^{-1})g_y f, f) + (-g_z^{-1})g_{zz}((-g_z^{-1})g_y f,(-g_z^{-1})g_y f)\\
&\quad + (-g_z^{-1})g_{yy}(f,f) + (-g_z^{-1})g_{yz}(f,(-g_z^{-1})g_y f) + (-g_z^{-1})g_y f_y f\\
&\quad + (-g_z^{-1})g_y f_z(-g_z^{-1})g_y f\\
z''' &= ...
\end{aligned}
$$

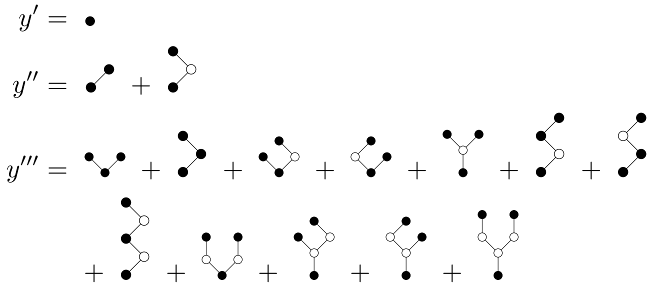
3.1.2 Trees and Differentials

Each summand within the previous differentiations of given differential and algebraic parts represents an exact elementary differential. Obviously, occurring differentials become increasingly complicated for derivatives of higher order. Hence, for convenience, they are expressed graphically by tree structures. These trees must consist of two different types of vertices. For this purpose, Roche [64] considered a meager vertex (•) to represent the differential term f and a fat vertex (○) to visualize the algebraic term $(-g_z^{-1})g$. Derivatives of f and derivatives of g are expressed by a new vertex that is attached to a corresponding meager or fat vertex via a branch. In this context, a branch followed by a new meager vertex corresponds to derivatives with respect to differential component y, a branch followed by a new fat vertex corresponds to derivatives with respect to algebraic component z.

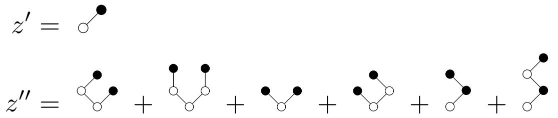
Example 3.1

As shown in [64] and [24], elementary differentials for y', y'' and y''' explicitly ex-

pressed in previous subsection can be described by trees as follows:



Elementary differentials for z' and z'' can be visualized by trees using:



The first element of each tree is called the root. Hence, resulting structures are referred to as rooted trees. In order to describe all rooted trees required for visualizing elementary differentials of the exact solution, Roche [64] defined the set of differential-algebraic rooted trees (DAT).

Definition 3.1 *Let $DAT = DAT_y \cup DAT_z$ ($DAT_y \cap DAT_z = \varnothing$) denote the set of differential-algebraic rooted trees. It is recursively defined by:*

(a) $\tau_y = \bullet \in DAT_y$, $\tau_z =$ $\in DAT_z$

(b) $[t_1, ..., t_m, u_1, ..., u_n]_y \in DAT_y$,
if $t_1, ..., t_m \in DAT_y$, $u_1, ..., u_n \in DAT_z$

(c) $[t_1, ..., t_m, u_1, ..., , u_n]_z \in DAT_z$,
if $t_1, ..., t_m \in DAT_y$, $u_1, ..., u_n \in DAT_z$ *and* $m + n \geq 2$

(d) $[t_1]_z \in DAT_z$,
if $t_1 \in DAT_y$

$[t_1, ..., t_m, u_1, ..., u_n]_{y,z}$ *represent unordered (m+n)-tuples.*

Remark 3.2 *In item a) of Definition 3.1 the tree τ_z is redundant. It is theoretically covered by $[t_1]_z$ in item d) when using $t_1 = \tau_y$ according to item a). However, denoting τ_z explicitly allows to keep consistency with descriptions given in literature such as Hairer and Wanner [24] or Roche [64].*

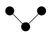
Within Definition 3.1, τ_y and τ_z are the rooted trees that correspond to elementary differentials of y' and z', respectively. They are the basis for constructing further trees of subsets DAT_y and DAT_z. Elements of DAT_y correspond to elementary differentials resulting for derivatives of the differential parts, generally expressed via letter t. Elements of DAT_z correspond to elementary differentials resulting for derivatives of the algebraic parts, generally expressed via letter u. Trees belonging to subset DAT_y are characterized by a meager root. So, they are constructed by attaching a new meager root to given sub-trees t_i and u_i, indicated by $[.]_y$. Trees belonging to subset DAT_z are characterized by a fat root. So, they are constructed by attaching a new fat root to given sub-trees t_i and u_i, indicated by $[.]_z$.

Example 3.2

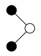
t_1

u_1

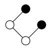
are sub-trees that enable to construct trees such as



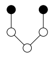
$t = [t_1, t_1]_y$



$t = [u_1]_y$



$u = [u_1, t_1]_z$



$u = [u_1, u_1]_z$

Remark 3.3 *Within the exact solution, there are no elementary differentials existing that include expressions of the form $(-g_z^{-1})g_z$. Hence, there can be no tree structures where a fat vertex is directly followed by another fat vertex via a single branch, i.e. trees of the form $u = [u_1]_z$ are not admissible [24].*

Rooted trees required to describe all the elementary differentials that result for a specific grade of differentiating y or z are characterized by the same number of meager vertices. This property is used to assign a distinct order to every tree.

Definition 3.2 *The order of a tree $t \in DAT_y$ or $u \in DAT_z$ is denoted by $\rho(t)$ or $\rho(u)$, respectively, and corresponds to the number of its meager vertices [64] .*

Remark 3.4 *The order of trees describing elementary differentials for a specific grade of differentiation equals this grade of differentiation.*

Based on Definition 3.1 that describes rooted trees of DAT by recursive construction, corresponding elementary differentials can be identified distinctly via recursive formulation as well [24, 64].

Definition 3.3 *Rooted trees of the set DAT are recursively mapped to elementary differentials $F(t)$ and $F(u)$ as follows:*

(a) $F(\tau_y) = f$, $F(\tau_z) = (-g_z^{-1})g_y f$

(b) $F(t) = \frac{\partial^{m+n} f}{\partial y^m \partial z^n} \left(F(t_1), ..., F(t_m), F(u_1), ..., F(u_n)\right),$
if $t = [t_1, ..., t_m, u_1, ..., u_n]_y \in DAT_y$

(c) $F(u) = (-g_z^{-1}) \frac{\partial^{m+n} g}{\partial y^m \partial z^n} \left(F(t_1), ..., F(t_m), F(u_1), ..., F(u_n)\right),$
if $u = [t_1, ..., t_m, u_1, ..., u_n]_z \in DAT_z$ *and* $m + n \geq 2$

(d) $F(u) = (-g_z^{-1})g_y F(t_1)$,
if $u = [t_1]_z \in DAT_z$

Due to the symmetry of partial derivatives, elementary differentials $F(t)$ and $F(u)$ are well defined. This is because Definition 3.3 is unaffected by permutations of $t_1, ..., t_m, u_1, ..., u_n$ [24]. However, for the same reason some differentials must occur multiple times when deriving the differentiations to formulate Taylor expansions of the exact solution. To determine these elementary differentials completely, the following steps defined by Hairer and Wanner [24] can be applied for constructing corresponding rooted trees.

Procedure 3.1 *Considering all the rooted trees of an order q that belong to the subset DAT_y or DAT_z and thus represent the elementary differentials required to express derivatives of the same order with respect to differential or algebraic parts completely, trees to describe all the elementary differentials for derivatives of order $q+1$ result from*

i) attaching a new branch with τ_y to each given vertex,

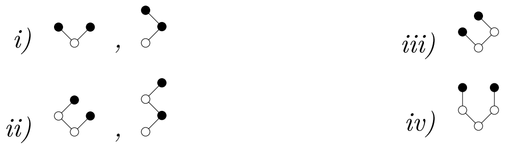
ii) attaching a new branch with τ_z to each given vertex,

iii) splitting each fat vertex into two new fat vertices directly connected by a single branch and attaching another branch with τ_y to the lower of these fat vertices,

iv) splitting each fat vertex into two new fat vertices directly connected by a single branch and attaching another branch with τ_z to the lower of these fat vertices.

Example 3.3

The only tree of DAT_z that regards order $q = 1$ is $z' =$ *. Based on this element, all trees of the set DAT_z required for constructing z'' (i.e. regarding order $q = 2$) of the analytical solution are determined by using the steps i) to iv) of Procedure 3.1. Resulting trees per step read:*



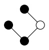
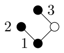
Steps given in Procedure 3.1 follow directly from processes to determine derivatives of y and z. Step *i*) represents differentiation of functions f and g with respect to y and subsequent multiplication of the resulting terms by $y' = f$. Step *ii*) represents differentiation of functions f and g with respect to z and subsequent multiplication of the resulting terms by $z' = (-g_z^{-1})g_y f$. Steps *iii*) and *iv*) represent summands that occur for differentiations of $(-g_z^{-1})$ according to (3.2) [24].

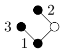
In order to distinguish tree structures that occur multiple times, Roche [64] introduced the concept of labeled trees.

Definition 3.4 *A tree $t \in DAT_y$ or $u \in DAT_z$ is called monotonically labeled if all of its meager vertices are labeled by an integer i with $1 \leq i \leq \rho(t)$ or $1 \leq i \leq \rho(u)$, respectively. Starting from the root, these integers must be monotonically increasing following any branch of the tree considered. The set including all monotonically labeled trees is denoted by $LDAT$, where $LDAT = LDAT_y \cup LDAT_z$.*

Remark 3.5 *Trees occurring multiple times for completely describing a derivative of specific order can be easily determined. They result by permuting integer labels over all meager vertices of trees that are given according to Definition 3.1. However, valid permutations must ensure that labels are monotonically increasing within every branch.*

Example 3.4

 *can only be monotonically labeled using:* 

and 

Hence, this tree occurs twice when deriving y'''.

3.1.3 Taylor Expansion

As the set $LDAT$ includes all trees required to determine derivatives of a specific order, Taylor expansions of the exact solution can be described according to the following Theorem.

Theorem 3.1 *(Roche 1988, [64])*
Derivatives for the exact solution of an index-one semi-explicit DAE system given by (2.1) correspond to

$$y^{(q)}(x_0) = \sum_{\substack{t \in LDAT_y \\ \rho(t)=q}} F(t)(y_0, z_0) = \sum_{\substack{t \in DAT_y \\ \rho(t)=q}} \alpha(t) F(t)(y_0, z_0)$$

$$z^{(q)}(x_0) = \sum_{\substack{u \in LDAT_z \\ \rho(u)=q}} F(u)(y_0, z_0) = \sum_{\substack{u \in DAT_z \\ \rho(u)=q}} \alpha(u) F(u)(y_0, z_0)$$

with $\alpha(t)$ and $\alpha(u)$ the numbers of possible monotonic labellings of a tree in DAT according to Remark 3.5. Hence, Taylor series of the exact solution read

$$y(x_0+h) = \sum_{t \in LDAT_y} F(t)(y_0, z_0) \cdot \frac{h^{\rho(t)}}{\rho(t)!} \quad \text{and} \quad z(x_0+h) = \sum_{u \in LDAT_z} F(u)(y_0, z_0) \cdot \frac{h^{\rho(u)}}{\rho(u)!}.$$

Proof *See [24] and [64].*

3.2 Numerical Solution

Based upon the approach for realizing the Taylor expansion with respect to the exact solution of DAE system (2.1), this section describes how to derive Taylor series with respect to its numerical solution by ROW-type method (2.3). An algebraic theory based on rooted trees is developed that combines and extends strategies introduced by Roche [64] regarding ROW methods with exact Jacobian for DAEs as well as Steihaug and Wolfbrandt [71] regarding W methods with non-exact Jacobian for ODEs. In this context, the special case given by $B_z = (g_z)_0$ is considered below. Resulting derivatives of differential and algebraic parts are considered first. Afterwards, new structures of rooted trees are introduced in order to express their elementary differentials and coefficients graphically. Finally, these elements are used to formulate the Taylor expansion of the numerical solution explicitly.

3.2.1 Derivatives

Derivatives of numerical solutions y_1 and z_1 are generally determined by including the information of rooted trees belonging to the set DAT. This approach is based on strategies introduced by Roche [64] and similar to steps considered by Hairer and Wanner [24] for finding derivatives for the numerical solution by ROW methods with exact Jacobian for DAEs and W methods with non-exact Jacobian for ODEs. In fact, the following derivation adapts the descriptions given in [24] in order to apply them to the generalized ROW-type formulation (2.3) assuming $B_z = (g_z)_0$.

First of all, to determine differentiations of numerical solutions according to (2.3a), unknowns y_1 and z_1 are considered to be functions of the step-size h. Regarding Taylor expansions at $h = 0$, their derivatives read [24]:

$$y_1^{(q)}(0) = \sum_{i=1}^{s} b_i (k_i)^{(q)}(0) \qquad \text{and} \qquad z_1^{(q)}(0) = \sum_{i=1}^{s} b_i (k_i^{alg})^{(q)}(0). \tag{3.3}$$

To express these terms explicitly derivatives of given increments k_i and k_i^{alg} must be determined. For k_i derivatives result from Leibniz' rule applied to the first equation of (2.3b) while assuming $h = 0$. Leibniz' rule is generally defined by [23]:

$$(h\varphi(h))^{(q)}\Big|_{h=0} = q \cdot (\varphi(h))^{(q-1)}\Big|_{h=0}. \tag{3.4}$$

Differentiating increments k_i thus reads

$$\begin{aligned}(k_i)^{(q)}\Big|_{h=0} &= \left(h \left[f(v_i, w_i) + A_y \sum_{j=1}^{i} \gamma_{ij} k_j + A_z \sum_{j=1}^{i} \gamma_{ij} k_j^{alg} \right] \right)^{(q)} \Bigg|_{h=0} \\ &= q \left(f(v_i, w_i) + A_y \sum_{j=1}^{i} \gamma_{ij} k_j + A_z \sum_{j=1}^{i} \gamma_{ij} k_j^{alg} \right)^{(q-1)} \Bigg|_{h=0}\end{aligned}$$

or, equivalently,

$$(k_i)^{(q)} = q f(v_i, w_i)^{(q-1)} + q A_y \sum_{j=1}^{i} \gamma_{ij} (k_j)^{(q-1)} + q A_z \sum_{j=1}^{i} \gamma_{ij} (k_j^{alg})^{(q-1)}. \tag{3.5}$$

For k_i^{alg} derivatives are determined by dividing the second equation of (2.3b) by h first. Assuming computations at $h = 0$ and exact Jacobian entries $B_z = (g_z)_0$, differentiation leads to

$$0 = g(v_i, w_i)^{(q)} + B_y \sum_{j=1}^{i} \gamma_{ij} (k_j)^{(q)} + (g_z)_0 \sum_{j=1}^{i} \gamma_{ij} (k_j^{alg})^{(q)}. \tag{3.6}$$

Derivatives of f in (3.5) and g in (3.6) can be determined using Faà di Bruno's formula [23, 24]. Purpose of this formula is to express these derivatives as a sum that considers the information of rooted trees belonging to the set $LDAT$. In fact, only trees of a the special subset $SLDAT \subset LDAT$ need to be taken into account. In this context, $SLDAT = SLDAT_y \cup SLDAT_z$ denotes the set of special monotonically labeled trees. It is reduced to trees of the set $LDAT$ that have no ramifications except for the root. Also, fat vertices of these trees are allowed to occur as the root or in a direct connection to the root exclusively.

With respect to derivatives of f given in (3.5) Faà di Bruno's formula reads [24]:

$$(f(v_i, w_i))^{(q-1)} = \sum_{\substack{t \in SLDAT_y \\ \rho(t)=q}} \frac{\partial^{m+n} f(v_i, w_i)}{\partial y^m \partial z^n} \left(v_i^{(\mu_1)}, ..., v_i^{(\mu_m)}, w_i^{(\nu_1)}, ..., w_i^{(\nu_n)} \right). \tag{3.7}$$

With respect to derivatives of g given in (3.6) Faà di Bruno's formula reads [24]:

$$\begin{aligned}(g(v_i, w_i))^{(q)} = &\sum_{\substack{u \in SLDAT_z \\ \rho(u)=q}} \frac{\partial^{m+n} g(v_i, w_i)}{\partial y^m \partial z^n} \left(v_i^{(\mu_1)}, ..., v_i^{(\mu_m)}, w_i^{(\nu_1)}, ..., w_i^{(\nu_n)} \right) \\ &+ g_z(v_i, w_i) w_i^{(q)}. \end{aligned} \tag{3.8}$$

In both these formulas m corresponds to the number of sub-trees $t_1, ..., t_m \in DAT_y$ and n corresponds to the number of sub-trees $u_1, ..., u_n \in DAT_z$ that are used to construct elements $t = [t_1, ..., t_m, u_1, ..., u_n]_y \in SLDAT_y$ or $u = [t_1, ..., t_m, u_1, ..., u_n]_z \in SLDAT_z$. Based on Definition 3.2, given integer values $\mu_1, ..., \mu_m$ and $\nu_1, ..., \nu_n$ are the orders of sub-trees $t_1, ..., t_m$ characterized by a meager root and $u_1, ..., u_n$ characterized by a fat root, respectively. As a consequence, it holds $\mu_1 + ... + \mu_m + \nu_1 + ... + \nu_n = q - 1$ with respect to (3.7) and $\mu_1 + ... + \mu_m + \nu_1 + ... + \nu_n = q$ with respect to (3.8) [24]. Besides, a significant difference between formulations (3.7) and (3.8) is the term with g_z additionally included in (3.8). It is considered as trees of the form $u = [u_1]_z$ are not defined for set DAT according to Definition 3.1. Hence, this expression required is not covered by the given sum over all elements of $SLDAT_z$. It thus must be considered separately [24].

For both expressions of Faà di Bruno's formula occurring derivatives of v_i and w_i which are defined by (2.3c) correspond to [24]:

$$v_i^{(\mu_\xi)} = \sum_{\kappa_\xi=1}^{i-1} \alpha_{i\kappa_\xi} (k_{\kappa_\xi})^{(\mu_\xi)} \qquad \text{and} \qquad w_i^{(\nu_\xi)} = \sum_{\kappa_{m+\xi}=1}^{i-1} \alpha_{i\kappa_{m+\xi}} (k_{\kappa_{m+\xi}}^{alg})^{(\nu_\xi)}. \tag{3.9}$$

Here, parameter ξ regards values $1, ..., m$ with respect to the derivatives $v_i^{(\mu_\xi)}$ and values $1, ..., n$ with respect to the derivatives $w_i^{(\nu_\xi)}$.

Derivatives of values v_i and w_i can now be inserted into (3.7) that defines differentiations of function f. Afterwards, substituting resulting expressions of (3.7) into (3.5) yields derivatives of k_i. As all expressions have to be computed at $h = 0$ (cf. [24]) the resulting formulation finally reads

$$\begin{aligned}(k_i)^{(q)} =& q \cdot \sum_{\substack{t \in SLDAT_y \\ \rho(t)=q}} \frac{\partial^{m+n} f(y_0, z_0)}{\partial y^m \partial z^n} \left(v_i^{(\mu_1)}, ..., v_i^{(\mu_m)}, w_i^{(\nu_1)}, ..., w_i^{(\nu_n)} \right) \\ &+ q \cdot A_y \sum_{j=1}^{i} \gamma_{ij} (k_j)^{(q-1)} + q \cdot A_z \sum_{j=1}^{i} \gamma_{ij} (k_j^{alg})^{(q-1)}. \end{aligned} \tag{3.10}$$

with $v_i^{(\mu_\xi)}$ for $\xi = 1, ..., m$ and $w_i^{(\nu_\xi)}$ for $\xi = 1, ..., n$ according to (3.9).

In order to determine a corresponding formulation with respect to k_i^{alg}, derivatives of values v_i and w_i must be inserted into (3.8) that defines differentiations of function g. Afterwards, substituting resulting expressions of (3.8) into (3.6) and assuming all expressions to be evaluated at $h = 0$ (cf. [24]) yields

$$\begin{aligned} 0 =& \sum_{\substack{u \in SLDAT_z \\ \rho(u)=q}} \frac{\partial^{m+n} g(y_0, z_0)}{\partial y^m \partial z^n} \left(v_i^{(\mu_1)}, ..., v_i^{(\mu_m)}, w_i^{(\nu_1)}, ..., w_i^{(\nu_n)} \right) \\ &+ (g_z)_0 \sum_{j=1}^{i-1} \alpha_{ij} (k_j^{alg})^{(q)} + (g_z)_0 \sum_{j=1}^{i} \gamma_{ij} (k_j^{alg})^{(q)} + B_y \sum_{j=1}^{i} \gamma_{ij} (k_j)^{(q)} \end{aligned} \tag{3.11}$$

when considering $g_z(v_i, w_i) = (g_z)_0$ as well as $v_i^{(\mu_\xi)}$ for $\xi = 1, ..., m$ and $w_i^{(\nu_\xi)}$ for $\xi = 1, ..., n$ according to (3.9).

By introducing coefficients $\beta_{ij} = \alpha_{ij} + \gamma_{ij}$, expressions with g_z given in the last line of (3.11) can be summed up. The resulting term includes all expressions of $(k_i^{alg})^{(q)}$. Writing this term to the left thus yields

$$\begin{aligned} -(g_z)_0 \sum_{j=1}^{i} \beta_{ij} (k_j^{alg})^{(q)} =& \sum_{\substack{u \in SLDAT_z \\ \rho(u)=q}} \frac{\partial^{m+n} g(y_0, z_0)}{\partial y^m \partial z^n} \left(v_i^{(\mu_1)}, ..., v_i^{(\mu_m)}, w_i^{(\nu_1)}, ..., w_i^{(\nu_n)} \right) \\ &+ B_y \sum_{j=1}^{i} \gamma_{ij} (k_j)^{(q)}. \end{aligned} \tag{3.12}$$

By introducing $(\omega_{ij})_{i,j=1}^{s} = \mathcal{B}^{-1}$ with $\mathcal{B} = (\beta_{ij})_{i,j=1}^{s}$ this formulation can be resolved for given derivatives of k_i^{alg} explicitly. For the convenience of subsequent definitions it is advantageous to regard the given sum over all elements of $SLDAT_z$ restricted to trees satisfying $m+n \geq 2$. Therefore, the case $m=1$, $n=0$ is additionally excluded from the sum over all elements of $SLDAT_z$. This finally leads to:

$$\begin{aligned}
(k_i^{alg})^{(q)} =&(-g_z^{-1})_0 \sum_{j=1}^{i} \omega_{ij} \cdot \sum_{\substack{u \in SLDAT_z \\ \rho(u)=q \\ m+n \geq 2}} \frac{\partial^{m+n} g(y_0, z_0)}{\partial y^m \partial z^n} \left(v_j^{(\mu_1)}, ..., v_j^{(\mu_m)}, w_j^{(\nu_1)}, ..., w_j^{(\nu_n)}\right) \\
&+ (-g_z^{-1})_0 \sum_{j=1}^{i} \omega_{ij} \cdot g_y \sum_{k=1}^{j-1} \alpha_{jk} (k_k)^{(q)} \\
&+ (-g_z^{-1})_0 \sum_{j=1}^{i} \omega_{ij} \cdot B_y \sum_{k=1}^{j} \gamma_{jk} (k_k)^{(q)}. \qquad (3.13)
\end{aligned}$$

By inserting the formulations (3.10) and (3.13) into expressions of (3.3) differentiations of numerical solutions y_1 and z_1 can be stated explicitly for lower orders. For instance, assuming the realization of a second order scheme derivatives with respect to differential parts read

$$y_1' = \sum b_i \cdot f \qquad (3.14a)$$

$$\begin{aligned}
y_1'' = 2 \cdot &\sum b_i \alpha_{ij} \cdot f_y f \\
+ 2 \cdot &\sum b_i \gamma_{ij} \cdot A_y f \\
+ 2 \cdot &\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} \cdot f_z (-g_z^{-1}) g_y f \\
+ 2 \cdot &\sum b_i \alpha_{ij} \omega_{jk} \gamma_{kl} \cdot f_z (-g_z^{-1}) B_y f \\
+ 2 \cdot &\sum b_i \gamma_{ij} \omega_{jk} \alpha_{kl} \cdot A_z (-g_z^{-1}) g_y f \\
+ 2 \cdot &\sum b_i \gamma_{ij} \omega_{jk} \gamma_{kl} \cdot A_z (-g_z^{-1}) B_y f \qquad (3.14b)
\end{aligned}$$

while derivatives with respect to algebraic parts are

$$\begin{aligned}
z_1' = &\sum b_i \omega_{ij} \alpha_{jk} \cdot (-g_z^{-1}) g_y f \\
+ &\sum b_i \omega_{ij} \gamma_{jk} \cdot (-g_z^{-1}) B_y f \qquad (3.14c)
\end{aligned}$$

3.2.2 Trees and Differentials

As for the analytical solution, derivatives of higher order become increasingly complicated. For this reason, the theory of rooted trees previously considered for describing exact solutions should also be applied to express components of the numerical solution. However, the theory of rooted trees given in Section 3.1.2 requires to describe exact elementary differentials that occur for derivatives of the analytical solution only. In contrast, applications regarding the numerical solution have to consider additional non-exact elementary differentials plus given coefficients. For this purpose, the number of vertices and the strategies for constructing trees as considered by Roche [64] or Hairer and Wanner [24] are not sufficient. Hence, the concept of rooted trees must be extended.

Besides exact elementary differentials already known for the analytical solution, additional non-exact differentials resulting for derivatives of the numerical solution are a consequence of arbitrary Jacobian entries given within ROW-type method (2.3). In this context, exact differentials are characterized by expressions that consider exact derivatives f_y, f_z, g_y and g_z exclusively. By restricting to the case $B_z = (g_z)_0$, non-exact differentials are characterized by including at least one non-exact component A_y, A_z or B_y.

The theory of rooted trees considered by Roche [64] covers rooted trees that are able to describe exact differentials only. These trees are characterized by two types of vertices: Meager vertices (•) that describe exact differential components f and fat vertices (○) that describe exact differential components $(-g_z^{-1})g$. However, when having to take into account additional non-exact differentials, these vertices are not sufficient. This is because non-exact components A_y, A_z and B_y cannot be expressed. In order to completely describe all the elementary differentials occurring within the numerical solution, it therefore is necessary to introduce further types of vertices.

A first extension regarding the set of vertices for constructing corresponding rooted trees was already published as part of this thesis in [30]. However, the approach presented therein was restricted to arbitrary approximated Jacobian entries with respect to given differential parts. This means, only possible non-exact entries A_y and A_z were taken into account while assuming exact Jacobian entries with respect to algebraic parts given, i.e. considering $B_y = (g_y)_0$ and $B_z = (g_z)_0$. For this purpose, a meager vertex framed by a square (⊡) was introduced to express the non-exact differential components A.

Below, the approach considered in [30] will be further extended such that arbitrary Jacobian entries B_y are included as well. Besides meager vertices with and without frame and fat vertices this requires to introduce a fourth vertex type. For this purpose, a fat vertex framed by a square (⧇) is considered to describe resulting differential components of the form $(-g_z^{-1})B$. Together with meager and fat vertices introduced by Roche [64] to describe exact differential components f and $(-g_z^{-1})g$, respectively, and meager vertices framed by a square introduced in [30] to describe non-exact differential components A, this enables to express all elementary differentials that occur for the numerical solution by ROW-type method (2.3) when assuming $B_z = (g_z)_0$. Furthermore, strategies given in [64] and [30] remain preserved.

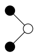
As for the theory considered in [30], derivatives of f and g are still characterized by a branch leaving a meager or fat vertex, respectively, while derivatives of A are still characterized by a branch leaving a meager vertex framed by a square. Analogously, occurring derivatives of non-exact differential components B are now characterized by a branch leaving the introduced fat vertex framed by a square. In this context, any branch that is followed by a meager vertex with or without a square frame represents derivatives with respect to y while a branch that is followed by a fat vertex with or without a square frame represents derivatives with respect to z.

Example 3.5

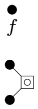
Differentials for derivatives y_1' and y_1'' given in (3.14a) and (3.14b) are represented by

$f_y f$

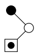
$A_y f$



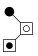
$f_z(-g_z^{-1})g_y f$



$f_z(-g_z^{-1})B_y f$



$A_z(-g_z^{-1})g_y f$



$A_z(-g_z^{-1})B_y f$

Differentials given for derivatives of z_1' given in (3.14c) are represented by

$(-g_z^{-1})g_y f$

$(-g_z^{-1})B_y f$

By using four different types of vertices, rooted trees that describe exact differentials (i.e. trees consisting of meager and fat vertices without frame exclusively) are still realized by the rules presented in Definition 3.1 and Procedure 3.1. However, regarding tree structures for non-exact differential components (i.e. trees including at least one meager or fat vertex framed by a square) new definitions and procedures must be formulated.

In [30] the set $ADAT^D$ was defined, i.e. the set of approximated differential-algebraic rooted trees with respect to given differential parts. Elements of this set are rooted trees which express all the non-exact differentials that occur for derivatives of the numerical solution when assuming $B_y = (g_y)_0$ and $B_z = (g_z)_0$. They only cover trees having meager vertices framed by a square. By considering a ROW-type scheme that includes additional arbitrary Jacobian entries B_y, this set must be extended without violating properties known for $ADAT^D$. This new set will be denoted by $ADAT^{DA}$.

Definition 3.5 *Let $ADAT^{DA} = ADAT_y^{DA} \cup ADAT_z^{DA}$ ($ADAT_y^{DA} \cap ADAT_z^{DA} = \varnothing$) denote the set of approximated differential-algebraic rooted trees with respect to given differential and algebraic parts. Elements of $ADAT^{DA}$ include at least one meager or fat vertex framed by a square that is followed by a single branch. Corresponding structures represent an arbitrary approximation to Jacobian expressions f_y, f_z and g_y denoted by A_y, A_z and B_y, respectively. Elements of $ADAT_y^{DA}$ are characterized by a meager root with or without square frame. Elements of $ADAT_z^{DA}$ are characterized by a fat root with or without square frame.*

Remark 3.6

- *$ADAT^{DA}$ corresponds to a superset. Let the set $ADAT^D$ cover all rooted trees that exclusively include just meager vertices with a square frame besides standard meager and fat vertices as defined in [30]. Analogously, let the set $ADAT^A$ cover all rooted trees that exclusively include just fat vertices with a square frame besides standard meager and fat vertices. Finally, let the set $ADAT^C$ cover all rooted trees that exclusively include a combined number of meager and fat vertices framed by a square besides standard meager and fat vertices. In this* *context, elements of $ADAT^D$ and $ADAT^A$ must include at least one framed vertex while elements of $ADAT^C$ must include at least two framed vertices. Then $ADAT^{DA} = ADAT^D \cup ADAT^A \cup ADAT^C$.*

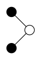
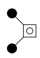
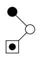
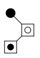
- *The name $ADAT^{DA}$ is used instead of $ADAT$ as it is not the most superior superset possible. This is because it is restricted to the case $B_z = (g_z)_0$. So, it excludes the possibility of regarding non-exact Jacobian entries B_z. The term $ADAT$ should be reserved for the set that also includes tree elements for the case $B_z \neq (g_z)_0$.*

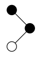
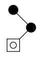
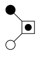
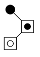
Analogously to elements of the set $ADAT^D$ introduced in [30], constructing rooted trees of the set $ADAT^{DA}$ follows the rules known for realizing W methods applied to ODEs as considered in [24]. This means that vertices with square frame can occur within singly branched trees only. They will never be given in the center of a ramification and they will never be given at the end of a branch. This is a consequence of (3.10) regarding meager vertices with square frame and (3.13) regarding fat vertices with square frame. Also, as known for ROW methods solving DAEs considered in [64], fat vertices without square frame will never be given at the end of a single branch, too. In addition, fat vertices with or without square frame can never be followed by another fat vertex with or without square frame in a single branch.

Remark 3.7 *With respect to W methods applied to ODEs Hairer and Wanner [24] consider similar properties by defining the set TW. TW is the subset of P-trees with singly-branched fat vertices and meager end-vertices. In this context, fat vertices considered by Hairer and Wanner correspond to meager framed vertices in this thesis.*

In fact, due to the properties described, rooted trees belonging to the set $ADAT^{DA}$ can be determined analogously to the way introduced in [30] regarding elements of $ADAT^D$. This means, all the elements of $ADAT^{DA}$ can be found by permuting square frames over all singly branched vertices that are given for trees of the set DAT which describe the exact solution. This permutation must include given roots as long as these are singly branched, but it excludes the end-vertices of any branch. However, contrary to the strategy introduced in [30] this procedure is not restricted to meager vertices when determining elements of the set $ADAT^{DA}$. As elements of $ADAT^{DA}$ also consider approximations to original Jacobian entries g_y, the process of permuting a square frame must be applied to singly branched fat vertices also. In this context, this strategy distinctly determines all elements of the subset $ADAT_y^{DA}$ based on trees of the subset DAT_y and all elements of the subset $ADAT_z^{DA}$ based on trees of the subset DAT_z.

Example 3.6

 $\in DAT_y$ *yields trees*  ,  ,  $\in ADAT_y^{DA}$

 $\in DAT_z$ *yields trees*  ,  ,  $\in ADAT_z^{DA}$

As any element of the set DAT yields a number of elements of the set $ADAT^{DA}$ by permutation, special subsets of trees denoted by $\Phi(t)$ and $\Phi(u)$ can be defined in analogy to [30]. These can be used to describe specific properties of the given ROW-type scheme.

Definition 3.6 *Let $\Phi(t) = t, t(1), t(2), ...$ and $\Phi(u) = u, u(1), u(2), ...$ denote compilations of rooted trees that are characterized by following properties:*

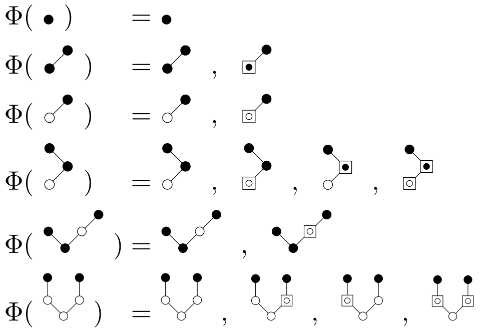
a) $\Phi(t)$ includes $t \in DAT_y$ and all elements $t(1), t(2), ... \in ADAT_y^{DA}$ that result from permuting a square frame over all meager and fat vertices of t, excluding those vertices that consist of ramifications or correspond to the end of a branch.

b) $\Phi(u)$ includes $u \in DAT_z$ and all elements $u(1), u(2), ... \in ADAT_z^{DA}$ that result from permuting a square frame over all meager and fat vertices of u, excluding those vertices that consist of ramifications or correspond to the end of a branch.

Remark 3.8

- *Regarding trees of the sets $ADAT_y^{DA}$ and $ADAT_z^{DA}$ that result from applying the permutation strategies mentioned, denotation $t(1), t(2), ...$ and $u(1), u(2), ...$ is used instead of $t_1, t_2, ...$ and $u_1, u_2, ...$, respectively. This way, conflicts with previous formulations are avoided where $t_1, t_2, ...$ and $u_1, u_2, ...$ denote given subtrees.*
- *An important property of compilations according to Definition 3.6 is that all elements of a compilation $\Phi(t)$ or $\Phi(u)$ will be equal to the tree $t \in DAT_y$ or $u \in DAT_z$, respectively, when using the special approximations $A_y = (f_y)_0$, $A_z = (f_z)_0$ and $B_y = (g_y)_0$. So, these approximations correspond to replacing meager and fat vertices with square frame by meager and fat vertices without square frame.*

Example 3.7



The total of elements given in the sets DAT and $ADAT^{DA}$ graphically represents all components that result for derivatives of numerical solutions by ROW-type scheme (2.3) when assuming $B_z = (g_z)_0$. So, all occurring exact and non-exact differentials as well as their coefficients are distinctly expressed. For convenience of subsequent definitions that explain how to assign each of these trees to corresponding differentials and coefficients in detail, their general construction is recursively defined by the unifying set $CDAT^{DA}$. By including all elements of DAT and $ADAT^{DA}$, the construction rules defined for the set $CDAT^{DA}$ enable to find all trees required to describe derivatives of the numerical solution directly. Hence, there is no need to determine elements of the set DAT based on Definition 3.1 in a first step before knowing corresponding elements of $ADAT^{DA}$ by permutation.

Definition 3.7 *Let $CDAT^{DA} = CDAT_y^{DA} \cup CDAT_z^{DA}$ ($CDAT_y^{DA} \cap CDAT_z^{DA} = \varnothing$) denote the set of combined differential algebraic rooted trees with respect to differential and algebraic parts characterized by $CDAT^{DA} = DAT \cup ADAT^{DA}$ ($DAT \cap ADAT^{DA} = \varnothing$). The set is recursively defined by:*

(a) $\tau_y = \bullet \in CDAT_y^{DA}$, $\tau_z =$ $\in CDAT_z^{DA}$

(b) $[t_1, ..., t_m, u_1, ..., u_n]_y \in CDAT_y^{DA}$,
if $t_1, ..., t_m \in CDAT_y^{DA}$, $u_1, ..., u_n \in CDAT_z^{DA}$

(c) $[t_1]_{\tilde{y}} \in CDAT_y^{DA}$,
if $t_1 \in CDAT_y^{DA}$

(d) $[u_1]_{\tilde{y}} \in CDAT_y^{DA}$,
if $u_1 \in CDAT_z^{DA}$

(e) $[t_1, ..., t_m, u_1, ..., u_n]_z \in CDAT_z^{DA}$,
if $t_1, ..., t_m \in CDAT_y^{DA}$, $u_1, ..., u_n \in CDAT_z^{DA}$ *and* $m + n \geq 2$

(f) $[t_1]_z \in CDAT_z^{DA}$,
if $t_1 \in CDAT_y^{DA}$

(g) $[t_1]_{\tilde{z}} \in CDAT_z^{DA}$,
if $t_1 \in CDAT_y^{DA}$

$[t_1, ..., t_m, u_1, ..., u_n]_{y,\tilde{y},z,\tilde{z}}$ *represent unordered (m+n)-tuples.*

Remark 3.9

- *Items a) to f) include the construction of the set $CDAT^D$ described by Definition 7 in [30]. So, the set $CDAT^{DA}$ is a direct extension of the set $CDAT^D$ by including elements of the subsets $ADAT^A$ and $ADAT^C$ according to the first item of Remark 3.6 via item g). Hence, it follows $CDAT^{DA} = CDAT^D \cup ADAT^A \cup ADAT^C$.*

- *Items a) and b) as well as items e) and f) given in Definition 3.7 include items a) to d) of Definition 3.1. So, these items are responsible for realizing trees that represent the exact solution when $t_1, ..., t_m \in DAT_y$ and $u_1, ..., u_n \in DAT_z$.*

- *The name $CDAT^{DA}$ is used instead of $CDAT$ as it is not the most superior superset possible. This is because it is restricted to the case $B_z = (g_z)_0$. So, it excludes the possibility of regarding non-exact Jacobian entries B_z. The term $CDAT$ should be reserved for the set that also includes tree elements for the case $B_z \neq (g_z)_0$.*

- *By Definition 3.1 dealing with set DAT and Definition 3.7 dealing with set $CDAT^{DA}$ it holds $ADAT^{DA} = CDAT^{DA} \backslash DAT$.*

- *$CDAT_y^{DA} = DAT_y \cup ADAT_y^{DA}$ and $CDAT_z^{DA} = DAT_z \cup ADAT_z^{DA}$.*

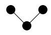
- *Compilations $\Phi(t)$ and $\Phi(u)$ according to Definition 3.6 are subsets of $CDAT_y^{DA}$ and $CDAT_z^{DA}$, respectively.*

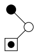
- *In item a) of Definition 3.7 the tree τ_z is redundant. It is theoretically covered by $[t_1]_z$ in item f) when using $t_1 = \tau_y$ according to item a). However, denoting τ_z explicitly allows to keep consistency with descriptions given in literature such as Hairer and Wanner [24] or Roche [64].*

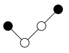
Regarding graphical representation, $[.]_y$ denotes attaching a meager root and $[.]_z$ denotes attaching a fat root to all sub-trees given within brackets by a corresponding number of branches. This corresponds to the approach known for constructing elements of the set DAT according to [24, 64]. Besides, $[.]_{\tilde{y}}$ denotes attaching a meager root framed by a square as known for constructing elements of the set $ADAT^D$ according to [30]. Analogously, $[.]_{\tilde{z}}$ denotes attaching a fat root framed by a square, corresponding to the new type of vertices introduced.

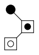
Example 3.8

Let $t_1 =$ • , $t_2 =$, $u_1 =$ be given sub-trees.

$[t_1, t_1]_y$ = 

$[u_1]_{\tilde{y}}$ = 

$[t_1, u_1]_z$ = 

$[t_2]_{\tilde{z}}$ = 

To be consistent with theories and procedures defined in literature, working with tree structures of the set $CDAT^{DA}$ that include new vertices requires to redefine the order term according to Definition 3.2.

Definition 3.8 *The order of a tree $t \in CDAT_y^{DA}$ or $u \in CDAT_z^{DA}$ is denoted by $\rho(t)$ or $\rho(u)$, respectively, and corresponds to the number of all its meager vertices that are given with or without square frame.*

Each tree of a specific order q that belongs to the set $CDAT^{DA}$ distinctly represents an elementary differential that results for derivatives of the numerical solution regarding the same order. In this context, any tree without framed vertices corresponds to trees of the set DAT according to Definition 3.1. Therefore, these trees describe exact differentials as given in Definition 3.3. However, any tree including at least one framed vertex corresponds to a non-exact differential. As the definition of trees belonging to the unifying set $CDAT^{DA}$ is given by recursive construction according to Definition 3.7 all the corresponding exact and non-exact elementary differentials that result for derivatives of the numerical solution can be defined by recursive construction as well.

Definition 3.9 *Rooted trees of the set $CDAT^{DA}$ are recursively mapped to elementary differentials $F(t)$ and $F(u)$ as follows:*

(a) $F(\tau_y) = f,\ F(\tau_z) = (-g_z^{-1})g_y f$

(b) $F(t) = \frac{\partial^{m+n} f}{\partial y^m \partial z^n}\left(F(t_1), ..., F(t_m), F(u_1), ..., F(u_n)\right),$
if $t = [t_1, ..., t_m, u_1, ..., u_n]_y \in CDAT_y^{DA}$

(c) $F(t) = A_y F(t_1),$
if $t = [t_1]_{\tilde{y}} \in CDAT_y^{DA}$

(d) $F(t) = A_z F(u_1),$
if $t = [u_1]_{\tilde{y}} \in CDAT_y^{DA}$

(e) $F(u) = (-g_z^{-1})\frac{\partial^{m+n}}{\partial y^m \partial z^n}\left(F(t_1), ..., F(t_m), F(u_1), ..., F(u_n)\right),$
if $u = [t_1, ..., t_m, u_1, ..., u_n]_z \in CDAT_z^{DA}$ *and* $m + n \geq 2$

(f) $F(u) = (-g_z^{-1})g_y F(t_1),$
if $u = [t_1]_z \in CDAT_z^{DA}$

(g) $F(u) = (-g_z^{-1})B_y F(t_1),$
if $u = [t_1]_{\tilde{z}} \in CDAT_z^{DA}$

Remark 3.10

- *Items a) to g) within Definition 3.9 match to items a) to g) within Definition 3.7, i.e., a rooted tree given according to an item in Definition 3.7 yields the elementary differential according to the same item in Definition 3.9.*
- *Items a) and b) as well as items e) and f) of Definition 3.9 include items a) and b) as well as c) and d) of Definition 3.3.*
- *Items a) to f) of Definition 3.9 include items a) to f) of Definition 9 in [30].*

As for Definition 3.3, all resulting elementary differentials $F(t)$ and $F(u)$ are well defined as Definition 3.9 is unaffected by permutations of $t_1, ..., t_m, u_1, ..., u_n$. This is due to the symmetry of partial derivatives. However, for the same reason some differentials will occur multiple times when deriving differentiations to formulate Taylor expansions of the numerical solution. In order to determine all occurring exact and non-exact elementary differentials required for defining Taylor expansions completely, the steps defined by Hairer and Wanner [24] according to Procedure 3.1 can be supplemented. In this context, the property that all elements of the set $CDAT^{DA} = DAT \cup ADAT^{DA}$ are generally based on elements of the set DAT is used. First, the approach for finding elements of the set DAT is applied to describe exact differentials. Afterwards, elements to describe non-exact differentials result from applying a suitable permutation along singly branched vertices.

Procedure 3.2 *Considering all the rooted trees of an order q that belong to subset DAT_y or DAT_z and thus represent the elementary differentials to express derivatives of the same order with respect to the analytical solution, trees of the subset $CDAT_y^{DA}$ or $CDAT_z^{DA}$ to describe all the exact and non-exact differentials for derivatives of the numerical solution with respect to the order $q+1$ completely result from*

i) attaching a new branch with τ_y to each given vertex,

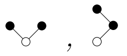
ii) attaching a new branch with τ_z to each given vertex,

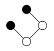
iii) splitting each fat vertex into two new fat vertices directly connected by a single branch and attaching another branch with τ_y to the lower of these fat vertices,

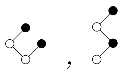
iv) splitting each fat vertex into two new fat vertices directly connected by a single branch and attaching another branch with τ_z to the lower of these fat vertices,

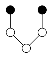
v) permuting a square frame over all inner vertices of given single branches for the trees constructed by steps i) - iv), including their singly branched roots.

Example 3.9
The only tree of DAT_z that regards order $q = 1$ is $z' =$ *. Based on this element, all trees of the set $CDAT_z^{DA}$ required for constructing z'' (i.e. regarding order $q = 2$) of the numerical solution are determined by using the steps i) - v) of Procedure 3.2. Resulting trees per step read:*

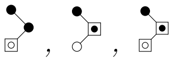
i) 

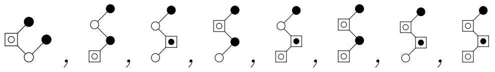
iii) 

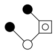
ii) 

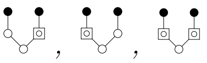
iv) 

v)


(*permuted trees of step i)*)


(*permuted trees of step ii)*)


(*permuted trees of step iii)*)

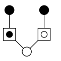

(*permuted trees of step iv)*)

Obviously, steps $i)$ - $iv)$ are the same as in Procedure 3.1 regarding trees for exact differentials represented by elements of the sets DAT_y and DAT_z. Just permutations with respect to singly branched inner vertices and roots must be considered in order to determine all the missing elements for describing corresponding non-exact differentials which are represented by elements of the sets $ADAT_y^{DA}$ and $ADAT_z^{DA}$. Hence, supplementing Procedure 3.1 by step $v)$ is sufficient to define an approach that finds all rooted trees of the unifying set $CDAT^{DA}$ in order to describe components which result for derivatives of the numerical solution by generalized ROW-type scheme (2.3) completely, at least as long as $B_z = (g_z)_0$ is satisfied.

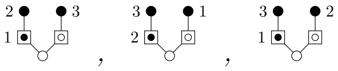
Analogously to definitions regarding the derivation of exact solutions, some of the rooted trees must occur multiple times due to the symmetry of partial derivatives. In order to differentiate these trees distinctly, the concept of labeled trees considered in Definition 3.4 with respect to elements of the set DAT is also applied to elements of the set $CDAT^{DA}$.

Definition 3.10 *$LCDAT^{DA}$ denotes the set of monotonically labeled rooted trees of $CDAT^{DA}$ with $LCDAT^{DA} = LCDAT_y^{DA} \cup LCDAT_z^{DA}$. Elements $t \in LCDAT_y^{DA}$ and $u \in LCDAT_z^{DA}$ are characterized by integer labels i with $1 \le i \le \rho(t)$ and $1 \le i \le \rho(u)$. Labels are given for each meager vertex with and without frame and increase monotonically for every branch starting from the root.*

Example 3.10



can be monotonically labeled by

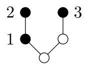


Remark 3.11

- *It holds $LCDAT^{DA} = LDAT \cup LADAT^{DA}$, with $LDAT$ covering monotonically labeled trees of the set DAT according to Definition 3.4 and $LADAT^{DA}$ covering monotonically labeled trees of the set $ADAT^{DA}$.*
- *Definition 3.10 regarding elements of $CDAT^{DA}$ is the same as for Definition 10 in [30] regarding elements of $CDAT^{D}$. However, new trees are included that are characterized by framed fat vertices, i.e. trees belonging to the subset resulting for $ADAT^{DA} \backslash ADAT^{D}$ with $ADAT^{DA}$ according to Definition 3.5 and $ADAT^{D}$ according to Definition 5 in [30].*

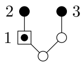
Note that labeling of trees $t \in LDAT_y$ and $u \in LDAT_z$ according to Definition 3.4 (originally defined in [64]) is adopted by every element of corresponding subsets $\Phi(t)$ and $\Phi(u)$. Considering $B_z = (g_z)_0$, this ensures that for special approximations $A_y = (f_y)_0$, $A_z = (f_z)_0$ and $B_y = (g_y)_0$ all elements of $\Phi(t)$ and $\Phi(u)$ will become equal to elements $t \in LDAT_y$ and $u \in LDAT_z$, respectively, and therefore represent the same exact differential.

Example 3.11

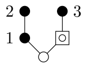


$u \in LDAT_z$

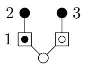
yields



$u_1 \in LDAT_z^{DA}$,



$u_2 \in LDAT_z^{DA}$,



$u_3 \in LDAT_z^{DA}$

$$\Phi(u) = \{u, u_1, u_2, u_3\}$$

3.2.3 Coefficients

Trees of the set $CDAT^{DA}$ do not just identify occurring elementary differentials with respect to derivatives of the numerical solution by generalized ROW-type scheme (2.3) when assuming $B_z = (g_z)_0$. They are also used to identify coefficients. In this context, coefficients consisting of values α_{ij}, γ_{ij} and ω_{ij} are considered separately from integer coefficients. They are formulated similar to the coefficients described by Hairer and Wanner [24]. As for elementary differentials, each tree then distinctly represents a special combination of coefficients. Their construction is given by recursive definitions as well.

Definition 3.11 *Let $\phi_i(t)$ and $\phi_i(u)$ be the coefficients occurring in front of elementary differentials for increment derivatives of generalized ROW-type method (2.3) assuming $B_z = (g_z)_0$. Based on trees of the set $CDAT^{DA}$, constructing $\phi_i(t)$ and $\phi_i(u)$ is recursively defined by:*

(a) $\phi_i(\tau_y) = 1,\ \phi_i(\tau_z) = \sum \omega_{ij}\alpha_{jk}$

(b) $\phi_i(t) = \sum \alpha_{i\kappa_1} \cdot ... \cdot \alpha_{i\kappa_m} \cdot \alpha_{i\kappa_{m+1}} \cdot ... \cdot \alpha_{i\kappa_{m+n}} \cdot$
$\phi_{\kappa_1}(t_1) \cdot ... \cdot \phi_{\kappa_m}(t_m) \cdot \phi_{\kappa_{m+1}}(u_1) \cdot ... \cdot \phi_{\kappa_{m+n}}(u_n)$
if $t = [t_1, ..., t_m, u_1, ..., u_n]_y \in CDAT_y^{DA}$

(c) $\phi_i(t) = \sum \gamma_{i\kappa} \cdot \phi_\kappa(t_1)$
if $t = [t_1]_{\tilde{y}} \in CDAT_y^{DA}$

(d) $\phi_i(t) = \sum \gamma_{i\kappa} \cdot \phi_\kappa(u_1)$
if $t = [u_1]_{\tilde{y}} \in CDAT_y^{DA}$

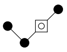
(e) $\phi_i(u) = \sum \omega_{ij} \cdot \alpha_{j\kappa_1} \cdot ... \cdot \alpha_{j\kappa_m} \cdot \alpha_{j\kappa_{m+1}} \cdot ... \cdot \alpha_{j\kappa_{m+n}} \cdot$
$\phi_{\kappa_1}(t_1) \cdot ... \cdot \phi_{\kappa_m}(t_m) \cdot \phi_{\kappa_{m+1}}(u_1) \cdot ... \cdot \phi_{\kappa_{m+n}}(u_n)$
if $u = [t_1, ..., t_m, u_1, ..., u_n]_z \in CDAT_z^{DA},\ m + n \geq 2$

(f) $\phi_i(u) = \sum \omega_{ij} \cdot \alpha_{j\kappa} \cdot \phi_\kappa(t_1)$
if $u = [t_1]_z \in CDAT_z^{DA}$

(g) $\phi_i(u) = \sum \omega_{ij} \cdot \gamma_{j\kappa} \cdot \phi_\kappa(t_1)$
if $u = [t_1]_{\tilde{z}} \in CDAT_z^{DA}$

Given sums run over all $j, \kappa, \kappa_1, ..., \kappa_m, \kappa_{m+1}, ..., \kappa_{m+n}$.

Example 3.12

 *yields* $\phi_i(t) = \sum \alpha_{ij}\alpha_{ik}\omega_{kl}\gamma_{lm}$

Remark 3.12

- *Items a) to g) within Definition 3.11 match to items a) to g) within Definition 3.7, i.e., a rooted tree given according to an item in Definition 3.7 yields the coefficients according to the same item in Definition 3.11.*
- *Items a) to g) of Definition 3.11 include items a) to f) of Definition 11 in [30]. In this context, $\phi_i(\tau_y)$ within item a) and items b) to e) of Definition 3.11 cover the same items of Definition 11 in [30] directly. However, the element $\phi_i(\tau_z)$ as well as item f) of Definition 11 in [30] require to assume $B_y = (g_y)_0$ and to take into account $\beta_{ij} = \alpha_{ij} + \gamma_{ij}$ as well as $(\omega_{ij})^s_{i,j=1} = \mathcal{B}^{-1}$ with $\mathcal{B} = (\beta_{ij})^s_{i,j=1}$. Under these conditions Definition 3.11 yields $\phi_i(\tau_z)$ and item f) of Definition 11 in [30] when summing up $\phi_i(\tau_z)$ with item g) (regarding $\phi_k(t_1) = \phi_k(\tau_y)$) and when summing up item f) with item g), respectively.*

Definition 3.12 *Let $\gamma(t)$ and $\gamma(u)$ be the integer coefficients occurring in front of elementary differentials for increment derivatives of generalized ROW-type method (2.3) assuming $B_z = (g_z)_0$. Based on trees of the set $CDAT^{DA}$, constructing $\gamma(t)$ and $\gamma(u)$ is recursively defined by:*

(a) $\gamma(\tau_y) = 1$, $\gamma(\tau_z) = 1$

(b) $\gamma(t) = \rho(t) \cdot \gamma(t_1) \cdot ... \cdot \gamma(t_m) \cdot \gamma(u_1) \cdot ... \cdot \gamma(u_n)$
if $t = [t_1, ..., t_m, u_1, ..., u_n]_y \in CDAT_y^{DA}$

(c) $\gamma(t) = \rho(t) \cdot \gamma(t_1)$
if $t = [t_1]_{\tilde{y}} \in CDAT_y^{DA}$

(d) $\gamma(t) = \rho(t) \cdot \gamma(u_1)$
if $t = [u_1]_{\tilde{y}} \in CDAT_y^{DA}$

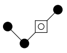
(e) $\gamma(u) = \gamma(t_1) \cdot ... \cdot \gamma(t_m) \cdot \gamma(u_1) \cdot ... \cdot \gamma(u_n)$
if $u = [t_1, ..., t_m, u_1, ..., u_n]_z \in CDAT_z^{DA}$, $m + n \geq 2$

(f) $\gamma(u) = \gamma(t_1)$
if $u = [t_1]_z \in CDAT_z^{DA}$

(g) $\gamma(u) = \gamma(t_1)$
if $u = [t_1]_{\tilde{z}} \in CDAT_z^{DA}$

$\rho(t)$ and $\rho(u)$ are the orders of trees $t \in CDAT_y^{DA}$ and $u \in CDAT_z^{DA}$.

Example 3.13

 *yields* $\gamma(t) = 3$

Remark 3.13

- *Items a) to g) within Definition 3.12 match to items a) to g) within Definition 3.7, i.e., a rooted tree given according to an item in Definition 3.7 yields the integer coefficients according to the same item in Definition 3.12.*
- *Items a) to f) of Definition 3.12 include items a) to f) of Definition 12 in [30]. This is because the integer coefficients are not affected when assuming $B_y \neq (g_y)_0$. Just item g) must be additionally taken into account.*

3.2.4 Taylor Expansions

Based on Definition 3.9 for constructing elementary differentials, Definition 3.11 for determining coefficients and Definition 3.12 for finding integer coefficients of trees resulting for the combining set $CDAT^{DA}$ according to Definition 3.7, derivatives of increments k_i and k_i^{alg} can be formulated explicitly regarding monotonically labeled elements of the set $LCDAT^{DA}$ defined in Definition 3.10. In this context, the graphical representation of characteristics via rooted trees as well as resulting definitions significantly simplify the formulation of Taylor expansions with respect to numerical solutions by generalized ROW-type method (2.3) when assuming $B_z = (g_z)_0$.

Theorem 3.2 *Derivatives of increments k_i and k_i^{alg} given with respect to generalized ROW-type method (2.3) when assuming $B_z = (g_z)_0$ read*

$$(k_i)^{(q)} = \sum_{\substack{t \in LCDAT_y^{DA} \\ \rho(t)=q}} \gamma(t)\phi_i(t)F(t)(y_0, z_0)$$

$$(k_i^{alg})^{(q)} = \sum_{\substack{u \in LCDAT_z^{DA} \\ \rho(u)=q}} \gamma(u)\phi_i(u)F(u)(y_0, z_0)$$

with elementary differentials and coefficients according to Definitions 3.9, 3.11 and 3.12.

Proof *Analogously to ROW methods for DAEs by induction on q for (3.10) and (3.13) and regarding resulting summations afterwards as described in [24].*

Remark 3.14 *Theorem 3.2 is given analogously to Theorem 4.6 in [24] regarding classical ROW methods for DAEs with exact Jacobian as well as Theorem 2 in [30] regarding generalized ROW-type methods for DAEs with arbitrary approximations to Jacobian entries restricted to its differential components, i.e. assuming arbitrary A_y and A_z but $B_y = (g_y)_0$ and $B_z = (g_z)_0$. However, it refers to a superior set of rooted trees.*

Based on Theorem 3.2 derivatives of the numerical solution as well as Taylor expansions given with respect to the generalized ROW-type method (2.3) when assuming $B_z = (g_z)_0$ can finally be expressed by following Theorem 3.3.

Theorem 3.3 *Derivatives for the numerical solution of an index-one semi-explicit DAE system given by (2.1) using generalized ROW-type method (2.3) with* $B_z = (g_z)_0$ *correspond to*

$$y_1^{(q)}\Big|_{h=0} = \sum_{\substack{t \in LCDAT_y^{DA} \\ \rho(t)=q}} \gamma(t) \cdot \sum_{j=1}^{s} b_i \phi_i(t) F(t)(y_0, z_0)$$

$$z_1^{(q)}\Big|_{h=0} = \sum_{\substack{u \in LCDAT_z^{DA} \\ \rho(u)=q}} \gamma(u) \cdot \sum_{j=1}^{s} b_i \phi_i(u) F(u)(y_0, z_0)$$

with elementary differentials and coefficients according to Definition 3.9, 3.11 and 3.12. Hence, Taylor expansions of the numerical solution with $B_z = (g_z)_0$ *read*

$$y_1(x_0 + h) = \sum_{\substack{t \in LCDAT_y^{DA} \\ \rho(t)=q}} \gamma(t) \cdot \sum_{j=1}^{s} b_i \phi_i(t) F(t)(y_0, z_0) \cdot \frac{h^{\rho(t)}}{\rho(t)!}$$

$$z_1(x_0 + h) = \sum_{\substack{u \in LCDAT_z^{DA} \\ \rho(u)=q}} \gamma(u) \cdot \sum_{j=1}^{s} b_i \phi_i(u) F(u)(y_0, z_0) \cdot \frac{h^{\rho(u)}}{\rho(u)!}.$$

Proof *In analogy to the proof of Theorem 3.1.*

4 Consistency, Convergence and Stability

Reliable schemes for time integration require to be consistent, convergent and stable. In this context, there are relevant general results with respect to one-step methods for solving semi-explicit DAEs of index one present in literature. Corresponding findings can also be applied to ROW-type method (2.3). Hence, the given section recalls some general aspects regarding the expected orders of consistency and convergence first. Afterwards, order conditions required to achieve a specific order of consistency by ROW-type method (2.3) when assuming $B_z = (g_z)_0$ are detailed. For that purpose, the theory of rooted trees previously established is utilized. Finally, some general remarks with respect to stability properties are considered.

4.1 General Aspects

ROW-type method (2.3) corresponds to the general class of one-step schemes for DAEs defined and analyzed by Deuflhard, Hairer and Zugck [10]:

$$y_{n+1} = y_n + h \cdot \mathcal{F}(y_n, z_n, h) \tag{4.1a}$$

$$z_{n+1} = \mathcal{G}(y_n, z_n, h). \tag{4.1b}$$

Hence, certain properties of method (2.3) can be determined by considering results presented in [10], even though it is characterized by non-exact Jacobian approximations. This particularly means that the definition with respect to the order of consistency given by Roche in [64] can be applied.

Definition 4.1 *(Roche 1988, [64])*
A method of the form (4.1) is consistent of order p if

$$y(x_0 + h) - y_1 = \mathcal{O}(h^{p+1}) \qquad \textit{and} \qquad z(x_0 + h) - z_1 = \mathcal{O}(h^p)$$

with y_1, z_1 being the numerical solutions and y, z being the exact solutions at $x_0 + h$.

The order of convergence with respect to the limit case $h \to 0$ is fixed due to the Global Convergence Theorem I introduced by Deuflhard et al. [10, 62].

Theorem 4.1 *(Deuflhard et al. 1987, [10])*
For solving semi-explicit DAE systems (2.1) with consistent initial values, let the generalized one-step method (4.1) be applied. If the method is characterized by

a) consistency order p, and

b) $\left\| \frac{\partial \mathcal{G}(y,z,h=0)}{\partial z} \right\| \leq \alpha < 1$ in a neighborhood of the solution,

then, the one-step method yields order of convergence p. This means, for fixed $x = x_0 + n \cdot h$ it follows

$$y_n - y(x) = \mathcal{O}(h^p) \qquad \textit{and} \qquad z_n - z(x) = \mathcal{O}(h^p)$$

with y_n and z_n being the numerical solutions of given differential and algebraic parts after having applied (4.1) n times.

Proof *See [10].*

Remark 4.1

- *A necessary condition for applying Theorem 4.1 is that $|R(\infty)| < 1$ is satisfied, with $R(\tilde{z})$ being the stability function of the one-step method used. This follows from aspect b) in Theorem 4.1 which is called contractivity condition with contractivity number α [10].*
- *According to Roche [64] it holds $\alpha = R(\infty)$ directly for ROW methods that solve DAEs by means of exact Jacobian entries. In this context, $R(\tilde{z})$ is the stability function of the ROW method applied to the Dahlquist test equation $y' = \lambda y$ with $y(0) = 1$, $\lambda \in \mathbb{C}$ and $\tilde{z} = h\lambda$.*

4.2 Consistency

Order conditions that are required to achieve a specific order of consistency can be derived by using the theory of rooted trees. In this context, conditions for realizing schemes up to order three are shown in detail below. Also, propositions to reduce the total number of order conditions as well as further relevant properties are presented within subsequent subsections.

4.2.1 Definition of Order Conditions

Conditions to achieve a specific order of consistency are determined by comparing the Taylor series of the exact solution (see Theorem 3.1) to the Taylor series of the numerical solution (see Theorem 3.3). The comparison yields the following results:

Theorem 4.2 *A generalized ROW-type method of the form (2.3) with $B_z = (g_z)_0$ that is consistent of order p satisfies*

$$y(x_0+h) - y_1 = \mathcal{O}(h^{p+1}) \quad \textit{iff}$$

$$\sum_{i=1}^{s} b_i \phi_i(t) = 1/\gamma(t) \qquad \textit{for} \quad t \in DAT_y,\ \rho(t) \le p$$

$$\sum_{i=1}^{s} b_i \phi_i(t) = 0 \qquad \textit{for} \quad t \in CDAT_y^{DA} \backslash DAT_y,\ \rho(t) \le p$$

$$z(x_0+h) - z_1 = \mathcal{O}(h^{p}) \quad \textit{iff}$$

$$\sum_{i=1}^{s} b_i \phi_i(u) = 1/\gamma(u) \qquad \textit{for} \quad u \in DAT_z,\ \rho(u) \le p$$

$$\sum_{i=1}^{s} b_i \phi_i(u) = 0 \qquad \textit{for} \quad u \in CDAT_z^{DA} \backslash DAT_z,\ \rho(u) \le p$$

with coefficients $\phi_i(t)$, $\phi_i(u)$ and $\gamma(t)$, $\gamma(u)$ according to Definition 3.11 and Definition 3.12 considering rooted trees of the set DAT and $CDAT^{DA}$ according to Definition 3.1 and Definition 3.7.

Proof *In analogy to [24].*

Theorem 4.2 states that only components belonging to exact elementary differentials are allowed to contribute to the solution. Hence, coefficient terms given in front of each exact differential (represented by a tree of the set DAT) must be equal to one while coefficient terms given in front of each non-exact differential (represented by a tree of the set $CDAT^{DA}\backslash DAT$) must be equal to zero. This way, the Taylor series of the numerical solution finally equals the Taylor expansion of the exact solution.

Remark 4.2

- *Theorem 4.2 is given in direct analogy to Theorem 4 presented in [30]. However, it considers extended sets of rooted trees that enable to take into account non-exact Jacobian entries B_y besides non-exact Jacobian entries A_y and A_z.*

- *In principle, Theorem 4.2 corresponds to an extended combination of Theorem 7.7 given in chapter IV.7 of [24] that regards ROW methods with non-exact Jacobian applied to ODEs and Theorem 4.8 given in chapter VI.4 of [24] regarding ROW methods with exact Jacobian applied to DAEs.*

4.2.2 Construction of Order Conditions

Both the left-hand side and the right-hand side of each order condition that results according to Theorem 4.2 can be distinctly identified by using rooted trees of the set $CDAT^{DA}$. The general approach is based upon Definition 3.11 and Definition 3.12. In his context, it corresponds to an altered version of strategies presented by Hairer and Wanner in [24].

In order to realize the procedure, trees that result according to Definition 3.7 must be labeled using summation indices (i, j, k, ...) with respect to their vertices first. In this context, it is necessary to attach one index to each meager vertex given with or without square frame and two indices to each fat vertex given with or without frame. Afterwards, the left-hand side of the order conditions can be identified by a sum over all the summation indices given. This sum is applied to a product whose factors are coefficients distinctly assigned to vertices of the tree element considered. The order of these factors represents the structure defined in Definition 3.11. It is determined by following rules. There is

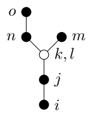
b_i	with i being the index of a meager root with or without frame or the first index of a fat root with or without frame,
α_{ij}	with i being the index of a meager vertex without frame or the second index of a fat vertex without frame and j being the index of an attached meager vertex with or without frame or the first index of an attached fat vertex with or without frame,

γ_{ij} with i being the index of a meager vertex with frame or the second index of a fat vertex with frame and j being the index of an attached meager vertex with or without frame or the first index of an attached fat vertex with or without frame,

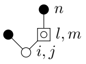
ω_{ij} with i being the first index and j being the second index of a given fat vertex with or without frame.

The right-hand side of the order conditions can be identified by given tree structures as well. It equals zero with respect to rooted trees of the set $CDAT^{DA} \backslash DAT$, i.e., trees that include at least one framed vertex. It equals one divided by a specific value with respect to rooted trees of the set DAT, i.e., trees that include no framed vertex. The specific value corresponds to the integer coefficients according to Definition 3.12. It can be determined by taking into account the orders of tree components included that are characterized by the number of their meager vertices. The process is as follows: Starting from the order of a tree given to represent an order condition, the specific value results after successively multiplying the orders of subtrees that result when eliminating the present root. However, this approach only applies to sub-trees with meager root (i.e. elements belonging to subset DAT_y). The actual step is skipped for sub-trees with fat root (i.e. elements belonging to subset DAT_z). This is because integer coefficients are not affected when attaching a fat vertex.

Example 4.1



$$\sum_{i,j,k,l,m,n,o} b_i \alpha_{ij} \alpha_{jk} \omega_{kl} \alpha_{lm} \alpha_{ln} \alpha_{no} = \frac{1}{5 \cdot 4 \cdot 2 \cdot 1} = \frac{1}{40}$$



$$\sum_{i,j,k,l,m,n} b_i \omega_{ij} \alpha_{jk} \alpha_{jl} \omega_{lm} \gamma_{mn} = 0$$

Remark 4.3 *The original procedure for determining the order condition that belongs to a given tree was introduced by Hairer and Wanner in chapter VI.4 of [24]. It considered the derivation with respect to ROW methods with exact Jacobian applied to DAEs. The procedure presented thus corresponds to a direct extension by supplementing combinations of coefficients which result for using non-exact Jacobian entries.*

4.2.3 Resulting Conditions and Properties

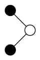
Applying the approach introduced to all the trees of the set $CDAT^{DA}$ that result by the rules of Definition 3.7 finally yields the order conditions for the ROW-type method

(2.3) when assuming $B_z = (g_z)_0$. There are 9 conditions for realizing the method up to consistency order two, while there are 85 conditions for realizing the method up to consistency order three. Table 4.1 summarizes the conditions that result for order two. A detailed list of the conditions that result for order three is shown by Table A.1 given in Appendix A.

Note that rooted trees within given tables are numbered using a specific notation. For example, within Table 4.1 the first entries given for conditions of the differential parts and algebraic parts are called (1D1.0) and (1A1.0). This notation was introduced in [30]. The first number specifies the order of the given tree (i.e. the order of consistency the given condition belongs to). The subsequent letter identifies elements of the set $CDAT_y^{DA}$ (i.e. conditions of the differential parts) by D and elements of the set $CDAT_z^{DA}$ (i.e. conditions of the algebraic parts) by A. The next number counts the subsets $\Phi(t)$ and $\Phi(u)$ (see Definition 3.6). Hence, trees that have equal numbers in this position of the notation belong to a common subset. Finally, the last number is used to distinguish trees within these subsets. Also, it shows if corresponding trees represent exact or non-exact elementary differentials: The last number of the notation is always equal to zero when regarding trees of the set DAT and unequal to zero when regarding trees of the set $ADAT^{DA}$.

Within Table 4.1 and Table A.1 all trees that belong to a common subset $\Phi(t)$ or $\Phi(u)$ are confined by horizontal lines. Due to the definition of vertices, elements of the set $ADAT^{DA}$ in between these lines will turn into the element of the set DAT automatically when applying exact Jacobian entries (i.e. choosing $A_y = (f_y)_0$, $A_z = (f_z)_0$ and $B_y = (g_y)_0$ within the given ROW-type method (2.3) while assuming $B_z = (g_z)_0$). As they represent the same elementary differential then, their order conditions sum up and result in a condition that was originally introduced in [64].

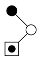
Example 4.2 *Regarding rooted tree (2D2.0) given in Table 4.1*



$f_z(-g_z^{-1})g_y f$

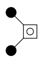
$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}=1/2$

that results for constructing elements of consistency order $p = 2$, remaining trees for describing the corresponding subset $\Phi(t)$ are given by the elements (2D2.1), (2D2.2) and (2D2.3):



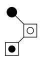
$A_z(-g_z^{-1})g_y f$

$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}=0$,



$f_z(-g_z^{-1})B_y f$

$\sum b_i\alpha_{ij}\omega_{jk}\gamma_{kl}=0$,



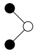
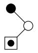
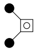
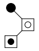
$A_z(-g_z^{-1})B_y f$

$\sum b_i\gamma_{ij}\omega_{jk}\gamma_{kl}=0$

Now, assuming $A_y = (f_y)_0$, $A_z = (f_z)_0$ and $B_y = (g_y)_0$ all these trees and their elementary differentials become equal to element (2D2.0). Hence, the order conditions given for all four trees of the subset $\Phi(t)$ sum up. By considering $\beta_{ij} = \alpha_{ij} + \gamma_{ij}$ the resulting element reads:

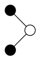
Table 4.1: Conditions for Consistency Order p=2.

Differential Part

$\rho(t)$	Req.	No.	DAT_y	$ADAT_y^{DA}$	Differential	Condition	Comment			
1	X	1D1.0			f	$\sum b_i = 1$	RK	W	Re/Ro/St	Ja/St
2	X	2D1.0			$f_y f$	$\sum b_i \alpha_{ij} = 1/2$	RK	W	Re/Ro/St	Ja/St
	X	2D1.1			$A_y f$	$\sum b_i \gamma_{ij} = 0$		W		Ja/St
	X	2D2.0			$f_z(-g_z^{-1})g_y f$	$\sum b_i \alpha_{ij}\omega_{jk}\alpha_{kl} = 1/2$				
		2D2.1			$A_z(-g_z^{-1})g_y f$	$\sum b_i \gamma_{ij}\omega_{jk}\alpha_{kl} = 0$				
		2D2.2			$f_z(-g_z^{-1})B_y f$	$\sum b_i \alpha_{ij}\omega_{jk}\gamma_{kl} = 0$				
		2D2.3			$A_z(-g_z^{-1})B_y f$	$\sum b_i \gamma_{ij}\omega_{jk}\gamma_{kl} = 0$				

Algebraic Part

$\rho(t)$	Req.	No.	DAT_z	$ADAT_z^{DA}$	Differential	Condition	Comment
1	X	1A1.0			$(-g_z^{-1})g_y f$	$\sum b_i \omega_{ij}\alpha_{jk} = 1$	
		1A1.1			$(-g_z^{-1})B_y f$	$\sum b_i \omega_{ij}\gamma_{jk} = 0$	



$f_z(-g_z^{-1})g_y f$

$\sum b_i\beta_{ij}\omega_{jk}\beta_{kl}=1/2$

Because $(\omega_{ij})_{i,j=1}^{s}=\mathcal{B}^{-1}$ *with* $\mathcal{B}=(\beta_{ij})_{i,j=1}^{s}$ *the condition reduces to:*

$f_z(-g_z^{-1})g_y f$

$\sum b_i\beta_{ij}=1/2$

This is a combination of (2D1.0) and (2D1.1) within Table 4.1 when using $A_y=(f_y)_0$*. Resulting tree, elementary differential and order condition thus equal to condition (4.11b) that was defined in [64] to realize ROW methods with exact Jacobian for DAEs.*

Remark 4.4 *Due to the properties of elements that belong to a common subset* $\Phi(t)$ *or* $\Phi(u)$*, given order conditions implicitly cover the conditions defined by Roche in [64]. Conditions of [64] explicitly covered are the element* τ_y *and trees that consist just of ramifications, i.e. trees without branches where single vertices are directly connected to each other. This is because such tree structures cannot include non-exact Jacobian entries by definition. As detailed by Definition 3.7, framed vertices that describe non-exact components are not allowed to occur in the center of a ramification or at the end of a given branch.*

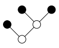
Example 4.3 *Conditions resulting for rooted trees such as*

$\sum b_i=1$,

$\sum b_i\alpha_{ij}\alpha_{ik}=\frac{1}{3}$,

$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{jl}=1$

or



$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{jl}\omega_{lm}\alpha_{mn}\alpha_{mo}=1$

directly correspond to the order conditions introduced by Roche in [64].

Besides the order conditions of Roche [64] that are implicitly included, conditions of standard explicit one-step Runge-Kutta methods for ODEs [23] (RK), of W methods for ODEs introduced by Steihaug and Wolfbrandt [71] (W), of the ROW-type method for DAEs (Type II) presented by Rentrop, Roche and Steinebach [62] (Re/Ro/St) as well as the ROW-type method for DAEs considered in [30] (Ja/St) are explicitly covered. For that reason, they are additionally commented within given tables. In this context, the scheme by Rentrop et al. [62] corresponds to ROW-type method (2.3) when using $A_y=0$, $A_z=0$ together with $B_y=(g_y)_0$ and $B_z=(g_z)_0$. So, it is a scheme that applies explicit integration to given differential parts exclusively by assuming them to be non-stiff. The scheme published in [30] corresponds to ROW-type method (2.3) when using arbitrary Jacobian entries A_y and A_z together with $B_y=(g_y)_0$ and $B_z=(g_z)_0$.

Remark 4.5

- *Given order conditions are a direct extension of the conditions presented in [30]. This is because Jacobian entries of the scheme considered in [30] are nearly the same as for the method (2.3). Differences are given by including non-exact Jacobian entries B_y within (2.3) while assuming exact entries B_z.*

- *Correspondingly resulting additional conditions are expressed by trees that include singly branched fat vertices. Within literature, such tree structures are generally not considered. This is because because singly branched fat vertices followed by a meager vertex can be neglected when applying exact Jacobian entries B_y (see Example 4.2, see Proposition 4.6 in [64] and Proposition 1 in [30]). As a consequence, these structures supplement conditions in [30] where fat vertices occur in the center of a ramification at most.*

Due to the general characteristics of given tree structures covered, order conditions of familiar ROW-type schemes known from literature are easily identified. For this purpose, subsequent propositions can be considered that are inspired by Roche's Proposition 4.10 presented in [64].

Proposition 4.1 *Regarding the ROW-type method (2.3) applied to ODEs, order conditions represented by rooted trees $t \in DAT_y \subset CDAT_y^{DA}$ are identical to those of classical explicit one-step Runge-Kutta methods for ODEs (see Chapter II.2 in [23]). Corresponding trees are characterized by having only meager vertices without frame.*

Proof *Assume Jacobian entries A_y to be equal to zero with respect to Definitions 3.7 - 3.12. These definitions are then reduced to item a) (regarding element τ_y) and b) regarding trees $t \in DAT_y$ that are characterized by meager vertices without frame. The proof is completed by comparing resulting components for Theorem 4.2 with Theorem 2.11 presented in Chapter II.2 of [23].*

Proposition 4.2 *Regarding the ROW-type method (2.3) applied to ODEs, order conditions represented by rooted trees $t \in DAT_y \cup ADAT_y^D \subset CDAT_y^{DA}$ are identical to those of classical W methods for ODEs introduced in [71]. Corresponding trees are characterized by having only meager vertices with and without frame.*

Proof *Assume Jacobian entries A_y to be arbitrary approximations with respect to Definitions 3.7 - 3.12. These definitions are then reduced to items a) (regarding element τ_y), b) and c) regarding trees $t \in DAT_y \cup ADAT_y^D$ that are characterized by meager vertices with and without frame. The proof is completed by comparing resulting components for Theorem 4.2 with Theorem 7.7 presented in Chapter IV.7 of [24].*

Proposition 4.3 *Regarding the ROW-type method (2.3) applied to DAEs, order conditions represented by rooted trees $t \in DAT_y \subset CDAT_y^{DA}$ and $u \in DAT_z \subset CDAT_z^{DA}$ are identical to those of the ROW-type method (Type II) introduced in [62]. Corresponding trees are characterized by having only meager and fat vertices without frame and, in addition, include no singly branched fat vertices.*

Proof *Assume Jacobian entries A_y, A_z to be equal to zero and B_y, B_z to be exact approximations with respect to Definitions 3.7 - 3.12. These definitions are then reduced to items a), b) and e) regarding trees $t \in DAT_y$ and $u \in DAT_z$ that are characterized by meager and fat vertices without frame and, in addition, include no singly branched fat vertices. The proof is completed by comparing resulting components for Theorem 4.2 with elements given in Table 3.2 presented in [62].*

Proposition 4.4 *Regarding the ROW-type method (2.3) applied to DAEs, order conditions represented by rooted trees $t \in DAT_y \cup ADAT_y^D \subset CDAT_y^{DA}$ and $u \in DAT_z \cup ADAT_z^D \subset CDAT_z^{DA}$ are identical to those of the generalized ROW-type method introduced in [30]. Corresponding trees are characterized by having only meager vertices with and without frame plus fat vertices without frame that occur in the center of a ramification at most.*

Proof *Assume Jacobian entries A_y, A_z to be arbitrary approximations and B_y, B_z to be exact approximations with respect to Definitions 3.7 - 3.12. These definitions are then reduced to items a), b), c), d), e) and f) regarding trees $t \in DAT_y \cup ADAT_y^D$ and $u \in DAT_z \cup ADAT_z^D$ that are characterized by meager vertices with and without frame plus fat vertices without frame. The proof is completed by comparing resulting components for Theorem 4.2 with Theorem 4 presented in [30].*

Remark 4.6 *Besides the covered order conditions of different ROW-type schemes discussed, the given theory implicitly includes the order conditions of every method mentioned within Table 2.1 and Table 2.2. This means, when choosing the Jacobian entries as stated within these tables, the presented approach automatically yields the order conditions introduced within the corresponding references.*

4.2.4 Redundant Conditions

Although they describe properties of ROW-type method (2.3) completely when assuming $B_z = (g_z)_0$, not all of the order conditions that result for tree structures derived must be considered explicitly for its realization. In fact, many of the order conditions turn out to be redundant. Hence, there is a difference regarding conditions that are needed for describing the given characteristics completely and conditions that are needed for implementing the given scheme appropriately. In order to identify redundant conditions, propositions similar to those presented in [30] or [64] can be defined that reduce the number of required conditions significantly.

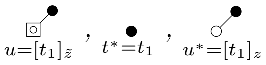
Proposition 4.5 *Order conditions of rooted trees characterized by $u = [t_1]_{\tilde{z}}$ with $t_1 \in CDAT_y^{DA}$ are automatically satisfied by a combination of order conditions that result for the trees $t^* = t_1$ and $u^* = [t_1]_z$. It holds:*

$$\gamma(u) \sum b_i \phi_i(u) = \gamma(t^*) \sum b_i \phi_i(t^*) - \gamma(u^*) \sum b_i \phi_i(u^*).$$

Proof *By Theorem 4.2 together with items a) and g) (elements of order one) as well as items b), f) and g) (elements of order greater one) of Definition 3.11 and Definition 3.12 taking into account $(\omega_{ij})_{i,j=1}^s = B^{-1}$ with $B = (\beta_{ij})_{i,j=1}^s$ and $\beta_{ij} = \alpha_{ij} + \gamma_{ij}$.*

Example 4.4 *Consider the simple tree:*

which yields



With respect to resulting order conditions it follows:

$$\sum b_i\omega_{ij}\gamma_{jk} = 0 \qquad \hat{=}$$

$$\sum b_i\omega_{ij}(\beta_{jk}-\alpha_{jk}) = 0$$

$$\underbrace{\sum b_i\omega_{ij}\beta_{jk}}_{\substack{\sum b_i=1 \\ \hat{=}}} - \underbrace{\sum b_i\omega_{ij}\alpha_{jk}}_{\substack{\sum b_i\omega_{ij}\alpha_{jk}=1 \\ \hat{=}}} = 0$$

Remark 4.7

- *As a consequence of Proposition 4.5, all the conditions resulting for rooted trees that include at least one framed fat vertex can be completely neglected when realizing ROW-type method (2.3) with $B_z = (g_z)_0$ up to a specific order.*
- *Note that for $B_z \neq (g_z)_0$ related order conditions expressed by tree structures that include fat framed vertices become relevant. Hence, having introduced the type of nodes characterized by framed fat vertices is justified.*

Proposition 4.6 *Order conditions of rooted trees characterized by $t = [u]_{\tilde y}$ where $u = [t_1, ..., t_m, u_1, ..., u_n]_z$ with $t_1, ..., t_m \in CDAT_y^{DA}$ and $u_1, ..., u_n \in CDAT_z^{DA}$ are automatically satisfied by a combination of order conditions that result for the trees $t^* = [t_1, ..., t_m, u_1, ..., u_n]_y$ and $t^{**} = [u]_y$. It holds:*

$$\gamma(t)\sum b_i\phi_i(t) = \gamma(t^*)\sum b_i\phi_i(t^*) - \gamma(t^{**})\sum b_i\phi_i(t^{**}).$$

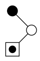
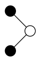
Proof *By Theorem 4.2 together with items b), d) and e) of Definition 3.11 and Definition 3.12 taking into account $B = (\beta_{ij})_{i,j=1}^s$ with $\beta_{ij} = \alpha_{ij} + \gamma_{ij}$ and $(\omega_{ij})_{i,j=1}^s = B^{-1}$.*

Remark 4.8 *Contrary to Proposition 4.5 given Proposition 4.6 cannot be applied to elements of order one. Relevant tree structures require at least order two. Hence, in contrast to the proof of Proposition 4.5 it is not necessary to differ between elements of order one and elements of order greater one within the proof of Proposition 4.6.*

Example 4.5 *Consider the simple tree:*

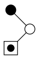
● t_1 *which yields* $u=[t_1]_z$

and therefore

 $t=[u]_{\tilde{y}}$, $t^*=[t_1]_y$,  $t^{**}=[u]_y$

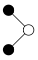
With respect to resulting order conditions it follows:

$$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl} = 0 \qquad \hat{=}$$



$$\sum b_i(\beta_{ij}-\alpha_{ij})\omega_{jk}\alpha_{kl} = 0$$

$$\underbrace{\sum b_i\beta_{ij}\omega_{jk}\alpha_{kl}}_{\substack{\sum b_i\alpha_{ij}=1/2 \\ \hat{=}}} - \underbrace{\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}}_{\substack{\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}=1/2 \\ \hat{=}}} = 0$$



Remark 4.9

- *As a consequence of Proposition 4.6, all the conditions resulting for rooted trees that include at least one framed meager vertex directly followed by a fat vertex regarding a single branch can be completely neglected when realizing ROW-type method (2.3) with $B_z = (g_z)_0$ up to a specific order.*

- *Note that for $B_z \neq (g_z)_0$ related order conditions expressed by tree structures that include meager framed vertices directly followed by a fat vertex become relevant.*

- *Framed meager vertices directly followed by framed fat vertices would lead to analogous rules for identifying redundant conditions. However, this case is not considered within Proposition 4.6 explicitly as trees including framed fat vertices were excluded by Proposition 4.5 already.*

Remark 4.10 *With respect to propositions that reduce the total number of order conditions for ROW-type methods presented in [30] or [64] the following additional statements can be made:*

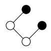
- *Proposition 4.6 corresponds to Proposition 2 presented in [30]. This is because they refer to tree structures and combinations of coefficients that generally occur within Taylor series of ROW-type method (2.3) when assuming arbitrary Jacobian entries A_y and A_z together with exact Jacobian entries $B_y = (g_y)_0$ and $B_z = (g_z)_0$ (the case of Jacobian approximations originally considered in [30]).*

- *A significant difference of Proposition 4.6 compared to Proposition 2 given in [30] is that it is not restricted to sub-trees $u = [t_1, ..., t_m, u_1, ..., u_n]_z$ with $m + n \geq 2$. This means, Proposition 4.6 must also be applied to rooted trees that include* *singly branched sub-trees $u = [t_1]_z$. Corresponding cases were not considered by Proposition 2 defined in [30] explicitly as such tree structures are excluded by Proposition 1 given in [30].*

- *Proposition 1 given in [30] (also Proposition 4.6 given in [64]) is not applicable regarding generalized ROW-type method (2.3) assuming arbitrary Jacobian entries A_y, A_z and B_y together with exact Jacobian entries $B_z = (g_z)_0$. It is only valid when assuming $B_y = (g_y)_0$ in addition.*

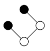
Remark 4.11 *It is well-known that for all classical one-step schemes derived via rooted trees elements that consider the same combination of sub-trees attached to a common root in varying order are equivalent. This is a consequence of the symmetry of partial derivatives (Schwarz's Theorem). Hence, corresponding order conditions are redundant [24, 81]. Arising of such elements cannot be neglected by Procedure 3.2. For that reason, this property is mentioned here although it is considered to be obvious.*

Example 4.6 *Order conditions for the following tree structures are equal:*



$\sum b_i \omega_{ij} \alpha_{jk} \omega_{kl} \alpha_{lm} \alpha_{jn}$

and



$\sum b_i \omega_{ij} \alpha_{jk} \alpha_{jl} \omega_{lm} \alpha_{mn}$

By using Proposition 4.5, Proposition 4.6 and Remark 4.11 the total amount of order conditions that must be considered for realizing ROW-type method (2.3) assuming $B_z = (g_z)_0$ can be significantly reduced. In fact, the number of relevant conditions reduces from 85 to 26 when realizing a scheme of order three while the number of relevant conditions reduces from 9 to 5 when realizing a scheme of order two. For that reason, the conditions that are necessarily required for implementation are additionally marked within tables that include redundant tree structures. In this context, redundant conditions are considered within Table 4.1 and Table A.1 for the sake of completeness and for demonstrating given properties. Note that the 26 conditions which result for a scheme of order three after applying the rules that reduce the number of conditions are summarized within Table 4.2 below.

4.2.5 Further Properties

Besides the characteristics previously mentioned some additional observations can be made with respect to the resulting order conditions:

1. Given propositions to reduce the number of conditions only affect tree structures that occur by including approximations of Jacobian entries B_y. Tree structures that describe order conditions of methods already defined in literature remain completely unaffected and, thus, are directly covered.

2. When satisfying the order conditions necessarily required, Taylor expansions of the numerical solution by ROW-type method (2.3) are not affected in any way by the choice of Jacobian entries B_y. This is because corresponding non-exact elementary differentials cancel automatically due to Proposition 4.5.

Table 4.2: Conditions Required for Order 3 (1 of 3).

Differential Part

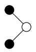
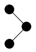
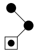
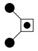
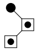
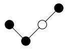
$\rho(t)$	No.	DAT_y	$ADAT_y^{DA}$	Differential	Condition	Comment			
1	1D1.0			f	$\sum b_i = 1$	RK	W	Re/Ro/St	Ja/St
2	2D1.0			$f_y f$	$\sum b_i \alpha_{ij} = 1/2$	RK	W	Re/Ro/St	Ja/St
	2D1.1			$A_y f$	$\sum b_i \gamma_{ij} = 0$		W		Ja/St
	2D2.0			$f_z(-g_z^{-1}) g_y f$	$\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} = 1/2$				
3	3D1.0			$f_{yy}(f, f)$	$\sum b_i \alpha_{ij} \alpha_{ik} = 1/3$	RK	W	Re/Ro/St	Ja/St
	3D2.0			$f_y f_y f$	$\sum b_i \alpha_{ij} \alpha_{jk} = 1/6$	RK	W	Re/Ro/St	Ja/St
	3D2.1			$A_y f_y f$	$\sum b_i \gamma_{ij} \alpha_{jk} = 0$		W		Ja/St
	3D2.2			$f_y A_y f$	$\sum b_i \alpha_{ij} \gamma_{jk} = 0$		W		Ja/St
	3D2.3			$A_y A_y f$	$\sum b_i \gamma_{ij} \gamma_{jk} = 0$		W		Ja/St
	3D3.0			$f_{yz}(f, (-g_z^{-1}) g_y f)$	$\sum b_i \alpha_{ij} \alpha_{ik} \omega_{kl} \alpha_{lm} = 1/3$				

Table 4.2: Conditions Required for Order 3 (2 of 3).

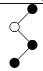
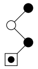
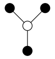
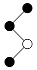
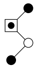
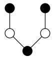
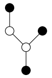
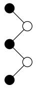
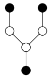
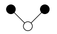
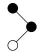
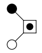
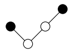
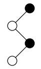
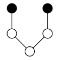
				Differential Part		
$\rho(t)$	No.	DAT_y	$ADAT_y^{DA}$	Differential	Condition	Comment
3	3D4.0			$f_y f_z(-g_z^{-1}) g_y f$	$\sum b_i \alpha_{ij} \alpha_{jk} \omega_{kl} \alpha_{lm} = 1/6$	
	3D4.1			$A_y f_z(-g_z^{-1}) g_y f$	$\sum b_i \gamma_{ij} \alpha_{jk} \omega_{kl} \alpha_{lm} = 0$	
	3D5.0			$f_z(-g_z^{-1}) g_{yy}(f, f)$	$\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} \alpha_{km} = 1/3$	Re/Ro/St Ja/St
	3D6.0			$f_z(-g_z^{-1}) g_y f_y f$	$\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} \alpha_{lm} = 1/6$	
	3D6.1			$f_z(-g_z^{-1}) g_y A_y f$	$\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} \gamma_{lm} = 0$	
	3D7.0			$f_{zz}((-g_z^{-1}) g_y f, (-g_z^{-1}) g_y f)$	$\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} \alpha_{im} \omega_{mn} \alpha_{np} = 1/3$	
	3D8.0			$f_z(-g_z^{-1}) g_{zy}((-g_z^{-1}) g_y f, f)$	$\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} \omega_{lm} \alpha_{mn} \alpha_{kp} = 1/3$	
	3D9.0			$f_z(-g_z^{-1}) g_y f_z(-g_z^{-1}) g_y f$	$\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} \alpha_{lm} \omega_{mn} \alpha_{np} = 1/6$	
	3D10.0			$f_z(-g_z^{-1}) g_{zz}((-g_z^{-1}) g_y f, (-g_z^{-1}) g_y f,)$	$\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} \omega_{lm} \alpha_{mn} \alpha_{kp} \omega_{pq} \alpha_{qr} = 1/3$	

Table 4.2: Conditions Required for Order 3 (3 of 3).

				Algebraic Part		
$\rho(u)$	No.	DAT_z	$ADAT_z^{DA}$	Differential	Condition	Comment
1	1A1.0			$(-g_z^{-1})g_y f$	$\sum b_i\omega_{ij}\alpha_{jk} = 1$	
2	2A1.0			$(-g_z^{-1})g_{yy}(f,f)$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{jl} = 1$	Re/Ro/St Ja/St
	2A2.0			$(-g_z^{-1})g_y f_y f$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{kl} = 1/2$	
	2A2.1			$(-g_z^{-1})g_y A_y f$	$\sum b_i\omega_{ij}\alpha_{jk}\gamma_{kl} = 0$	
	2A3.0			$(-g_z^{-1})g_{yz}(f,(-g_z^{-1})g_y f)$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{jl}\omega_{lm}\alpha_{mn} = 1$	
	2A4.0			$(-g_z^{-1})g_y f_z(-g_z^{-1})g_y f$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{kl}\omega_{lm}\alpha_{mn} = 1/2$	
	2A5.0			$(-g_z^{-1})g_{zz}((-g_z^{-1})g_y f,(-g_z^{-1})g_y f)$	$\sum b_i\omega_{ij}\alpha_{jk}\omega_{kl}\alpha_{lm}\alpha_{jn}\omega_{np}\alpha_{pq} = 1$	

3. When satisfying the order conditions necessarily required, Taylor expansions of the numerical solution by ROW-type method (2.3) are not affected in any way by the choice of Jacobian entries A_z. This is because corresponding non-exact elementary differentials cancel automatically due to Proposition 4.6.

4. As a consequence of point 2. and point 3. given above, it is theoretically possible to consider versions of ROW-type method (2.3) that are characterized by $A_z = 0$ and $B_y = 0$ directly in order to exploit sparse Jacobian structures.

5. Due to item 3 of Remark 4.10 order conditions must be taken into account that are represented by singly branched fat vertices. Corresponding tree structures generally cancel out for ROW-type schemes presented in literature. An exception might be given by schemes introduced by Strehmel and Weiner [77, 78, 79] that, however, did not consider a derivation based on rooted trees and did not express conditions explicitly. Hence, order conditions belonging to these tree structures have not been defined explicitly in literature so far. So, they are interpreted to be new conditions.

6. Note that due to the Jacobian approximations applied in ROW-type method (2.3) order conditions related to tree structures including singly branched fat vertices are not the same as for other ROW-type methods defined in literature. As a consequence, the order conditions with respect to these trees are new but the tree structures themselves are not. This means, new order conditions are not necessarily represented by trees including new types of vertices. However, new order conditions characterized by new tree structures not considered in literature before can be identified when considering $B_z \neq (g_z)_0$.

7. The number of order conditions required for implementing ROW-type method (2.3) when assuming $B_z = (g_z)_0$ can be further reduced by considering special choices of given Jacobian entries A_y, A_z and B_y. For example, realizing a scheme of consistency order three demands to satisfy just 10 conditions when using arbitrary entries A_y and A_z together with $B_y = (g_y)_0$ (i.e. the ROW-type method given in [30], see comment Ja/St in Table 4.2) while there are just 6 conditions relevant when using $A_y = 0$ and $A_z = 0$ together with $B_y = (g_y)_0$ (i.e. the ROW-type method given in [62], see comment Re/Ro/St in Table 4.2). The reduction of conditions generally results from the fact that for entries equal to zero given non-exact differentials cancel automatically without having to satisfy corresponding conditions. Furthermore, using exact entries $B_y = (g_y)_0$ enables to apply Proposition 1 given in [30] (also Proposition 4.6 given in [64]). However, special choices of given Jacobian entries generally reduce the adaptivity properties of the ROW-type method with respect to implicit and explicit strategies involved.

8. Another well-known approach that allows to reduce the number of given order conditions is to consider original Jacobian entries perturbed by terms of the form $\mathcal{O}(h)$, i.e. applying time-lagged Jacobian entries. The reduction of conditions

then generally results from a combination of several conditions within subsets $\Phi(t)$ and $\Phi(u)$ while other conditions switch to higher orders.

9. Numerical computation of the Jacobian is generally realized by columns. For that reason, when applying ROW-type method (2.3) with $B_z = (g_z)_0$ Jacobian entries $A_z = (f_z)_0$ can be computed along without increasing the computational effort significantly. Hence, there is no reason for regarding $A_z = (f_z) + \mathcal{O}(h)$ or $A_z = 0$ as long as $B_z = (g_z)_0$. As a consequence, when realizing corresponding ROW-type methods, schemes of practical interest would probably reduce the the possibilities of regarding non-exact Jacobian entries to components A_y and B_y. In this context, relevant schemes could be characterized by $A_z = (f_z)_0$, $B_z = (g_z)_0$ together with:

• $A_y = 0$	$B_y = 0$
• $A_y = (f_y)_0$	$B_y = (g_y)_0$
• $A_y = (f_y)_0 + \mathcal{O}(h)$	$B_y = (g_y)_0 + \mathcal{O}(h)$
• $A_y =$ arbitrary	$B_y =$ arbitrary

4.3 Convergence

After having determined the order conditions for consistency of order p, convergence of order p can be assured. For this purpose, item b) of Theorem 4.1 must be satisfied. This especially requires to determine the contractivity number α. Following descriptions given in [62] and [64] it holds:

Theorem 4.3 *Regarding ROW-type method (2.3) with $B_z = (g_z)_0$, the contractivity number α is given by*

$$\alpha = \underset{\tilde{z}\to\infty}{lim} |R(\tilde{z})| \tag{4.2}$$

with $R(\tilde{z})$ being the stability function of classical ROW methods applied to the Dahlquist test equation $y' = \lambda y$, $y(0) = 1$ and $\tilde{z} = h\lambda$ where $\lambda \in \mathbb{C}$.

Proof *The proof is given in analogy to [64] assuming the scalar case, i.e. $dim(z) = 1$. Defining $b = (b_1, ..., b_s)^T$ and $(k^{alg})_z = \left(\frac{\partial k_1^{alg}}{\partial z}(0), ..., \frac{\partial k_s^{alg}}{\partial z}(0)\right)^T$, the limit case of b) in Theorem 4.1 yields*

$$\alpha = \left\| \frac{\partial \mathcal{G}(y, z, h = 0)}{\partial z} \right\| = 1 + b^T (k^{alg})_z. \tag{4.3}$$

In this context, (4.3) corresponds to case of classical ROW methods with exact Jacobian applied to DAEs (see equation (4.13.1) in [64]). Now, after division by h, the second line given in (2.3b) reads

$$0 = g(v_i, w_i) + \sum_{j=1}^{i} \gamma_{ij} \left(B_y k_j + B_z k_j^{alg}\right) \qquad for \qquad i = 1, ..., s \tag{4.4}$$

with

$$v_i = y_0 + \sum_{j=1}^{i-1} \alpha_{ij} k_j \qquad \text{and} \qquad w_i = z_0 + \sum_{j=1}^{i-1} \alpha_{ij} k_j^{alg}.$$

When assuming $B_z = (g_z)_0$, differentiation of (4.4) with respect to z_0 (i.e. derivation by z and computing the result at $h = 0$ [64]) finally yields

$$0 = (g_z)_0 \left(1 + \sum_{j=1}^{i} \beta_{ij} \frac{\partial k_j^{alg}}{\partial z}(0) \right) \qquad for \qquad i = 1, ..., s \tag{4.5}$$

where $\beta_{ij} = \alpha_{ij} + \gamma_{ij}$. The term including non-exact Jacobian entries B_y cancels due to the definition of k_i given by the first row of (2.3b). Note that equation (4.5) corresponds to case of classical ROW methods for DAEs [64]. Next, (4.5) is divided by $(g_z)_0$ and afterwards transferred into the form of matrices and vectors. By considering $\mathcal{B} = (\beta_{ij})_{i=1,...,s}^{j=1,...,i-1}$ and $e = (1, ..., 1)^T \in \mathbb{R}^s$ this yields

$$\vec{0} = e + \mathcal{B}(k^{alg})_z$$

and, thus, after solving for $(k^{alg})_z$:

$$(k^{alg})_z = -\mathcal{B}^{-1} e. \tag{4.6}$$

Due to the lower triangular structure of $\mathcal{B}$ the given inverse can be expressed alternatively by $\mathcal{B}^{-1} = \frac{1}{\gamma} \sum_{j=0}^{s-1} (-\frac{1}{\gamma}\mathcal{B})^j$ [62]. Hence, (4.6) corresponds to

$$(k^{alg})_z = \sum_{j=1}^{s} (-1)^j \frac{1}{\gamma^j} \mathcal{B}^{j-1} e \tag{4.7}$$

which equals equation (4.13.2) given in [64]. As the stability function of classical ROW methods applied to the Dahlquist test equation satisfies [64]

$$\lim_{\tilde{z} \to \infty} |R(\tilde{z})| = 1 + \sum_{j=1}^{s} (-1)^j \frac{1}{\gamma^j} b^T \mathcal{B}^{j-1} e.$$

Theorem 4.3 can finally be confirmed by inserting (4.7) into (4.3).

Remark 4.12

- *Note that in literature it is common to denote the stability function by $R(z)$. However, $R(\tilde{z})$ is used here in order to avoid a repeated use of parameter z.*
- *The proof of Theorem 4.3 is reduced to the scalar case only because a proof regarding the multidimensional case turned out to be quite complex and could not be finished with reliable results before the due date of the thesis.*

- *Theorem 4.3 corresponds to the findings introduced by Roche [64] with respect to ROW methods for DAEs assuming* $A_y = (f_y)_0$*,* $A_z = (f_z)_0$*,* $B_y = (g_y)_0$*,* $B_z = (g_z)_0$ *as well as findings introduced by Rentrop, Roche and Steinebach [62] with respect to ROW methods for DAEs assuming* $A_y = 0$*,* $A_z = 0$*,* $B_y = (g_y)_0$*,* $B_z = (g_z)_0$*.*

As pointed out by Roche in [64] (see Remark 4.14 therein) the limit case of the stability function given by $R(\infty)$ must be smaller than 1. This is because DAEs of the form (2.1) correspond to a limit case of the singularly perturbed ODE $\varepsilon z' = g(z)$ by assuming $\varepsilon \to 0$ (respectively $\lambda \to \infty$). As a consequence, item *b*) of Theorem 4.1 is satisfied by the results of Theorem 4.3. So, the ROW-type method (2.3) which is characterized by non-exact Jacobian entries yields convergence of order p when it is consistent of order p and $B_z = (g_z)_0$.

Remark 4.13

- *Rentrop et al. [62] mentioned that condition (4.2) with* $\alpha < 1$ *is weaker than the A-stability condition*

$$|R(\tilde{z})| < 1 \qquad \forall \tilde{z} \textit{ with } Re(\tilde{z}) < 0. \tag{4.8}$$

- *Rentrop et al. [62] also mentioned that (4.8) leads to (4.2) when the test equation* $y' = \lambda y$ *degenerates to* $y = 0$ *after formulating* $\frac{1}{\lambda}y' = y$ *and assuming* $\lambda \to \infty$*.*

Remark 4.14 *Alternative approaches for analyzing the convergence of classical ROW methods with exact Jacobian applied to DAEs are presented by Hairer and Wanner [24]. These are more related to analyses introduced by Deuflhard et al. [10] regarding general one-step methods. However, corresponding approaches will not be detailed here.*

4.4 Stability

Realizing an appropriate stability analysis for linearly implicit ROW-type schemes with non-exact Jacobian applied to DAEs is complicated. In fact, in literature there seem to be no detailed stability concepts for DAE problems existent so far. At least, Ascher and Petzold give some comments regarding the stability with respect to linear index-1 problems in [1]. These are considered to be stable, if

- the system can be transformed into a semi-explicit problem and afterwards into a corresponding ODE system,
- the given transformations are all well conditioned,
- the obtained ODE system is stable.

Indeed, the stability analysis for index-1 DAEs solved by linearly implicit one-step schemes is usually reduced to investigations regarding the standard ODE test equation by Dahlquist. With respect to ROW-type methods, corresponding examples can be found in [24], [41], and [64]. In this context, the stability function of ROW methods for ODEs based on exact Jacobian entries is generally known when applied to Dahlquist's test equation. It reads [24]

$$R(\tilde{z}) = 1 + \tilde{z} b^T (I - \tilde{z}\mathcal{B})^{-1} e \tag{4.9}$$

Properties of this equation are especially known by diagonally-implicit Runge-Kutta schemes [37].

However, when including non-exact Jacobian entries determining an appropriate stability function becomes difficult. According to Hairer and Wanner [24] this is especially due to the fact that the exact and numerical Jacobian cannot be diagonalized simultaneously. As a consequence, the application of scalar test equations such as the one introduced by Dahlquist is not justified [24]. For that reason, stability analyses of ROW-type methods with non-exact Jacobian are usually realized by considering special assumptions. Regarding classical W methods applied to ODEs, these assumptions include the possibility of diagonalizing the exact and numerical Jacobian simultaneously by considering non-exact Jacobian entries that are close to the original ones [24, 55, 71].

As the ROW-type method (2.3) reduces to a classical W method when regarding the ODE case, stability analyses known for W methods can be considered. Detailed descriptions for that purpose can especially be found in [22], [71], [55] and [79]. In fact, after rewriting the classical W method for ODEs into the form [55]

$$\begin{aligned} y_1 &= y_0 + h \sum_{i=1}^{s} b_i k_i \\ (I - h\gamma A_y) k_i &= f(y_1^i) + h A_y \sum_{j=1}^{i-1} \gamma_{ij} k_j, \qquad i = 1, ..., s \end{aligned}$$

with

$$y_1^i = y_0 + h \sum_{j=1}^{i-1} \alpha_{ij} k_j$$

the stability function of a W method can alternatively be expressed by

$$R(\tilde{z}) = 1 + \sum_{k=0}^{s-1} b^T \mathcal{B}^k e \tilde{\omega}^{k+1}$$

when applying the resulting scheme to Dahlquist's test equation $y' = \lambda y$ with $\lambda \in \mathbb{C}$ assuming $A_y = \lambda$ [55]. Within the resulting stability function there is $\tilde{\omega} = (I - \gamma \tilde{z})^{-1} \tilde{z}$ with $\tilde{z} = hA_y$. Details regarding the derivation of this stability function can be found in [79].

Some references analyze the stability of W methods by regarding (4.9) directly. Due to the equality of the given stability function (4.9) to the results known for Runge-Kutta methods that are diagonally implicit and singly diagonally implicit (by regarding

$\gamma_{ii} = \gamma \; \forall i$) corresponding characteristics then can be applied straightforward. In this context, details given below correspond to information presented by Lang and Verwer in [37].

In general, W methods with $A_y = \lambda$ that are characterized by this stability function are A-stable when the corresponding stability domain $S = \{\tilde{z} \in \mathbb{C} : |R(\tilde{z})| \leq 1\}$ is a subset of $\mathbb{C}^- = \{\tilde{z} \in \mathbb{C} : Re(\tilde{z}) \leq 0\}$, i.e. the left complex half-plane. They are L-stable if $R(-\infty) = 0$ holds in addition [37]. Moreover, when the given W method is of order p its stability function corresponds to a rational function that satisfies [37]

$$e^{\tilde{z}} - R(\tilde{z}) = C\tilde{z}^{p+1} + \mathcal{O}(\tilde{z}^{p+2}) \quad \text{for} \quad \tilde{z} \to 0 \tag{4.10}$$

with $C \neq 0$ being the so-called error constant. The rational function reads [37]

$$R(\tilde{z}) = \frac{P(\tilde{z})}{(1-\gamma\tilde{z})^s} \quad \text{with} \quad P(\tilde{z}) = det(I - \tilde{z}\mathcal{B} + \tilde{z}eb^T) \tag{4.11}$$

where $P(\tilde{z})$ is at most a polynomial of degree s. When defining $P(\tilde{z}) = \sum_{i=0,\dots,s} a_i \tilde{z}^i$ L-stability of the given method is realized by setting $a_s = 0$. This can generally be achieved by an appropriate choice of entries regarding matrix $\mathcal{B}$ and vector b [37]. In fact, regarding orders $p \geq s-1$, coefficients a_i for $i \neq s$ as well as the error constant C can be uniquely calculated in dependence of γ. It holds [37]:

$$a_i = (-1)^s L_s^{(s-i)}\left(\frac{1}{\gamma}\right)\gamma^i \quad \text{with} \quad i = 0, ..., s-1 \tag{4.12}$$

and

$$C = (-1)^s L_s\left(\frac{1}{\gamma}\right)\gamma^s.$$

L_s therein denotes the Laguerre polynomial of degree s which is defined by [37]:

$$L_s(y) = \sum_{j=0}^{s} (-1)^j \binom{s}{j} \frac{y^j}{j!}.$$

Besides, terms $L_s^{(k)}$ given in (4.12) correspond to the kth derivatives. The given results show that regions of L-stability as well as (small) error constants can be determined by variations of γ [37]. The general formulation of this rational function is also discussed by Hairer and Wanner in [24]. In fact, Table 6.4 of Section IV.6 in [24] details values of γ that are required in order to realize L-stability.

Besides A-stability and L-stability, the terms strongly A-stable, A(α)-stable and A(0)-stable are common regarding Runge-Kutta methods and, thus, also applied to the family of ROW-type schemes (see e.g. [56] and [71]). In this context, a Runge-Kutta method is strongly A-stable, if [81]

$$\lim_{Re(\tilde{z}) \to -\infty} |R(\tilde{z})| < 1.$$

It is considered to be A(α)-stable when [81]

$$|R(\tilde{z})| \leq 1 \quad \text{for all} \quad \tilde{z} \in \mathbb{C}^- \quad \text{with} \quad |arg(\tilde{z}) - \pi| \leq \alpha$$

where $\alpha \in (0, \pi/2)$ and $\mathbb{C}^- = \{\tilde{z} \in \mathbb{C} : \; Re(\tilde{z}) \leq 0\}$. It is considered to be A(0)-stable, if [81]

$$|R(\tilde{z})| \leq 1 \quad \text{for} \quad \tilde{z} \in \mathbb{R}^-.$$

Remark 4.15

- *Another stability term is L(α)-stable (also: stiffly stable). The definition with respect to W methods can be found in [71]. However, it will not be further detailed here as this term is rarely used compared to terms dealing with A-stability.*
- *Compared to implicit Runge-Kutta methods linearly implicit schemes such as ROW methods and W methods cannot have strong contractivity properties such as B-stability or algebraic stability [22].*
- *Recent analyses of stability aspects with respect to W methods are especially considered by González-Pinto et al. in [16], [17] and [18]. However, these generally consider special cases solving parabolic PDE problems in the context of approximate-matrix-factorization (AMF). In this context, alternative test functions are considered.*
- *Further detailed stability analyses of ROW-type methods for DAEs that are characterized by special choices of non-exact Jacobian entries can be found in the works by Strehmel and Weiner such as [77] and [79].*

5 Sets of Coefficients

Based on findings presented before, ROW-type schemes convergent of order two and three when solving DAEs by means of non-exact Jacobian entries are realized within this section. In this context, new sets of coefficients with different properties are introduced. Most of them generally correspond to enhanced versions of familiar schemes from literature and were derived analytically. Nevertheless, some schemes that consist of an increased number of internal stages were determined numerically due to the increased difficulty of solving the resulting non-linear equation systems.

In addition, taking into account special properties such as stiffly accurate behavior allows to identify further rules for saving some of the order conditions given. Corresponding lemmas are formulated and proven. However, ROW-type schemes are known to show effects of order reduction when being applied to very stiff ODE and DAE problems as well as systems that result from semi-discretization of parabolic PDEs. The schemes derived are intended to be applied to such problems. Hence, besides satisfying the order conditions to allow for non-exact Jacobian representations, the new sets of coefficients are constructed such that they consider supplemental conditions known from literature which reduce these effects. Additional conditions for this purpose especially have been introduced by Scholz [68], Ostermann and Roche [49], Lubich and Ostermann [42] and, most recently, Rang [59, 60].

This section is organized as follows: First, some general aspects regarding additional conditions required to reduce effects of order reduction are summarized. Afterwards, schemes of order two based on three, four and five internal stages are introduced. Finally, schemes of order three with seven and eight internal stages are presented.

5.1 Additional Conditions

ROW-type schemes are known to show effects of order reduction when being applied to stiff ODEs or DAEs [59]. This phenomenon is also familiar to the solution of semi-discretized parabolic PDEs [60]. In order to overcome this problem additional conditions can be satisfied. In the following, some of these conditions are briefly summarized as they are used for constructing new sets of coefficients that realize ROW-type method (2.3). In this context, there will be often a reduction to the conditions for realizing a scheme of at most order three, as higher order schemes are not considered within this thesis.

Regarding the solution of very stiff ODE systems, Scholz [68] was among the first who expressed additional conditions that effectively reduce the occurring effects of order reduction. For this purpose, he analyzed the test problem of Prothero and Robinson [54]:

$$\dot{u}(x) = \lambda(u(x) - \varphi(x)) + \dot{\varphi}(x), \qquad u(0) = \varphi(0), \qquad \lambda \ll 0. \tag{5.1}$$

The exact solution of this problem reads $u(x) = \varphi(x)$. Note that this ODE becomes very stiff when λ is very small [60]. Due to the relevance of the test problem by Prothero and Robinson regarding the theory of B-convergence introduced by Frank, Schneid and Ueberhuber [11], Scholz adapts this concept slightly by introducing the concept of B_{PR}-convergence. For details on the concept of B_{PR}-convergence and the additional order conditions see [68].

With respect to the analysis of semi-discretized linear (parabolic) PDEs, Ostermann and Roche used the Prothero-Robinson test problem and the concept of B_{PR}-convergence to introduce their own conditions for reducing effects of order reduction in [49]. Regarding schemes that are at most B_{PR}-convergent of order $\tilde{p} = 3$, these conditions can be expressed by [49, 25]

$$b^T \mathcal{B}^i \left(\alpha^2 - 2\mathcal{B}^2 e\right) = 0 \qquad \text{with} \qquad 0 \leq i \leq s-1 \tag{5.2}$$

where $\alpha^2 = (\alpha_1^2, ..., \alpha_s^2)^T$ and $e = (1, ..., 1)^T \in \mathbb{R}^s$. Note that the order conditions by Ostermann and Roche cover those introduced by Scholz [49]. For more details see [49].

The theory by Ostermann and Roche given in [49] with respect to linear (parabolic) PDEs was enhanced by the theory of Lubich and Ostermann [42] that included the non-linear case. As shown in [35] these conditions are equal to (5.2) when realizing schemes of order three at most.

The Prothero-Robinson test problem has been further analyzed by Rang lately (see especially [57] - [60]). In this context, he introduces conditions that are more general than those considered in [68], [49] and [42] because he takes into account additional terms [59]. For more details see [59] and [60]. Below, only the final results by Rang are repeated.

Theorem 5.1 *(Rang, [60]) A ROW-type method is B_{PR}-consistent of order $\tilde{p}$ if the following conditions are satisfied:*

$$b^T \mathcal{B}^{-1} \alpha^k = 1 \tag{5.3}$$

$$b^T \mathcal{B}^{-(l+1)} \frac{1}{k-l} \alpha^{k-l} = b^T \mathcal{B}^{-l} \left[\alpha^{k-l-1} + \gamma \delta_{k-l-1,1}\right] \tag{5.4}$$

where $\delta_{i,j}$ is the Kronecker delta. Equation (5.3) must be considered for $k = 2, ..., \tilde{p}$. Equation (5.4) must be considered for $l = \max(1, k - \tilde{p}), ..., k-2$ and $k = 3, ..., \infty$.

Theorem 5.2 *(Rang, [60]) A ROW-type method is B_{PR}-convergent of order $\tilde{p}$ if it is A-stable with $R(\infty) < 1$ and B_{PR}-consistent of order $\tilde{p}$.*

In [59] Rang comments that equation (5.3) is equal to an order condition that must be satisfied by ROW methods for solving index-1 DAEs with order $p \geq 3$. Moreover, this condition is automatically satisfied for all $k \geq 2$ when the given ROW method is stiffly accurate [59]. Also, Rang states the relevance of satisfying condition (5.4) for all $k - l = \tilde{p}$ in order to avoid the dominance of undesired error terms [59].

In [60] Rang notes that with respect to ROW methods of second order condition (5.4) for $k = l - 2$ coincides with the conditions introduced by Scholz [68] as well as Lubich and Ostermann [42]. This is because only equations of the form

$$b^T \mathcal{B}^{-(l+1)} \alpha^2 = 2 b^T \mathcal{B}^{1-l} e$$

should be required to realize a scheme that is B_{PR}-convergent of order $\tilde{p} = 2$ [60]. In [59] Rang also states that condition (5.3) and (5.4) are more general than those presented in [68] and [42]. Finally, he comments in [59] that a ROW method is automatically B_{PR}-consistent of order $\tilde{p} = 2$ when it satisfies the order conditions

$$2b^T \mathcal{B}^k e = b^T \mathcal{B}^{k-2} \alpha^2 \qquad \text{for} \qquad k = 2, ..., p-1$$

by Ostermann and Roche [49].

Remark 5.1 *Note that the given theory refers to ROW methods using the exact Jacobian. For ROW-type methods using the non-exact Jacobian special conditions must be considered to estimate the error-bounds and the convergence. Regarding the application to non-linear parabolic PDEs corresponding results can be found in the paper by Lubich and Ostermann [42]. Due to their complexity they will not be considered below. Instead, only the conditions for realizing B_{PR}-convergence of ROW methods using the exact Jacobian will be taken into account.*

5.2 Analytically Determined Sets

In the following, sets of coefficients are derived that are characterized by different properties for application. In this context, schemes that realize ROW-type method (2.3) up to order $p = 2$ with respect to arbitrary Jacobian entries are generally constructed. Nevertheless, some of the resulting sets of coefficients are able to reach order $p = 3$ with respect to special Jacobian approximations that correspond to the realization of classical ROW-type schemes known from literature. Regarding B_{PR}-consistency, order conditions to reach $\tilde{p} = 2$ and $\tilde{p} = 3$ are taken into account. The sets of coefficients can be realized using $s = 3$ or $s = 4$ internal stages and, thus, be computed analytically. For this purpose, existing schemes recently described in [59] and [60] are considered and extended.

5.2.1 Order Two with Three Stages

Below, sets of coefficients for implementing the ROW-type scheme (2.3) up to order $p = 2$ are derived that can be realized by considering the minimum number of internal stages. In this context, the following theorem holds.

Theorem 5.3 *Realizing ROW-type method (2.3) up to convergence order $p = 2$ requires a minimum of $s = 3$ stages.*

Proof *To reach order $p = 2$ condition (2D2.0) is required to be unequal zero. However, this can only be achieved by taking into account $s = 3$ stages at least.*

When considering $s = 3$ internal stages, the order conditions (1D1.0) - (2D2.0) and (1A1.0) which are required for realizing the ROW-type method (2.3) up to order $p = 2$

can be expressed by:

$$
\begin{aligned}
b_1 + b_2 + b_3 &= 1 && \text{(1D1.0')}\\
b_2\alpha_2 + b_3\alpha_3 &= \frac{1}{2} && \text{(2D1.0')}\\
b_2\beta_2 + b_3\beta_3 &= \frac{1}{2} - \gamma && \text{(2D1.1')}\\
b_3\alpha_{32}\alpha_2 &= \frac{\gamma}{2} && \text{(2D2.0')}\\
b_3\omega_{32}\alpha_2 &= 1 - \frac{1}{2\gamma} && \text{(1A1.0')}
\end{aligned}
$$

Here, $\alpha_i = \sum_{j=1}^{i-1} \alpha_{ij}$ and $\gamma_i = \sum_{j=1}^{i-1} \gamma_{ij}$ together with $\beta_i = \alpha_i + \gamma_i$ are used to abbreviate occurring summations. Also, $\gamma_{ii} = \gamma$ and $\omega_{ii} = 1/\gamma$ is taken into account. Note that with these relations the formulation of (2D1.1') is a consequence of subtracting (1D1.0) times γ from (2D1.1) and afterwards adding (2D1.0). The formulation of (1A1.0') is a consequence of subtracting (2D1.0) times $1/\gamma$ from (1A1.0). The resulting system considers five equations and ten parameters. Hence, there are five degrees of freedom that can be used for realizing an appropriate set of coefficients.

GROW2

The given degrees of freedom allow for realizing sets of coefficients that satisfy several additional conditions to improve properties of the resulting scheme. However, to prove that the finally resulting order of convergence is a consequence of satisfying conditions (1D1.0') - (1A1.0') alone, there should be a set of coefficients without special characteristics derived first. For this purpose, ROS2 [37, 88] is enhanced to an appropriate version of ROW-type scheme (2.3) below.

ROS2 corresponds to a W method for ODEs of order $p = 2$ that is realized by $s = 2$ stages. It is L-stable but not stiffly accurate. The stability function regarding $s = 2$ stages reads [88]:

$$
R(\tilde{z}) = \frac{1 + (1 - 2\gamma)\tilde{z} + (\frac{1}{2} - 2\gamma + \gamma^2)\tilde{z}^2}{(1 - \gamma\tilde{z})^2} \tag{5.5}
$$

There is A-stability for $\gamma \geq 1/4$ [88]. To reach L-stability, i.e. $R(\infty) = 0$, $\gamma = 1 \pm 1/\sqrt{2}$ is required so that the highest coefficient of the numerator becomes zero [37, 88]. As a consequence, there are two different realizations of this method based on the value γ considered. While Verwer et al. consider ROS2 with $\gamma_+ = 1 + 1/\sqrt{2}$ in [88], Lang and Verwer realize ROS2 with $\gamma_- = 1 - 1/\sqrt{2}$ in [37]. In the following, the version with γ_- is considered as it minimizes the error constant [37]. Note that there are no additional conditions required to solve index-1 DAE systems by means of exact Jacobian entries, i.e. a W method of order $p = 2$ for ODEs automatically corresponds to a ROW method of order $p = 2$ for index-1 DAEs. However, as ROS2 considers just $s = 2$ stages it is not able to satisfy condition (2D2.0'). It violates condition (1A1.0'), too. Hence, when being applied to solve index-1 DAEs by means of non-exact Jacobian entries, this method has an order reduction down to $p = 1$.

In order to extend ROS2 to generalized ROW-type method (2.3) a third stage is required. This affects the stability function. By (4.9) it reads for $s = 3$

$$R(\tilde{z}) = 1 + \frac{(b_1 + b_2 + b_3)\tilde{z}}{(1 - \gamma\tilde{z})} + \frac{(b_2\beta_2 + b_3\beta_3)\tilde{z}^2}{(1 - \gamma\tilde{z})^2} + \frac{b_3\beta_{32}\beta_2\tilde{z}^3}{(1 - \gamma\tilde{z})^3} \tag{5.6}$$

and, thus, by including conditions (1D1.0') and (2D1.1')

$$R(\tilde{z}) = \frac{1 + (1 - 3\gamma)\tilde{z} + (\frac{1}{2} - 3\gamma + 3\gamma^2)\tilde{z}^2 + (-\gamma^3 + 2\gamma^2 - \frac{1}{2}\gamma + b_3\beta_{32}\beta_2)\tilde{z}^3}{(1 - \gamma\tilde{z})^3}. \tag{5.7}$$

Nevertheless, the objective is to realize a L-stable set with coefficients comparable to those of ROS2. In order to realize a L-stable scheme, the highest coefficient occurring for the polynomial of the numerator is required to become zero [37]. To achieve this while using the same value of γ as for ROS2 the condition

$$b_3\beta_{32}\beta_2 = \gamma^3 - \gamma^2 - \frac{3}{2}\gamma + \frac{1}{2} \qquad \text{(AC1)}$$

must be satisfied. This ensures that the term in front of $\tilde{z}^3$ regarding the numerator of (5.7) reduces to $\frac{1}{2} - 2\gamma + \gamma^2$, i.e. the same term as given for the highest polynomial within the numerator of (5.5). When satisfying condition (AC1) the given stability function is A-stable for $\gamma \geq 0.2575406722426549$.

Note that condition (AC1) contradicts (1A1.0') when using $\alpha_2 = 1$ and $\gamma_{21} = -2\gamma$ together with $b_2 = 1/2$ as suggested by Lang and Verwer constructing ROS2 in [37]. For that reason, either α_2 or γ_{21} should be altered. In the following, let $b_2 = 1/2$ and $\alpha_2 = 1$. Assuming $b_3 \neq 0$ (as $b_3 = 0$ would contradict (2D2.0'), (1A1.0') and (AC1)) it follows $\alpha_3 = 0$ from (2D1.0') directly. Because one degree of freedom is left, let $\alpha_{31} = 1/2$. This leads to $\alpha_{32} = -1/2$ and, thus, to $b_3 = -\gamma$ by (2D2.0').

The given parameters can now be used to determine the remaining coefficients. From (1A1.0') there is

$$\beta_{32} = \gamma - \frac{1}{2}. \tag{5.8}$$

So, it follows $\gamma_{32} = \gamma$ by taking into account α_{32}. Also, by using (AC1), there is

$$\beta_{32}(1 + \gamma_{21}) = -\gamma^2 + \gamma + \frac{3}{2} - \frac{1}{2\gamma}. \tag{5.9}$$

Hence, when dividing (5.9) by (5.8) the missing value γ_{21} can be determined. The result is $\gamma_{21} = -1$. The value of γ_{31} can be determined by computing β_{31} via (2D1.1') and afterwards subtracting $\alpha_{31} = 1/2$. The result is $\gamma_{31} = -1$ as well. Finally, $b_1 = 1/2 + \gamma$ is determined via (1D1.0').

The embedded method is required to be of order $p = 1$ only. Hence, just condition (1D1.0') must be satisfied. The condition thus reads:

$$\hat{b}_1 + \hat{b}_2 + \hat{b}_3 = 1.$$

Table 5.1: Set of Coefficients for GROW2

$\gamma = 2.92893218813452 43\text{E}{-01}$					
α_{21}	$=$	$1.0000000000000000\text{E}{+00}$	γ_{21}	$=$	$-1.0000000000000000\text{E}{+00}$
α_{31}	$=$	$5.0000000000000000\text{E}{-01}$	γ_{31}	$=$	$-1.0000000000000000\text{E}{+00}$
α_{32}	$=$	$-5.0000000000000000\text{E}{-01}$	γ_{32}	$=$	$2.9289321881345243\text{E}{-01}$
b_1	$=$	$7.9289321881345243\text{E}{-01}$	$\hat{b}_1$	$=$	$7.0000000000000000\text{E}{-01}$
b_2	$=$	$5.0000000000000000\text{E}{-01}$	$\hat{b}_2$	$=$	$7.0000000000000000\text{E}{-01}$
b_3	$=$	$-2.9289321881345243\text{E}{-01}$	$\hat{b}_3$	$=$	$-4.0000000000000000\text{E}{-01}$

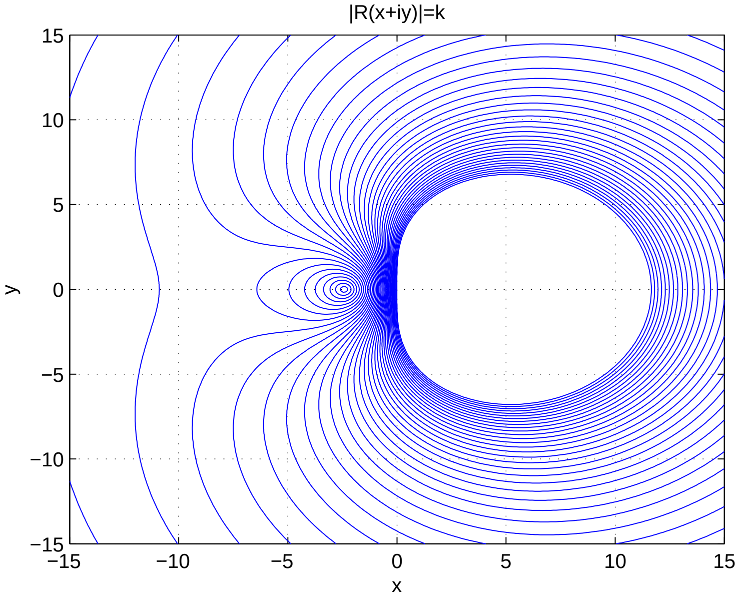


Figure 5.1: Stability region of GROW2 regarding `k=0:0.025:1`.

Note that the choice $\hat{b}_1 = 1/2$, $\hat{b}_2 = 1/2$ and $\hat{b}_3 = 0$ in order to be close to the original method ROS2 does not correspond to a stable method when there are $s = 3$ stages. For this reason, let $\hat{b}_1 = 7/10$, $\hat{b}_2 = 7/10$ and $\hat{b}_3 = -2/5$ be assumed. This way the embedded scheme yields $|\tilde{R}(\infty)| \approx 0.88$.

The resulting method is called GROW2 below. The corresponding set of coefficients is summarized in Table 5.1. Its stability region is shown in Figure 5.1.

Remark 5.2 *GROW2 is B_{PR}-consistent of order $\tilde{p} = 1$ only. This is because the given set of coefficients violates the additional order conditions that are required to achieve B_{PR}-consistency of order $\tilde{p} = 2$ (see Table 5.12).*

GROW2S

GROW2 does not satisfy additional conditions. Hence, it is expected to show effects of order reduction when solving very stiff differential equations or systems that result with respect to semi-discretized parabolic PDEs. For that reason, degrees of freedom which remain for constructing schemes of order $p = 2$ with $s = 3$ stages are used to realize enhanced characteristics below. In this context, especially stiffly accurate properties should be considered first. Stiffly accurate schemes ensure that $R(\infty) = 0$. Hence, corresponding schemes are automatically L-stable when they are A-stable. Moreover, stiffly accurate ROW-type methods yield asymptotically exact results when applied to the Prothero-Robinson test problem. Also, the numerical solution of stiffly accurate ROW-type methods with respect to z_1 equals the result of one simplified Newton iteration for $0 = g(x_0 + h, z)$ when solving DAEs [24].

Definition 5.1 *ROW-type methods whose coefficients are characterized by*

$$\beta_{si} = b_i, \quad \textit{for} \quad i = 1, ..., s \quad \textit{and} \quad \alpha_s = 1$$

are called stiffly accurate [24, 56].

ROW-type schemes which satisfy the requirements described in Definition 5.1 allow to safe one order condition. In fact, the following theorem can be considered.

Theorem 5.4 *Any stiffly accurate ROW-type method automatically satisfies order condition (1A1.0).*

Proof *Condition (1A1.0) written in matrix-vector notation generally reads*

$$b^T \mathcal{W} \alpha = 1$$

with $b = (b_1, ..., b_s)^T$, $\alpha = (\alpha_1, ..., \alpha_s)^T$ *and* $\mathcal{W} = (\omega_{ij})_{i,j=1}^s$. *There is* $\mathcal{W} = \mathcal{B}^{-1}$ *with* $\mathcal{B} = (\beta_{i,j})_{i,j=1}^s$. *Both matrices* $\mathcal{W}$ *and* $\mathcal{B}$ *are characterized by a lower triangular structure. For that reason, entries of* ω_{ij} *can be defined recursively. For* $i = 1, ..., s$ *there is*

$$\omega_{ii} = \frac{1}{\beta_{ii}} \tag{5.10a}$$

and

$$\omega_{ij} = (-1) \cdot \sum_{k=j}^{i-1} \omega_{kj} \frac{\beta_{ik}}{\beta_{ii}}. \tag{5.10b}$$

Now, when considering a stiffly accurate scheme, there is $\beta_{si} = b_i$ *for* $i = 1, ..., s$ *and* $\alpha_s = 1$. *Also,* $\beta_{ii} = \gamma$ *holds. Due to these characteristics, there will be always* $b^T \mathcal{W} = \tilde{e}$ *with* $\tilde{e} = (0, ..., 0, 1) \in \mathbb{R}^s$ *because given entries cancel appropriately. As a consequence, there is* $\tilde{e}\alpha = \alpha_s$. *Hence, for* $\alpha_s = 1$ *condition (1A1.0) is always satisfied for stiffly accurate schemes. Note, that these findings are independent of the stage number s.*

Example 5.1 *Considering a scheme with $s = 3$ stages, entries of matrix $\mathcal{W}$ become*

$$\omega_{11} = \frac{1}{\beta_{11}} \qquad \omega_{22} = \frac{1}{\beta_{22}} \qquad \omega_{33} = \frac{1}{\beta_{33}}$$

together with

$$\begin{aligned}
\omega_{21} &= -\omega_{11}\frac{\beta_{21}}{\beta_{22}} \\
\omega_{31} &= -\omega_{11}\frac{\beta_{31}}{\beta_{33}} - \omega_{21}\frac{\beta_{32}}{\beta_{33}} \\
\omega_{32} &= -\omega_{22}\frac{\beta_{32}}{\beta_{33}}
\end{aligned}$$

due to (5.10). Hence, by including the properties $\beta_{ii} = \gamma$ and $\beta_{si} = b_i$ known for stiffly accurate schemes there is

$$b^T\mathcal{W}\alpha = \left(b_1\tfrac{1}{\gamma} - b_2\beta_{21}\tfrac{1}{\gamma^2} - b_3b_1\tfrac{1}{\gamma^2} + b_3b_2\beta_{21}\tfrac{1}{\gamma^3} \;\middle|\; b_2\tfrac{1}{\gamma} - b_3b_2\tfrac{1}{\gamma^2} \;\middle|\; b_3\tfrac{1}{\gamma}\right)\begin{pmatrix}\alpha_1\\ \alpha_2\\ \alpha_3\end{pmatrix}.$$

Taking into account $b_3 = \gamma$ and $\alpha_3 = 1$ it finally follows

$$b^T\mathcal{W}\alpha = \left(0 \;\middle|\; 0 \;\middle|\; 1\right)\begin{pmatrix}\alpha_1\\ \alpha_2\\ 1\end{pmatrix} = 1.$$

Hence, condition (1A1.0) is automatically fulfilled.

Due to the assumptions given in Definition 5.1 order conditions can be significantly simplified when realizing a stiffly accurate scheme. In fact, for constructing a corresponding version of ROW-type method (2.3) that is characterized by order $p = 2$ and $s = 3$ internal stages, required order conditions (1D1.0') - (2D2.0') (taking into account Theorem 5.4) can be expressed by:

$$\begin{aligned}
b_1 + b_2 &= 1 - \gamma && \text{(1D1.0”)} \\
b_2\alpha_2 &= \frac{1}{2} - \gamma && \text{(2D1.0”)} \\
b_2\beta_2 &= \frac{1}{2} - 2\gamma + \gamma^2 && \text{(2D1.1”)} \\
\alpha_{32}\alpha_2 &= \frac{1}{2} && \text{(2D2.0”)}
\end{aligned}$$

In this context, (2D1.1”) results from (2D1.1') by considering $b_3 = \gamma$ and $\beta_3 = \beta_{31} + \beta_{32} = b_1 + b_2$. The value of β_3 is replaced by the right-hand side of (1D1.0”). The resulting system has four equations that depend on six parameters. Hence, there are two degrees of freedom left which can be used to satisfy additional conditions. Note that the stiffly accurate condition $\alpha_3 = 1$ is already considered and that it is finally used to determine α_{31}.

In order to derive a corresponding set of coefficients the method ROS2S is considered. ROS2S was introduced in [25]. It corresponds to a stiffly accurate W method of order $p = 2$ with $s = 3$ stages. Hence, it satisfies the W method conditions (1D1.0") - (2D1.1") already. Besides, ROS2S fulfills the conditions presented by Ostermann and Roche in [49] to reduce effects of order reduction (see (5.2)). In this context, compared to the conditions introduced by Rang in [59] and [60], ROS2S is B_{PR}-consistent of order $\tilde{p} = 2$ [60]. Note that ROS2S violates order condition (2D2.0") and therefore cannot reach order $p = 2$ when solving DAEs by means of non-exact Jacobian entries. Therefore, the objective is to extend ROS2S appropriately below.

To ensure that the enhanced set of coefficients preserves B_{PR}-consistency of order $\tilde{p} = 2$ it is sufficient to supplement the order conditions (1D1.0") - (2D2.0") by the conditions of Ostermann and Roche [49] (see [59, 60]). This is prefered against including the latest conditions by Rang as the original method is then extended by condition (2D2.0") only. After some reformulation, the supplementing conditions resulting from (5.2) read explicitly for $i = 0, 1, 2$ [25]:

$$b_2\alpha_2^2 - 2\gamma b_2\beta_2 = -2\gamma^2 + \gamma \qquad \text{(OR1)}$$
$$2\gamma b_2\alpha_2^2 - 6\gamma^2 b_2\beta_2 = -4\gamma^3 + 2\gamma^2 \qquad \text{(OR2)}$$
$$3\gamma^2 b_2\alpha_2^2 - 12\gamma^3 b_2\beta_2 = -6\gamma^4 + 3\gamma^3. \qquad \text{(OR3)}$$

By multiplying (OR1) with 2γ and subtracting it from (OR2) there is

$$b_2\beta_2 = 0 \tag{5.11}$$

Due to this result, $b_2 = 0$ or $\beta_2 = 0$ must be satisfied. Choosing $b_2 = 0$ would require $\gamma = 1/2$ from (2D1.0"). However, this would contradict (2D1.1") as $\gamma = 1/2$ is no root of $\gamma^2 - 2\gamma + \frac{1}{2} = 0$ (see [25]). As a consequence, $\beta_2 = 0$ must be satisfied. Inserting $\beta_2 = 0$ into (OR1) yields

$$b_2\alpha_2^2 = -2\gamma^2 + \gamma \tag{5.12}$$

and is automatically satisfied when inserting (5.12) into (OR3) [25]. The value α_2 can now be determined by dividing (5.12) by (2D1.0"). The result is $\alpha_2 = 2\gamma$. This can now be used to determine $b_2 = \frac{1}{4\gamma} - \frac{1}{2}$ by (2D1.0") and $b_1 = \frac{3}{2} - \gamma - \frac{1}{4\gamma}$ by (1D1.0"). Also, $\alpha_{32} = \frac{1}{4\gamma}$ follows from (2D2.0"). As $\alpha_3 = 1$ to achieve a stiffly accurate scheme, there must be $\alpha_{31} = 1 - \frac{1}{4\gamma}$. Finally, by (2D1.1") the choice of $\beta_2 = 0$ yields $\gamma = 1 \pm \sqrt{1/2}$. As for ROS2S let $\gamma_- = 1 - 1/\sqrt{2}$ be considered. Recall that it is the same value as for ROS2 and GROW2 which minimizes the error constant. This choice also ensures that $b_1 = b_2$.

Note that there can be no stiffly accurate embedded scheme: Due to $\alpha_2 = 1/2$ the relevant condition $\alpha_s = 1$ for an embedded scheme with $s = 2$ cannot be satisfied. A value $\alpha_2 = 1$ would be possible when $\alpha_{31} = 1/2$ and $\alpha_{32} = 1/2$ according to (2D2.0"). However, then $\hat{b}_2 = \beta_{22} = \gamma$ would be required to realize an embedded stiffly accurate scheme. This would lead to $\hat{b}_1 = 1 - \gamma$ to satisfy condition (1D1.0) which is the

Table 5.2: Set of Coefficients for GROW2S

$\gamma = 2.92893218881345243\text{E}{-01}$					
α_{21}	=	$5.85786437626904850\text{E}{-01}$	γ_{21}	=	$-5.85786437626904850\text{E}{-01}$
α_{31}	=	$1.46446609406726050\text{E}{-01}$	γ_{31}	=	$2.07106781186547910\text{E}{-01}$
α_{32}	=	$8.53553390593273950\text{E}{-01}$	γ_{32}	=	$-5.00000000000000000\text{E}{-01}$
b_1	=	$3.53553390593273950\text{E}{-01}$	$\hat{b}_1$	=	$3.33333333333333333\text{E}{-01}$
b_2	=	$3.53553390593273950\text{E}{-01}$	$\hat{b}_2$	=	$3.33333333333333333\text{E}{-01}$
b_3	=	$2.92893218881345243\text{E}{-01}$	$\hat{b}_3$	=	$3.33333333333333333\text{E}{-01}$

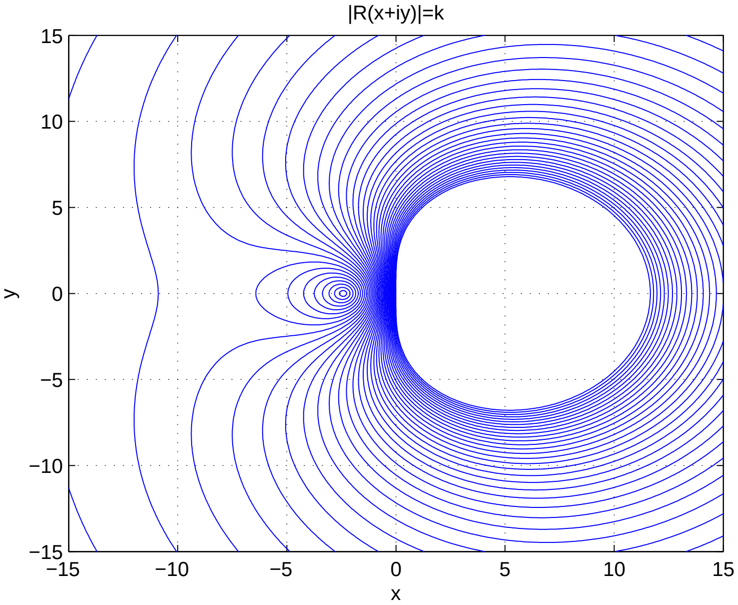


Figure 5.2: Stability region of GROW2S regarding `k=0:0.025:1`.

only condition that must be satisfied by the embedded scheme to reach order $p = 1$. However, this would contradict the requirement $\hat{b_1} = \beta_{21}$ with $\beta_{21} = 0$ to achieve a stiffly accurate embedded scheme. Hence, there is no chance for the embedded scheme to be stiffly accurate. Instead, $\hat{b}_1 = \hat{b}_2 = \hat{b}_3 = 1/3$ as for ROS2S is considered. Consider that the embedded scheme satisfies $|\tilde{R}(\infty)| \approx 0.33$.

The resulting method is called GROW2S below. The corresponding set of coefficients is summarized in Table 5.2. Its stability region is shown in Figure 5.2.

Finally, the fact that the enhanced set of coefficients determined reaches the same order of B_{PR}-consistency as the original scheme ROS2S is shown.

Lemma 5.1 *GROW2S is B_{PR}-consistent of order $\tilde{p} = 2$.*

Proof *Following the proof of B_{PR}-consistency given in [60] with respect to ROS2S it must be shown that for all $l \geq 1$*

$$b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B})e = 0.$$

In this context, $\mathcal{B}^{-k}$ for $k \geq 0$ can be determined analytically. Because of $\beta_2 = 0$ there is [60]:

$$\mathcal{B}^{-k} = \frac{1}{\gamma^{k+1}} \begin{bmatrix} \gamma & 0 & 0 \\ 0 & \gamma & 0 \\ -k\beta_{31} & -k\beta_{32} & \gamma \end{bmatrix}, \qquad k \geq 0.$$

Hence, it follows in general

$$b^T \mathcal{B}^{-l} = \frac{1}{\gamma^{l+1}} \left(b_1\gamma - lb_1b_3 \;\middle|\; b_2\gamma - lb_2b_3 \;\middle|\; b_3\gamma \right).$$

Also, as shown in [60] there is

$$\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e = \left(-2\gamma \;\middle|\; \tfrac{\alpha_2^2}{\gamma} - 2\gamma \;\middle|\; -\tfrac{b_2\alpha_2^2}{\gamma^2} + \tfrac{1}{\gamma} - 2 \right)^T.$$

By considering $\alpha_2 = 2\gamma$ for the given set of coefficients, this expression simplifies to:

$$\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e = \left(-2\gamma \;\middle|\; 2\gamma \;\middle|\; -4b_2 + \tfrac{1}{\gamma} - 2 \right)^T.$$

As a consequence

$$b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e) = \frac{1}{\gamma^{l+1}} \left(-2b_1\gamma^2 + 2lb_1\gamma^2 + 2b_2\gamma^2 - 2lb_2\gamma^2 - 4b_2\gamma^2 + \gamma - 2\gamma^2 \right).$$

After inserting $b_1 = 1 - b_2 - \gamma$ (i.e. considering condition (1D1.0”) assuming stiffly accurate schemes) the given expression can be alternatively expressed by

$$b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e) = \frac{1}{\gamma^{l+1}} \left(-4\gamma^2 + 2\gamma^3 + \gamma + l \cdot (2\gamma^2 - 2\gamma^3 - 4b_2\gamma^2) \right)$$

or by including $b_2 = \frac{1}{4\gamma} - \frac{1}{2}$

$$b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e) = \frac{1}{\gamma^{l+1}} \left(-4\gamma^2 + 2\gamma^3 + \gamma + l \cdot (4\gamma^2 - 2\gamma^3 - \gamma) \right).$$

The term given in brackets finally becomes equal to zero when

$$2\gamma^3 - 4\gamma^2 + \gamma = 0.$$

Roots of this equation are $\gamma_1 = 0$ and $\gamma_{2,3} = 1 \pm \sqrt{1/2}$. As the set of coefficients is realized by using $\gamma_- = 1 - 1/\sqrt{2}$ it follows

$$b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e) = 0$$

and therefore GROW2S proves to be B_{PR}-consistent of order $\tilde{p} = 2$.

GROW3P

The sets of coefficients previously considered are of order $p = 2$. Next, a scheme that ensures order $p = 2$ regarding the solution of DAEs with non-exact Jacobian but order $p = 3$ regarding the solution of DAEs with exact Jacobian is considered. For this purpose, ROS3P introduced by Lang and Verwer in [35] is extended to ROW-type method (2.3). ROS3P consists of $s = 3$ internal stages and reaches order $p = 3$ for nonlinear parabolic problems as well as DAEs when using exact Jacobian entries. However, to achieve the higher order by means of the given number of stages comes at a cost. ROS3P is not stiffly accurate and not L-stable. Instead, it proves to be strongly A-stable with $|R(\infty)| \approx 0.73$ [35]. Note that ROS3P violates the order conditions (2D1.0') and (2D2.0'). Hence, its order is expected to reduce to $p = 1$ when solving DAEs by means of non-exact Jacobian entries.

In order to realize an enhanced set of coefficients that preserves $p = 2$ when solving DAEs with non-exact Jacobian and $p = 3$ when solving DAEs with exact Jacobian, conditions (1D1.0') - (1A1.0') must be satisfied as well as the conditions

$$b_2\alpha_2^2 + b_3\alpha_3^2 = \frac{1}{3} \qquad \text{(3D1.0')}$$

$$b_3\beta_{32}\beta_2 = \frac{1}{6} - \gamma + \gamma^2 \qquad \text{(3D2.0')}$$

$$b_2\omega_{22}\alpha_2^2 + b_3(\omega_{32}\alpha_2^2 + \omega_{33}\alpha_3^2) = 1 \qquad \text{(2A1.0')}$$

So, conditions (1D1.0') - (1A1.0') are supplemented by the conditions of Roche [64]. Note that (3D2.0') corresponds to the sum of conditions (3D2.0) - (3D2.3) given in Table 4.2. Besides, the conditions by Lubich and Ostermann [42] to reduce effects of order reduction are taken into account. In this context, it is sufficient to satisfy (5.2) for $i = 1, 2$ only. The case $i = 0$ can be neglected as it is automatically satisfied when order conditions (1D1.0'), (2D1.1'), (3D1.0') and (3D2.0') are fulfilled. Using these conditions, the additional equations with respect to $i = 1, 2$ for realizing a scheme with $s = 3$ internal stages can be expressed by [35]:

$$b_3\beta_{32}\alpha_2^2 = \frac{1}{6} - \frac{2}{3}\gamma \qquad \text{(LO1)}$$

$$0 = \gamma^2 - \gamma + \frac{1}{6} \qquad \text{(LO2)}$$

Note that a method that any ROW-type method which satisfies the conditions (1D1.0'), (2D1.1'), (3D1.0') and (3D2.0') together with (LO1) and (LO2) automatically satisfies (2A1.0') (see Lemma 4.1 in [35]). However, according to Rang [59] the conditions (LO1) and (LO2) are not sufficient to obtain full B_{PR}-consistency of order $\tilde{p} = 3$. Hence, a corresponding scheme can be at most B_{PR}-consistent of order $\tilde{p} = 2$. Additional conditions to reach B_{PR}-consistency of order $\tilde{p} = 3$ with $s = 3$ internal stages can be found in [59]. They are not considered here. Further below, there is a proof that ROW-type schemes of the form (2.3) cannot be B_{PR}-consistent of order $\tilde{p} = 3$ as long as they consider $s = 3$ stages (see Lemma 5.3).

As (2A1.0') is automatically satisfied, the given system consists of nine equations and ten unknowns. Hence, there is one degree of freedom left. Due to the quadratic equation given by (LO2) an appropriate value for γ can be directly determined. Roots of this equation read $\gamma_{\pm} = \frac{1}{2} \pm \frac{1}{6}\sqrt{3}$. However, only $\gamma_{+} = \frac{1}{2} + \frac{1}{6}\sqrt{3}$ allows for constructing an A-stable scheme (see Tab. 6.3 in Section IV.6 of [24]). Consider that by including condition (3D2.0') stability function (4.9) (for $s = 3$ stages equal to (5.7)) becomes

$$R(\tilde{z}) = \frac{1 + (1-3\gamma)\tilde{z} + (\frac{1}{2} - 3\gamma + 3\gamma^2)\tilde{z}^2 + (-\gamma^3 + 3\gamma^2 - \frac{3}{2}\gamma + \frac{1}{6})\tilde{z}^3}{(1-\gamma\tilde{z})^3}. \tag{5.13}$$

Hence, the given value of γ leads to $|R(\infty)| \approx 0.73$ as for the original method ROS3P [35]. Equivalently, the term $b_3\beta_{32}\beta_2$ occurring in (5.7) can be neglected when satisfying (LO2) because $b_3\beta_{32}\beta_2 = 0$ by (3D2.0'). Note that due to (LO1) and (1A1.0') there is $b_3\beta_{32} \neq 0$. Hence, there is $\beta_2 = 0$ required to satisfy (3D2.0') by means of the given value γ (see [35]). Next, (LO1) divided by (1A1.0') is used to determine α_2. By taking into account $\omega_{32} = -\beta_{32}/\gamma^2$ there is $\alpha_2 = 2\gamma$. Due to $\beta_2 = 0$ there must be $\gamma_{21} = -\alpha_{21}$. So, γ_{21} is fixed. Next, for the sake of simplicity, the given degree of freedom is used to assume $\alpha_3 = 1$ as for ROS3P (see [35]). Then, conditions (2D1.0') and (3D1.0') can be used to determine

$$\frac{1}{2} - b_2\alpha_2 = \frac{1}{3} - b_2\alpha_2^2$$

As a consequence, there is

$$b_2 = \frac{\frac{1}{3} - \frac{1}{2}}{\alpha_2^2 - \alpha_2} = -\frac{1}{24\gamma^2 - 12\gamma}.$$

This result leads to

$$b_3 = \frac{1}{2} - b_2\alpha_2 = \frac{1}{2} + \frac{1}{12\gamma - 6}$$

via (2D1.0'). Now, α_{32} and β_{32} can be determined using (2D2.0') and (1A1.0'), respectively. Afterwards, α_{31} follows from $\alpha_{31} = \alpha_3 - \alpha_{32}$. With (2D1.1') the value of β_{31} can now be determined. There is:

$$\beta_{31} = \frac{1}{b_3}\left(\frac{1}{2} - \gamma\right) - \beta_{32}.$$

Finally, b_1 results from (1D1.0').

An appropriate embedded scheme, must be of order $p = 1$ when solving DAEs by means of a non-exact Jacobian and $p = 2$ when solving DAEs by means of an exact Jacobian. Corresponding order conditions read:

$$\hat{b}_1 + \hat{b}_2 + \hat{b}_3 = 1 \qquad \text{(e1D1.0')}$$

$$\hat{b}_2\beta_2 + \hat{b}_3\beta_3 = \frac{1}{2} - \gamma \qquad \text{(e2D1.1').}$$

Table 5.3: Set of Coefficients for GROW3P

$\gamma = 7.8867513459481287\mathrm{E}{-}01$					
α_{21}	$=$	1.5773502691896257E+00	γ_{21}	$=$	−1.5773502691896257E+00
α_{31}	$=$	6.8301270189221941E−01	γ_{31}	$=$	−8.6602540378443871E−01
α_{32}	$=$	3.1698729810778065E−01	γ_{32}	$=$	−5.0000000000000000E−01
b_1	$=$	3.9433756729740654E−01	$\hat{b}_1$	$=$	3.3333333333333333E−01
b_2	$=$	−1.8301270189221933E−01	$\hat{b}_2$	$=$	−1.2200846792814612E−01
b_3	$=$	7.8867513459481287E−01	$\hat{b}_3$	$=$	7.8867513459481287E−01

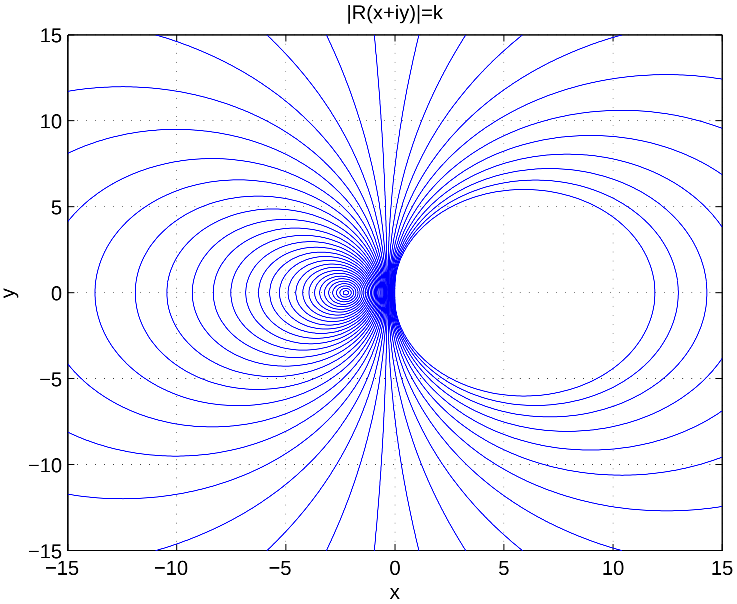


Figure 5.3: Stability region of GROW3P regarding `k=0:0.025:1`.

The system provides one degree of freedom. By the given results of β_2 and β_3 it is possible to determine $\hat{b}_3$ directly via (e2D1.1'). By choosing $\hat{b}_1 = \frac{1}{3}$ as for the original method ROS3P, $\hat{b}_2$ can finally be determined using (e1D1.0'). Note that the resulting embedded scheme satisfies $|\tilde{R}(\infty)| \approx 0.73$.

The resulting method is called GROW3P below. The corresponding set of coefficients is summarized in Table 5.3. Its stability region is shown in Figure 5.3.

Lemma 5.2 *GROW3P is B_{PR}-consistent of order $\tilde{p} = 2$.*

Proof *As for the proof with respect to Lemma 5.1 and the approach presented in [59]* *with respect to ROS2S it must be shown that for all $l \geq 1$ there is*

$$b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B})e = 0.$$

By taking into account $\beta_2 = 0$ it is possible to define

$$\mathcal{B}^{-k} = \frac{1}{\gamma^{k+l}} \begin{bmatrix} \gamma & 0 & 0 \\ 0 & \gamma & 0 \\ -k\beta_{31} & -k\beta_{32} & \gamma \end{bmatrix}, \qquad k \geq 0.$$

In addition, it holds

$$b^T \mathcal{B}^{-l} = \frac{1}{\gamma^{l+1}} \Big(b_1\gamma - lb_3\beta_{31} \;\Big|\; b_2\gamma - lb_3\beta_{32} \;\Big|\; b_3\gamma \Big)$$

and

$$\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e = \Big(-2\gamma \;\Big|\; \tfrac{\alpha_2^2}{\gamma} - 2\gamma \;\Big|\; -\tfrac{\beta_{32}\alpha_2^2}{\gamma^2} + \tfrac{\alpha_3^2}{\gamma} - 2\beta_{31} - 2\beta_{32} - 2\gamma \Big)^T.$$

For that reason, it follows

$$\begin{aligned} &b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e) \\ &= \frac{1}{\gamma^{l+1}} \bigg(-2\gamma^2 b_1 + 2l\gamma b_3(\beta_{31} + \beta_{32}) + b_2\alpha_2^2 - 2\gamma^2 b_2 - \frac{1}{\gamma} lb_3\beta_{32}\alpha_2^2 - \frac{1}{\gamma} b_3\beta_{32}\alpha_2^2 \\ &\qquad + b_3\alpha_3^2 - 2\gamma b_3(\beta_{31} + \beta_{32}) - 2\gamma^2 b_3 \bigg). \end{aligned}$$

By including (1D1.0') and (3D1.0') it follows

$$\begin{aligned} &b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e) \\ &= \frac{1}{\gamma^{l+1}} \left(\frac{1}{3} - 2\gamma^2 + 2l\gamma b_3(\beta_{31} + \beta_{32}) - \frac{1}{\gamma} lb_3\beta_{32}\alpha_2^2 - \frac{1}{\gamma} b_3\beta_{32}\alpha_2^2 - 2\gamma b_3(\beta_{31} + \beta_{32}) \right). \end{aligned}$$

Now, including $b_3\beta_3 = \frac{1}{2} - \gamma$ from (2D1.1') taking into account $\beta_2 = 0$ and $b_3\beta_{32}\alpha_2^2 = \frac{1}{6} - \frac{2}{3}\gamma$ from (LO1) the given equation becomes:

$$b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e) = \frac{1}{\gamma^{l+1}} \left(\frac{1}{3} - 2\gamma^2 + l\gamma - 2l\gamma^2 - \frac{1}{6\gamma} l + \frac{2}{3} l - \frac{1}{6\gamma} + \frac{2}{3} - \gamma + 2\gamma^2 \right).$$

Hence, there is

$$b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e) = \frac{1}{\gamma^{l+1}} \left(1 - \gamma - \frac{1}{6\gamma} + l\left(\gamma - 2\gamma^2 - \frac{1}{6\gamma} + \frac{2}{3} \right) \right).$$

Due to the choice of γ by (LO2) there is

$$1 - \gamma - \frac{1}{6\gamma} = 0.$$

Also, it can be shown that there is

$$\gamma - 2\gamma^2 - \frac{1}{6\gamma} + \frac{2}{3} = 0.$$

With $-\frac{1}{6\gamma} = \gamma - 1$ *(from (LO2)) this equation yields*

$$\gamma - \gamma^2 - \frac{1}{6} = 0$$

which, again, is satisfied by the given choice of γ*. Hence*

$$b^T \mathcal{B}^{-l}(\mathcal{B}^{-1}\alpha^2 - 2\mathcal{B}e) = 0$$

is satisfied and the scheme is B_{PR}*-consistent of order* $\tilde{p} = 2$*.*

The given set of coefficients is only B_{PR}-consistent of order $\tilde{p} = 2$ although it corresponds to a ROW-type method of order $p = 3$ when solving DAEs by means of exact Jacobian entries. Hence, it would be appropriate to realize a similar set of coefficients that is B_{PR}-consistent of order $\tilde{p} = 3$ as well. In fact, regarding the original method ROS3P, a corresponding scheme was introduced in [59]. It is called ROS3PR. ROS3PR violates conditions (2D1.0') and (2D2.0'). So, there is the idea of extending it to a suitable ROW-type methond of the form (2.3). But, contrary to ROS3P, ROS3PR cannot be extended to ROW-type method (2.3) while preserving B_{PR}-consistency of order $\tilde{p} = 3$. In fact, the following Lemma holds.

Lemma 5.3 *ROW-type schemes of the form (2.3) consisting of* $s = 3$ *stages can be at most* B_{PR}*-consistent of order* $\tilde{p} = 2$*.*

Proof *As shown in [59] a ROW-type scheme with* $s = 3$ *stages is* B_{PR}*-consistent of order* $\tilde{p} = 3$ *when it satisfies order conditions (1D1.0'), (2D1.1'), (3D1.0') and (3D2.0') plus the following additional conditions (BC1) - (BC5):*

$$b_3\beta_{32}\alpha_2^2 = \frac{1}{3}\gamma - \gamma^2 \qquad \text{(BC1)}$$

$$\gamma(b_2\alpha_2^3 + b_3\alpha_3^3) - b_3\beta_{32}\alpha_2^3 = \gamma^2 \qquad \text{(BC2)}$$

$$2b_3\beta_{32}\alpha_2^2 = \frac{1}{3}\gamma - 2\gamma^3 \qquad \text{(BC3)}$$

$$\gamma(b_2\alpha_2^3 + b_3\alpha_3^3) - 2b_3\beta_{32}\alpha_2^3 = 3\gamma^3 \qquad \text{(BC4)}$$

$$b_3\beta_{32}\beta_2 = -2\gamma^2 + 2\gamma - \frac{1}{3} \qquad \text{(BC5)}$$

By (BC4) minus (BC2) and dividing the result by (BC1) there is (see also [59]):

$$\alpha_2 = \frac{\gamma^2 - 3\gamma^3}{\frac{1}{3}\gamma - \gamma^2} = 3\gamma.$$

However, when realizing a ROW-type scheme of the form (2.3) condition (1A1.0') must be additionally considered. Regarding (BC4) minus (BC2) and dividing the result by (1A1.0') there is:

$$\alpha_2 = \frac{\gamma^2 - 3\gamma^3}{\frac{1}{2}\gamma - \gamma^2} = 2\gamma.$$

Hence, there is a contradiction as long as $\gamma \neq 0$*. Note that there is* $\gamma \neq 0$ *due to (BC1) and (BC3) which lead to the quadratic equation [59]*

$$\frac{1}{6} - \gamma + \gamma^2 = 0.$$

5.2.2 Order Two with Four Stages

To improve properties of the sets of coefficients, further stages must be taken into account. In this context, stiffly accurate schemes that realize ROW-type method (2.3) up to order $p = 2$ with $s = 4$ internal stages are considered below. The additional stage can be used either to increase the order up to $p = 3$ regarding classical methods included (notably the ROW method with exact Jacobian for solving DAEs [64], the W method for ODEs [71] and the ROW-type scheme by Rentrop et al. [62]) or to increase the B_{PR}-consistency order up to $\tilde{p} = 3$.

Following Definition 5.1, stiffly accurate schemes with $s = 4$ internal stages satisfy

$$\alpha_4 = 1 \qquad \text{together with} \qquad b_i = \beta_{4i} \qquad \text{for} \qquad i = 1, ..., s.$$

Recall that by Theorem 5.4 stiffly accurate methods satisfy condition (1A1.0) automatically. Hence, the order conditions (1D1.0) - (2D2.0) which are required for implementing the ROW-type method (2.3) up to order $p = 2$ can be expressed by:

$$b_1 + b_2 + b_3 = 1 - \gamma \qquad \text{(1D1.0''')}$$

$$b_2\alpha_2 + b_3\alpha_3 = \frac{1}{2} - \gamma \qquad \text{(2D1.0''')}$$

$$b_2\beta_2 + b_3\beta_3 = \frac{1}{2} - 2\gamma + \gamma^2 \qquad \text{(2D1.1''')}$$

$$b_3\alpha_{32}\alpha_2 + \gamma(\alpha_{42}\alpha_2 + \alpha_{43}\alpha_3) - \alpha_{43}\beta_{32}\alpha_2 = \frac{\gamma}{2} \qquad \text{(2D2.0''')}$$

When realizing a W method for ODEs that is of order $p = 3$ these conditions must be supplemented by (3D1.0) - (3D2.3). Regarding the stiffly accurate case, corresponding additional conditions read:

$$b_2\alpha_2^2 + b_3\alpha_3^2 = \frac{1}{3} - \gamma \tag{3D1.0'''}$$

$$b_3\alpha_{32}\alpha_2 + \gamma(\alpha_{42}\alpha_2 + \alpha_{43}\alpha_3) = \frac{1}{6} \tag{3D2.0'''}$$

$$b_3\beta_{32}\alpha_2 = \frac{1}{6} - \gamma + \gamma^2 \tag{3D2.1'''}$$

$$b_3\alpha_{32}\beta_2 + \gamma(\alpha_{42}\beta_2 + \alpha_{43}\beta_3) = \frac{1}{6} - \frac{\gamma}{2} \tag{3D2.2'''}$$

$$b_3\beta_{32}\beta_2 = \frac{1}{6} - \frac{3}{2}\gamma + 3\gamma^2 - \gamma^3 \tag{3D2.3'''}$$

Note that the formulation of (3D2.1"') results from taking into account (2D1.0"') and the formulation of (3D2.3"') results from taking into account (2D1.1"'). Conditions (3D1.0"') and (3D2.0"') are also required for realizing the scheme by Rentrop et al. [62] up to order $p = 3$. For implementing the method by Rentrop et al. [62] up to order $p = 3$ completely, these conditions must be supplemented by:

$$\frac{1}{\gamma}b_3\alpha_{32}\alpha_2^2 + \alpha_{42}\alpha_2^2 + \alpha_{43}\alpha_3^2 - \frac{1}{\gamma}\alpha_{43}\beta_{32}\alpha_2^2 = \frac{1}{3} \tag{3D5.0'''}$$

$$-\frac{1}{\gamma^2}b_3\beta_{32}\alpha_2^2 + \gamma(\omega_{42}\alpha_2^2 + \omega_{43}\alpha_3^2) = 1 - \frac{1}{3\gamma} \tag{2A1.0'''}$$

Here, the formulation of (2A1.0"') is a consequence of including (3D1.0"'). The given conditions automatically satisfy the conditions required for realizing the standard ROW method introduced in [64] which solves DAEs by means of exact Jacobian matrices up to order $p = 3$. However, when not realizing a W method for ODEs [71] or the ROW-type method by Rentrop et al. [62], these order conditions can be reduced significantly. In fact, to combine ROW-type method (2.3) up to order $p = 2$ with the ROW method for DAEs up to order $p = 3$ which considers the application of exact Jacobian entries only, just the order conditions (3D1.0"'), (3D2.3"') and (2A1.0"') must be additionally taken into account.

Regarding the stiffly accurate case, condition (5.3) should be automatically satisfied for all $k \geq 2$ [57, 59]. Hence, order conditions to ensure that a stiffly accurate ROW-type scheme with $s = 4$ is B_{PR}-consistent of order $\tilde{p} = 2$ read (see Lemma 6.7 in [58] and Lemma 15 in [60]):

$$\beta_2 = 0 \tag{BPR2.1}$$

$$b_3\beta_{32}\alpha_2^2 = 2\gamma^3 - 2\gamma^2 + \frac{1}{3}\gamma \tag{BPR2.2}$$

While additional conditions to achieve B_{PR}-consistency of order $\tilde{p} = 3$ read (see

Lemma 6.4 in [58] and Lemma 12 in [60]):

$$
\begin{aligned}
\alpha_2 &= 3\gamma && \text{(BPR3.1)}\\
\beta_{32}\alpha_2^2 + \gamma\alpha_3^2 &= \frac{1}{3}\alpha_3^3 && \text{(BPR3.2)}\\
b_2\alpha_2^2 + b_3\alpha_3^2 &= \frac{1}{3} - \gamma && \text{(BPR3.3)}
\end{aligned}
$$

Note that (BPR3.3) equals condition (3D1.0”’) that is generally required for realizing a ROW-type method of order $p = 3$.

Remark 5.3 *In [59] Rang considers a third condition for realizing stiffly accurate ROW-type methods with $s = 4$ stages that are B_{PR}-consistent of order $\tilde{p} = 2$. However, it is not explicitly described in [58] and [60] regarding the same assumptions. This equation reads [59]:*

$$
b_3\beta_{32}\alpha_2^2 = \frac{2}{3}\gamma^4 - \gamma^2 + \frac{2}{9}\gamma \qquad \textit{(BPR2.3)}
$$

Note that this condition together with (BPR2.2) yields the same cubic equation as (3D2.3”’) together with (BPR2.1) [59]. Hence, it becomes relevant in cases of ROW-type methods which violate (3D2.3”’) only and can be used to fix the value of γ (e.g. regarding the ROW-type scheme introduced by Rentrop et al. [62]).

Lemma 5.4 *A stiffly accurate ROW-type method of order $p = 3$ with $s = 4$ stages that satisfies condition (3D1.0”’) together with (BPR2.2) also satisfies (2A1.0”’).*

Proof *Inserting (BPR2.2) into (2A1.0”’) yields:*

$$
\omega_{42}\alpha_2^2 + \omega_{43}\alpha_3^2 = 2 - \frac{1}{\gamma}. \tag{5.14}
$$

Considering $(\omega_{ij})_{i,j=1}^{s} = \mathcal{B}^{-1}$ with $\mathcal{B} = \beta_{i,j=1}^{1}$, it is known that

$$
\omega_{42} = \frac{\beta_{43}\beta_{32} - \beta_{42}\gamma}{\gamma^3} \qquad \textit{and} \qquad \omega_{43} = -\frac{\beta_{43}}{\gamma^2}.
$$

Including this information in (5.14) regarding the stiffly accurate case there is:

$$
-\frac{1}{\gamma^2}b_2\alpha_2^2 - \frac{1}{\gamma^2}b_3\alpha_3^2 + \frac{1}{\gamma^3}b_3\beta_{32}\alpha_2^2 = 2 - \frac{1}{\gamma}. \tag{5.15}
$$

Using (3D1.0”’) it follows:

$$
b_3\beta_{32}\alpha_2^2 = 2\gamma^3 - 2\gamma^2 + \frac{1}{3}\gamma \tag{5.16}
$$

which is equal to (BPR2.2). So, (2A1.0”’) is automatically fulfilled when (3D1.0”’) and (BPR2.2) are satisfied.

Remark 5.4 *Lemma 5.4 is an analogy to Lemma 4.2 given in [56]. Here, the fact that ROW-type methods of order $p = 3$ with $s = 4$ stages automatically satisfy (2A1.0) when considering the additional conditions by Lubich and Ostermann [42] was shown. In this context, note that Lemma 5.4 should also apply to ROW-type methods that are not stiffly accurate.*

Lemma 5.5 *A stiffly accurate ROW-type method of order $p = 3$ with $s = 4$ stages that satisfies conditions (2D1.1’”) and (3D2.0’”) - (3D2.3’”) together with (BPR2.1) also satisfies (2D2.0’”).*

Proof *Inserting (3D2.0’”) into (2D2.0’”) yields:*

$$\alpha_{43}\beta_{32}\alpha_2 = \frac{1}{6} - \frac{\gamma}{2}. \tag{5.17}$$

From (3D2.2’”) and (BPR2.1) there is

$$\gamma\alpha_{43}\beta_3 = \frac{1}{6} - \frac{\gamma}{2}. \tag{5.18}$$

Therefore, (5.17) and (5.18) can be used to define:

$$\alpha_{43}\beta_{32}\alpha_2 = \gamma\alpha_{43}\beta_3. \tag{5.19}$$

Now, dividing (5.19) by α_{43} and multiplying the result by b_3 gives:

$$b_3\beta_{32}\alpha_2 = \gamma b_3\beta_3. \tag{5.20}$$

Considering (2D1.1’”) together with (BPR2.1) there is

$$b_3\beta_3 = \frac{1}{2} - 2\gamma + \gamma^2. \tag{5.21}$$

Hence, by inserting (3D2.1’”) and (5.21) into (5.20) it follows after reformulation:

$$\frac{1}{6} - \frac{3}{2}\gamma + 3\gamma^2 - \gamma^3 = 0. \tag{5.22}$$

It is the same cubic equation that results from (3D2.3’”) together with (BPR2.1). So, appropriate values of γ that satisfy (5.22) together with the order conditions considered ensure that (2D2.0’”) is automatically satisfied.

Remark 5.5 *Note that (2D2.0’”) is also automatically satisfied for stiffly accurate ROW-type methods of order $p = 3$ with $s = 4$ stages when regarding given order conditions together with the less stringent conditions by Lubich and Ostermann [42] instead of those by Rang [59, 60].*

Theorem 5.5 *Any stiffly accurate W method for ODEs that is of order $p = 3$ with $s = 4$ internal stages corresponds to an appropriate set of coefficients for realizing ROW-type method (2.3) with $B_z = (g_z)_0$ up to order $p = 2$ as long as it is B_{PR}-consistent of order $\tilde{p} = 2$.*

Proof *By the order conditions listed in Table 4.2 that are required to realize ROW-type method (2.3), taking into account Lemma 5.4 and Lemma 5.5.*

Remark 5.6

- *W methods of order $p = 3$ with $s = 4$ internal stages that are B_{PR}-consistent of order $\tilde{p} = 2$ are existing in literature already. Such schemes are ROS34PW2 [56] and ROS34PRW [57], for example. As both these schemes are stiffly accurate they satisfy the requirements of Theorem 5.5.*
- *W methods of order $p = 3$ with $s = 4$ internal stages cannot be B_{PR}-consistent of order $\tilde{p} = 3$. This is shown in [59]: When dividing condition (BPR2.2) by (3D2.1''') there is $\alpha_2 = 2\gamma$. This, however, violates condition (BPR3.1) which is required to reach B_{PR}-consistency of order $\tilde{p} = 3$.*

Note that for a ROW-type scheme with $s = 4$ internal stages the stability function generally becomes:

$$\begin{aligned} R(\tilde{z}) = 1 &+ \frac{(b_1 + b_2 + b_3 + b_4)\tilde{z}}{(1-\gamma\tilde{z})} + \frac{(b_2\beta_2 + b_3\beta_3 + b_4\beta_4)\tilde{z}^2}{(1-\gamma\tilde{z})^2} \\ &+ \frac{(b_3\beta_{32}\beta_2 + b_4(\beta_{42}\beta_2 + \beta_{43}\beta_3))\tilde{z}^3}{(1-\gamma\tilde{z})^3} + \frac{(b_4\beta_{43}\beta_{32}\beta_2)\tilde{z}^4}{(1-\gamma\tilde{z})^4} \end{aligned}$$

Hence, when regarding the stiffly accurate case and including the order conditions (1D1.0''') and (2D1.1''') it follows:

$$R(\tilde{z}) = \frac{1 + (1-4\gamma)\tilde{z} + (\frac{1}{2} - 4\gamma + 6\gamma^2)\tilde{z}^2 + (b_3\beta_{32}\beta_2 - \frac{1}{2}\gamma + 3\gamma^2 - 3\gamma^3)\tilde{z}^3}{(1-\gamma\tilde{z})^4}. \quad (5.23)$$

GROW34PRw

As mentioned in Remark 5.6 schemes that satisfy W method condition (3D2.1''') cannot be B_{PR}-consistent of order $\tilde{p} = 3$. For that reason, Rang introduced ROS34PRw in [59] which avoids the given conflict by neglecting (3D2.1'''). ROS34PRw is considered to be an alternative version of ROS34PRW which achieves B_{PR}-consistency of order $\tilde{p} = 3$. In more detail, ROS34PRw is a stiffly accurate scheme that satisfies the conditions (1D1.0''') - (2D1.1''') together with (3D1.0'''), (3D2.0'''), (3D2.2''') as well as (1A1.0''') (automatically satisfied by Lemma 5.5) and (2A2.0'''). It also satisfies condition (3D2.3''') which corresponds to a condition by Roche [64]. However, the original condition (3D2.3) is violated. The resulting scheme is B_{PR}-consistent of order $\tilde{p} = 3$, but no W methods of order $p = 3$. Only when solving DAEs with exact Jacobian it preserves order $p = 3$ and reduces to order $p = 2$ when being applied as a W method for ODEs or realizing the scheme by Rentrop et al. [62]. However, it violates (2D2.0'''), too. Hence, it reduces even to order $p = 1$ when being applied as ROW-type method (2.3). For that reason, the objective is to extend this method appropriately below such that it preserves order $p = 2$ at least.

However, realizing a corresponding scheme is not possible when taking into account the order conditions satisfied by the original method ROS34PRw. By the remark given above, we only know that there can be no full W method and that (3D2.1"') is problematic which is skipped by ROS34PRw. Below, Lemma 5.6 shows that there is no possibility of realizing a version of ROW-type scheme (2.3) up to order $p = 2$ with $s = 4$ stages that is B_{PR}-consistent of order $\tilde{p} = 3$ and that additionally takes into account (3D2.0"'), (3D2.2"') and (3D2.3"'). Note that (3D2.3"') is required to realize a standard ROW method of order $p = 3$ to solve DAEs by means of exact Jacobian matrices.

Lemma 5.6 *A stiffly accurate ROW-type method (2.3) of order $p = 2$ that consists of $s = 4$ internal stages cannot be B_{PR}-consistent of order $\tilde{p} = 3$ when taking into account (3D2.0"'), (3D2.2"') and (3D2.3"') additionally.*

Proof *To reach order $p = 2$ with $s = 4$ internal stages, ROW-type method (2.3) requires to satisfy order conditions (1D1.0"') - (2D2.0"'). In addition, to reach B_{PR}-consistency of order $\tilde{p} = 3$, conditions (BPR2.1) - (BPR3.3) must be considered. Due to (BPR2.1), valid values of γ are fixed by the solutions of the cubic equation*

$$0 = \frac{1}{6} - \frac{2}{3}\gamma + 3\gamma^2 - \gamma^3 \tag{5.24}$$

that is given when taking into account (3D2.3"'). Now, (BPR2.2) in combination with (BPR3.1) yields

$$b_3\beta_{32} = \frac{2}{9}\gamma - \frac{2}{9} + \frac{1}{27\gamma}. \tag{5.25}$$

Inserting (BPR2.1) and (5.25) into (2D1.1"') leads to

$$b_3\beta_{31} = \frac{13}{18} - \frac{20}{9}\gamma + \gamma^2 - \frac{1}{27\gamma}. \tag{5.26}$$

Next, (2D2.0"') is reformulated by considering (3D2.0"') and (BPR3.1). It follows:

$$\alpha_{43}\beta_{32} = \frac{1}{18\gamma} - \frac{1}{6}. \tag{5.27}$$

Using this result in combination with (BPR2.1) order condition (3D2.2"') leads to

$$\alpha_{43}\beta_{31} = \frac{1}{9\gamma} - \frac{1}{3}. \tag{5.28}$$

Finally, divide (5.25) by (5.26) and (5.27) by (5.28) to get

$$\frac{3}{486} - \frac{27}{324}\gamma + \frac{111}{324}\gamma^2 - \frac{27}{54}\gamma^3 + \frac{1}{6}\gamma^4 = 0. \tag{5.29}$$

However, the roots of (5.29) do not conform to the roots of (5.24). Hence, it is not possible to realize a scheme that satisfies all the order conditions considered.

Due to the restrictions given there can be no set of coefficients based on ROS34PRw that satisfies the same order conditions while allowing to preserve order $p = 2$ when solving DAEs by means of non-exact Jacobian matrices. Instead, a scheme must be considered that neglects the order conditions that exclusively correspond to W methods for ODEs regarding order $p = 3$. As an alternative, satisfying the order conditions to include a ROW-type method according to Rentrop et al. [62] up to order $p = 3$ while ensuring B_{PR}-consistency of order $\tilde{p} = 3$ could be promising. A corresponding scheme requires to to consider (3D1.0), (3D2.0) and (3D5.0) only. Besides (3D2.3"') should be included to allow for ROW methods to solve DAEs with exact Jacobian up to order $p = 3$ as well. However, as following Lemma 5.6 shows, including reformulated conditions (3D2.0"') and (3D5.0"') leads to further conflicts.

Lemma 5.7 *A stiffly accurate ROW-type method (2.3) of order $p = 2$ that consists of $s = 4$ internal stages cannot be B_{PR}-consistent of order $\tilde{p} = 3$ when taking into account (3D2.0"') and (3D5.0"') additionally.*

Proof *First, note that the value of γ is fixed by (BPR2.2) in combination with (BPR2.3). They lead to the cubic equation:*

$$0 = \gamma^3 - 3\gamma^2 + \frac{3}{2}\gamma - \frac{1}{6}. \tag{5.30}$$

Among the root of this equation, only $\gamma \approx 0.43$ is within the range of appropriate values according to Table 6.4 in Section IV of [24].

Next, considering (2D2.0"') times α_2 minus (3D5.0"') times γ there is

$$\alpha_{43} = \frac{3\alpha_2 - 2}{6(\alpha_2\alpha_3 - \alpha_3^2)}. \tag{5.31}$$

Also, when considering (2D2.0"') minus (3D2.0"') there is

$$\alpha_{43} = -\frac{\frac{\gamma}{2} - \frac{1}{6}}{\beta_{32}\alpha_2}. \tag{5.32}$$

Now, setting (5.31) equal to (5.32) it follows

$$(3\alpha_2^2 - 2\alpha_2)\beta_{32} = (1 - 3\gamma)(\alpha_2\alpha_3 - \alpha_3^2). \tag{5.33}$$

Taking into account (BPR3.2) to express β_{32} in dependence of α_2 and α_3 as well as (BPR3.1) to express α_2 in dependence of γ, this equation can be rewritten as:

$$\left(1 - \frac{2}{9\gamma}\right)\alpha_3^3 - \left(6\gamma - \frac{5}{3}\right)\alpha_3^2 - \left(3\gamma - 9\gamma^2\right)\alpha_3 = 0 \tag{5.34}$$

By considering the given value of $\gamma \approx 0.43$, the solutions of this equations read (determined with MUPAD):

$$\alpha_3 = \left\{0, 3\gamma, -\frac{9\gamma - 27\gamma^2}{9\gamma - 2}\right\}.$$

Alternatively, regard (2D1.0”’) times α_2 *minus (3D1.0”’) to find*

$$b_3 = \frac{-\frac{1}{3} + \frac{5}{2}\gamma - 3\gamma^2}{3\gamma\alpha_3 - \alpha_3^2}. \tag{5.35}$$

Also, taking into account β_{32} *as defined by (BPR3.2), it follows from (BPR2.2):*

$$b_3 = \frac{2\gamma^3 - 2\gamma^2 + \frac{1}{3}\gamma}{\frac{1}{3}\alpha_3^3 - \gamma\alpha_3^2}. \tag{5.36}$$

Now, setting (5.35) equal to (5.36) appropriate values of α_3 *can be determined by solving the resulting cubic equation:*

$$0 = \left(\frac{1}{9} - \frac{5}{6}\gamma + \gamma^2\right)\alpha_3^3 + \left(-5\gamma^3 + \frac{9}{2}\gamma^2 - \frac{2}{3}\gamma\right)\alpha_3^2 + \left(6\gamma^4 - 6\gamma^3 + \gamma^2\right)\alpha_3. \tag{5.37}$$

For the given value of γ *roots of this equation read (determined with MUPAD):*

$$\alpha_3 = \left\{0, 3\gamma, \frac{36\gamma^3 - 36\gamma^2 + 6\gamma}{18\gamma^2 - 15\gamma + 2}\right\}.$$

By comparison with the solutions of equation (5.34) only $\alpha_3 = 0$ *and* $\alpha_3 = 3\gamma$ *can be adequate solutions. However, these choices set the denominator of (5.35) and (5.36) to zero when trying to compute an appropriate value of* b_3. *As a consequence, there is no possibility to realize a ROW-type method that is* B_{PR}*-consistent of order* $\tilde{p} = 3$ *and that combines the scheme formulation (2.3) up to order* $p = 2$ *with schemes that require for conditions (3D2.0”’) and (3D5.0”’).*

Because of Lemma 5.7 there can be no scheme that combines a ROW-type scheme (2.3) up to order $p = 2$ with the ROW-type method for DAEs introduced by Rentrop et al. [62] up to order $p = 3$ while realizing B_{PR}-consistency of order $\tilde{p} = 3$. Also, due to Remark 5.6 and Lemma 5.6 there can be no combination with a W method of order $p = 3$. For that reason, an extension of ROS34PRw is considered that satisfies (1D1.0”’) - (2D2.0”’) together with (3D1.0”’) and (3D2.3”’) as well as (BPR2.1) - (BPR3.2). Note that condition (2A1.0”’) is automatically satisfied due to Lemma 5.4. As a consequence, the resulting method neglects order conditions that are fulfilled by the original method ROS34PRw. However, it preserves the same orders (i.e. B_{PR}-consistency of order $\tilde{p} = 3$, order $p = 3$ w.r.t. the ROW methods for DAEs as introduced by Roche [64], order $p = 2$ w.r.t. the ROW-type methods for DAEs as introduced by Rentrop et al. [62] and W methods for ODEs as introduced by Steihaug and Wolfbrand [71]) and even increases the order regarding an application as ROW-type method (2.3) from $p = 1$ to $p = 2$.

There are 10 equations and 12 unknowns. In order to determine the enhanced set of coefficients steps described in the proof of Lemma 5.7 can be used. First, conditions (3D2.3”’) and (BPR2.1) are used to determine a value of γ. Note that the cubic equation to solve is equal to (5.30) that follows from (BPR2.2) and (BPR2.3). As mentioned before $\gamma \approx 0.43$ is the only root which lies within the range of appropriate values according to Table 6.4 given in Section IV of [24]. It is the same value as for the original method ROS34PRw [59]. Knowing γ, the value of $\alpha_2 = 3\gamma$ is fixed by (BPR3.1).

Next, cubic equation (5.37) is determined using (5.35) and (5.36). Corresponding solutions are described in the proof of Lemma 5.7. As mentioned before, $\alpha_3 = 0$ and $\alpha_3 = 3\gamma$ are no appropriate choices for realizing the set of coefficients. This is because they set the denominators of (5.35) and (5.36) to zero. For that reason, there must be

$$\alpha_3 = \frac{36\gamma^3 - 36\gamma^2 + 6\gamma}{18\gamma^2 - 15\gamma + 2} \approx 1.1115$$

with respect to the given value of γ. It is the same value as for the original method ROS34PRw [60].

The given value of α_3 allows to determine b_3 via (5.35) or (5.36). Afterwards β_{32} follows from (BPR3.2) and b_2 from (2D1.0"') or (3D1.0"'). As $b_4 = \gamma$ due to the stiffly accurate property, b_1 is determined using (1D1.0"'). Also, because $\beta_2 = 0$ by (BPR2.1), β_{31} can now be computed via (2D1.1"'). Next, one of the values α_{32}, α_{42} or α_{43} can be evaluated by (2D2.0"'). Because these coefficients do not occur within the other order conditions considered directly, the value to compute by (2D2.0"') can be chosen arbitrarily. The remaining coefficients correspond to free parameters. Finally, there is $\alpha_{31} = \alpha_3 - \alpha_{32}$ and $\alpha_{41} = 1 - \alpha_{42} - \alpha_{43}$. There is $\gamma_{41} = b_1 - \alpha_{41}$, $\gamma_{42} = b_2 - \alpha_{42}$ and $\gamma_{43} = b_3 - \alpha_{43}$ due to the stiffly accurate properties given, too. In order to realize an appropriate scheme $\alpha_{42} = 0.5$ and $\alpha_{43} \approx 0.55$ are considered below. They correspond to the values of the original scheme ROW34PRw [60].

An embedded scheme must be of order $p = 1$ regarding the application in the form of ROW-type method (2.3). As well, order $p = 1$ must be given when applying the scheme in the form of the ROW-type method introduced by Rentrop et al. [62] or in the form of a W method for ODEs. Nevertheless, regarding the application as ROW method with exact Jacobian for solving DAEs, order $p = 2$ must be ensured. Assuming $s = 4$ internal stages corresponding order conditions read

$$\hat{b}_1 + \hat{b}_2 + \hat{b}_3 + \hat{b}_4 = 1 \qquad \text{(e1D1.0"')}$$

$$b_3\beta_3 + b_4(1-\gamma) = \frac{1}{2} - \gamma \qquad \text{(e2D1.1"')}$$

after including $\beta_2 = 0$ and $\beta_4 = 1 - \gamma$. There are two degrees of freedom. After choosing $\hat{b}_1 \approx 0.59$ and $\hat{b}_3 \approx 0.55$ as for the original scheme ROS34PRw there is $\hat{b}_2 \approx -0.46$ and $\hat{b}_4 \approx 0.32$. Hence, the embedded scheme is the same as for the original method ROS34PRw. Note that it yields $|\tilde{R}(\infty)| = 0.25$.

The resulting method is called GROW34PRw below. The corresponding set of coefficients is summarized in Table 5.4. Its stability region is shown in Figure 5.4.

Table 5.4: Set of Coefficients for GROW34PRw

$\gamma = 4.3586652150845900\text{E}{-01}$					
α_{21}	=	$1.3075995645253771\text{E}{+00}$	γ_{21}	=	$-1.3075995645253771\text{E}{+00}$
α_{31}	=	$1.4417785675351402\text{E}{+00}$	γ_{31}	=	$-1.6070872240995751\text{E}{+00}$
α_{32}	=	$-3.3028050590993452\text{E}{-01}$	γ_{32}	=	$2.8304946117723884\text{E}{-01}$
α_{41}	=	$-5.3402207849443051\text{E}{-02}$	γ_{41}	=	$4.4024152788200843\text{E}{-01}$
α_{42}	=	$5.0000000000000000\text{E}{-01}$	γ_{42}	=	$-1.1778562785454629\text{E}{+00}$
α_{43}	=	$5.5340220784944305\text{E}{-01}$	γ_{43}	=	$3.0174822915499544\text{E}{-01}$
b_1	=	$3.8683932003256538\text{E}{-01}$	$\hat{b}_1$	=	$5.8643117861132599\text{E}{-01}$
b_2	=	$-6.7785627854546282\text{E}{-01}$	$\hat{b}_2$	=	$-4.6123460043657361\text{E}{-01}$
b_3	=	$8.5515043700443849\text{E}{-01}$	$\hat{b}_3$	=	$5.5283538820777700\text{E}{-01}$
b_4	=	$4.3586652150845900\text{E}{-01}$	$\hat{b}_4$	=	$3.2196803361747062\text{E}{-01}$

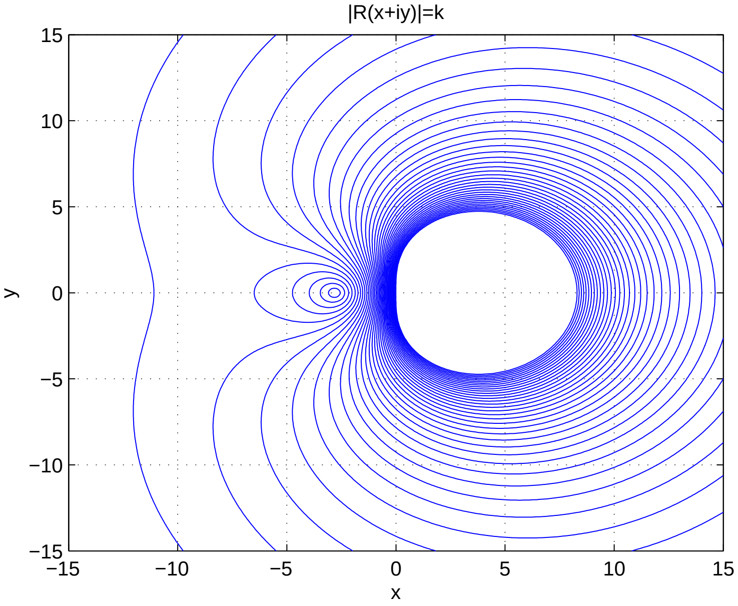


Figure 5.4: Stability region of GROW34PRw regarding `k=0:0.025:1`.

GROW3PRL2

Another scheme introduced by Rang in [59] that is B_{PR}-consistent of order $\tilde{p} = 3$ is ROS3PRL2. ROS3PRL2 is an extension of the method ROS3PL originally introduced by Lang and Teleaga in [36]. It is a stiffly accurate ROW method of order $p = 3$ that violates the order conditions (2D1.0''') - (2D2.0'''). Hence, when solving DAEs by means of non-exact Jacobian matrices it reduces its order to $p = 1$. This is the case when applying this set of coefficients in the form of the ROW-type method introduced by Rentrop et al. [62] as well as the ROW-type method (2.3). Below, an extension of ROS3PRL2 is considered such that it preserves its original properties but reaches order $p = 2$ when applying a non-exact Jacobian. In this context, the properties of the resulting scheme will be exactly the same as for GROW34PRw.

For realizing an appropriate set of coefficients, the steps for determining the coefficients of GROW34PRw can be applied again. As a consequence, the values of γ, α_2, α_3, β_2, β_{31}, β_{32} and b_1 - b_4 are exactly the same as for GROW34PRw. Differences are given when choosing α_{42} and α_{43} in order to compute α_{32} via (2D2.0'''). Here, $\alpha_{42} = 0.5$ and $\alpha_{43} = 0$ as for the original method ROS3PRL2 are considered.

The embedded scheme is computed by using (e1D1.0''') and (e2D1.1''') together with

$$-5\gamma^4 + 4\gamma^3(1 + \hat{b}_4) - 4\gamma^2(\hat{b}_3\beta_3 + \hat{b}_4) + 4\gamma\hat{b}_4 b_3\beta_3 = 0.$$

The additional condition was suggested by Rang in [59] when realizing the coefficients of ROS3PRL2. Together with $\hat{b}_1 = 0.5$ assumed to be the only degree of freedom it is used to ensure that the embedded method is strongly A-stable with $|\tilde{R}(\infty)| = 0.25$. The remaining values become $\hat{b}_2 \approx -0.37$, $\hat{b}_3 \approx 0.55$ and $\hat{b}_4 \approx 0.32$.

The resulting method is called GROW3PRL2 below. The corresponding set of coefficients is summarized in Table 5.5. Its stability region is shown in Figure 5.5.

5.3 Numerically Determined Sets

In order to realize schemes that are characterized by enhanced properties further internal stages must be taken into account. However, increasing the number of internal stages complicates the formulation of order conditions and, thus, makes it more challenging to find appropriate sets of coefficients. The process of realizing corresponding methods can be sped up by solving the order conditions given numerically. However, in contrast to schemes determined analytically, results might be less reliable. In the following, some sets of coefficients that were computed numerically using MATLAB's `fsolve`-function for solving the given systems of non-linear equations are introduced. In this context, all methods were evaluated based on the Levenberg-Marquardt algorithm. Detailed information with respect to this algorithm can be found in [38], [43] and [46].

5.3.1 Order Two with Five Stages

According to Lemma 5.7 there is no possibility to realize a stiffly accurate version of ROW-type method (2.3) up to order $p = 2$ with $s = 4$ internal stages which covers

Table 5.5: Set of Coefficients for GROW3PRL2

$\gamma = 4.3586652150845900\text{E}{-01}$	
$\alpha_{21} = 1.30759956452537710\text{E}{+00}$	$\gamma_{21} = -1.30759956452537710\text{E}{+00}$
$\alpha_{31} = 1.17144844213035750\text{E}{+00}$	$\gamma_{31} = -1.33675709869479230\text{E}{+00}$
$\alpha_{32} = -5.9950380505151696\text{E}{-02}$	$\gamma_{32} = 1.2719335772456014\text{E}{-02}$
$\alpha_{41} = 5.0000000000000000\text{E}{-01}$	$\gamma_{41} = -1.1316067996743462\text{E}{-01}$
$\alpha_{42} = 5.0000000000000000\text{E}{-01}$	$\gamma_{42} = -1.17785627854546290\text{E}{+00}$
$\alpha_{43} = 0.0000000000000000\text{E}{+00}$	$\gamma_{43} = 8.5515043700443849\text{E}{-01}$
$b_1 = 3.8683932003256538\text{E}{-01}$	$\hat{b}_1 = 5.0000000000000000\text{E}{-01}$
$b_2 = -6.7785627854546282\text{E}{-01}$	$\hat{b}_2 = -3.7480342182506449\text{E}{-01}$
$b_3 = 8.5515043700443849\text{E}{-01}$	$\hat{b}_3 = 5.5283538820763400\text{E}{-01}$
$b_4 = 4.3586652150845900\text{E}{-01}$	$\hat{b}_4 = 3.2196803361743076\text{E}{-01}$

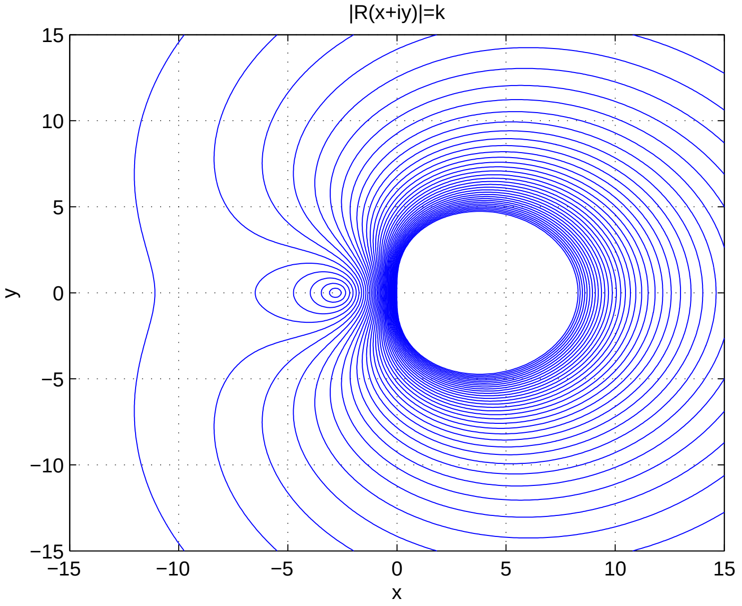


Figure 5.5: Stability region of GROW3PRL2 regarding `k=0:0.025:1`.

Table 5.6: Set of Coefficients for GROW35n

$\gamma = 4.35866521508452900\text{E}{-01}$					
α_{21}	$=$	$2.30289600239868860\text{E}{-01}$	γ_{21}	$=$	$-2.30666492716850410\text{E}{-01}$
α_{31}	$=$	$8.56014605401288380\text{E}{-01}$	γ_{31}	$=$	$-8.55583404600044450\text{E}{-01}$
α_{32}	$=$	$4.87782891478346910\text{E}{-01}$	γ_{32}	$=$	$-4.87782891477525740\text{E}{-01}$
α_{41}	$=$	$1.58586816267367500\text{E}{+00}$	γ_{41}	$=$	$1.23807403488298860\text{E}{+00}$
α_{42}	$=$	$6.84479185781682320\text{E}{-01}$	γ_{42}	$=$	$4.84345364691425080\text{E}{-01}$
α_{43}	$=$	$0.14541556841273919\text{E}{-01}$	γ_{43}	$=$	$1.00707337986753090\text{E}{+00}$
α_{51}	$=$	$-1.32793992068333780\text{E}{-01}$	γ_{51}	$=$	$2.65673469875456590\text{E}{-01}$
α_{52}	$=$	$9.92217321266755550\text{E}{-01}$	γ_{52}	$=$	$-5.59978724875248020\text{E}{-01}$
α_{53}	$=$	$1.71798490384802630\text{E}{-01}$	γ_{53}	$=$	$-1.36566117656897000\text{E}{-01}$
α_{54}	$=$	$-0.31221819583224428\text{E}{-01}$	γ_{54}	$=$	$-0.04995148851764650\text{E}{-01}$
b_1	$=$	$1.32879477807122780\text{E}{-01}$	$\hat{b}_1$	$=$	$2.27038807437743790\text{E}{-01}$
b_2	$=$	$4.32238596391507580\text{E}{-01}$	$\hat{b}_2$	$=$	$3.18169879735279380\text{E}{-01}$
b_3	$=$	$0.35232372727905630\text{E}{-01}$	$\hat{b}_3$	$=$	$0.54213020735109731\text{E}{-01}$
b_4	$=$	$-0.36216968434989033\text{E}{-01}$	$\hat{b}_4$	$=$	$-0.36346199176705209\text{E}{-01}$
b_5	$=$	$4.35866521508452900\text{E}{-01}$	$\hat{b}_5$	$=$	$4.36924491266059090\text{E}{-01}$

the ROW-type method introduced by Roche [64] as well as the ROW-type method introduced by Rentrop et al. [62] up to order $p = 3$ as long as the order conditions for realizing B_{PR}-consistency of order $\tilde{p} = 3$ are taken into account. However, when considering an increased number of $s = 5$ internal stages, a set of coefficients which satisfies all required order conditions (1D1.0’”) - (2D2.0’”) together with (3D1.0’”), (3D2.0’”), (3D5.0’”) and (BPR2.1) - (BPR3.2) can be found.

Regarding the embedded method, a set of coefficients with $s = 5$ stages was found that satisfies the order conditions (1D1.0’”) together with (2D2.0’”), (2D2.1’”) so that order $p = 1$ is ensured regarding ROW-type method (2.3) and $p = 2$ is ensured regarding the ROW-type methods introduced by Roche [64] and Rentrop et al. [62]. Note that the embedded method considers $|\tilde{R}(\infty)| \approx 0.0023$.

The resulting method is called GROW35n below. The corresponding set of coefficients is summarized in Table 5.6. Its stability region is shown in Figure 5.6.

5.3.2 Order Three with Seven Stages

So far, sets of coefficients have been considered that realize ROW-type scheme (2.3) up to order $p = 2$. In the following, sets of coefficients are introduced which enable to attain full order $p = 3$ in combination with B_{PR}-consistency of order $\tilde{p} = 3$. This means, methods are considered that satisfy all the order conditions shown in Table 4.2. Below, three corresponding sets of coefficients are introduced. Two of them even satisfy the order conditions that are required to preserve order $p = 3$ when including non-exact Jacobian entries of the form $B_z = g_z + \mathcal{O}(h)$. Regarding the degrees of

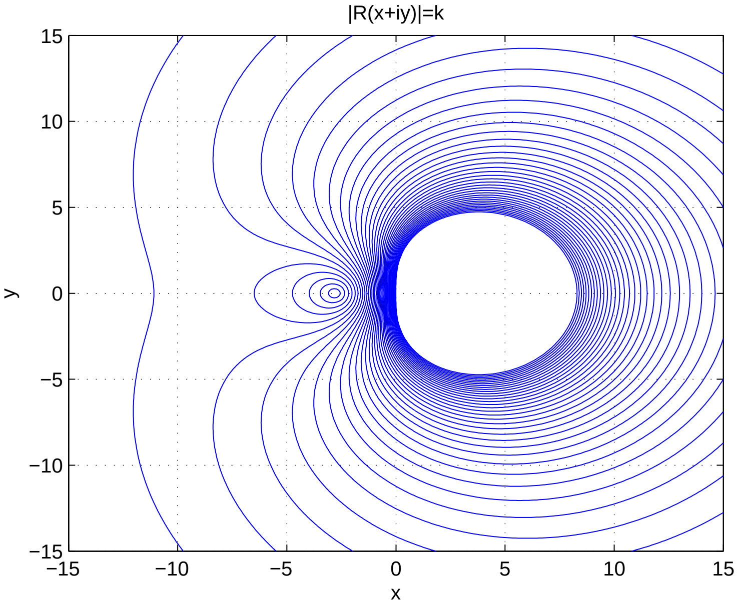


Figure 5.6: Stability region of GROW35n regarding `k=0:0.025:1`.

freedom required for a method of order $p = 3$, the following theorem can generally be applied.

Theorem 5.6 *Realizing ROW-type method (2.3) with $B_z = (g_z)_0$ up to convergence order $p = 3$ requires a minimum of $s = 5$ stages.*

Proof *To reach order $p = 3$ condition (3D9.0) is required to be unequal zero. However, this can only be achieved by taking into account $s = 5$ stages at least.*

Remark 5.7 *Although the minimum number of $s = 5$ internal stages required to realize ROW-type method (2.3) up to order $p = 3$ is defined by Theorem 5.6, a corresponding set of coefficients that satisfies the order conditions listed in Table 4.2 could not be found numerically. This was the case when neglecting all the supplementing conditions especially with respect to B_{PR}-consistency, too. Note that for $s = 5$ internal stages the given number of coefficients exactly matches the number of order conditions in Table 4.2. Hence, there are no degrees of freedom left for realizing an appropriate set of coefficients. As a consequence, there is an increased risk of conflicts within the order conditions given that might require for stage numbers $s > 5$.*

In order to satisfy all the conditions considered, at least $s = 7$ internal stages were required to find a corresponding set of coefficients numerically. Searches based on $s = 5$ as well as $s = 6$ stages failed. The three sets of coefficients determined are denoted

Table 5.7: Set of Coefficients for GROW37nr

$\gamma = 4.55341801261479610\text{E}{-01}$					
α_{21}	$=$	$6.39072649720047630\text{E}{-01}$	γ_{21}	$=$	$0.71452542081285195\text{E}{-01}$
α_{31}	$=$	$-7.38182533108201260\text{E}{-01}$	γ_{31}	$=$	$-1.18295112907884750\text{E}{+00}$
α_{32}	$=$	$6.59742926646062040\text{E}{-01}$	γ_{32}	$=$	$-6.59742926644450890\text{E}{-01}$
α_{41}	$=$	$1.94455521192068530\text{E}{+00}$	γ_{41}	$=$	$-7.21406648981638980\text{E}{-01}$
α_{42}	$=$	$5.87226347444779640\text{E}{-01}$	γ_{42}	$=$	$8.26573575289443420\text{E}{-01}$
α_{43}	$=$	$-8.85190339189261160\text{E}{-01}$	γ_{43}	$=$	$1.40807975653987570\text{E}{+00}$
α_{51}	$=$	$1.08350349665142100\text{E}{+00}$	γ_{51}	$=$	$7.46826117061979100\text{E}{-01}$
α_{52}	$=$	$-3.55543081017569600\text{E}{-01}$	γ_{52}	$=$	$1.12919896352708630\text{E}{+00}$
α_{53}	$=$	$1.34231587446636570\text{E}{-01}$	γ_{53}	$=$	$1.80121099486878910\text{E}{-01}$
α_{54}	$=$	$2.89455170754152280\text{E}{-01}$	γ_{54}	$=$	$-2.58646464447305990\text{E}{-01}$
α_{61}	$=$	$0.44229454295585967\text{E}{-01}$	γ_{61}	$=$	$-0.07088940896496590\text{E}{-01}$
α_{62}	$=$	$1.39607580262840040\text{E}{+00}$	γ_{62}	$=$	$1.02686782397905720\text{E}{+00}$
α_{63}	$=$	$0.46364255411056772\text{E}{-01}$	γ_{63}	$=$	$0.67022085664239714\text{E}{-01}$
α_{64}	$=$	$3.54708918451585590\text{E}{-01}$	γ_{64}	$=$	$0.23253241298293120\text{E}{-01}$
α_{65}	$=$	$0.70815545152659060\text{E}{-01}$	γ_{65}	$=$	$-1.07001185531280090\text{E}{+00}$
α_{71}	$=$	$3.69355722445910030\text{E}{-01}$	γ_{71}	$=$	$-1.42714051543582740\text{E}{-01}$
α_{72}	$=$	$1.05101662167042510\text{E}{+00}$	γ_{72}	$=$	$-7.11266806088810210\text{E}{-01}$
α_{73}	$=$	$-1.42817221470326190\text{E}{-01}$	γ_{73}	$=$	$2.33408391377320080\text{E}{-01}$
α_{74}	$=$	$0.24825037796669329\text{E}{-01}$	γ_{74}	$=$	$-0.09281027391284630\text{E}{-01}$
α_{75}	$=$	$-4.66051960281407110\text{E}{-01}$	γ_{75}	$=$	$3.95636829096837170\text{E}{-01}$
α_{76}	$=$	$1.63671799838729340\text{E}{-01}$	γ_{76}	$=$	$-2.21125136711960160\text{E}{-01}$
b_1	$=$	$2.26641670902327290\text{E}{-01}$	$\hat{b}_1$	$=$	$2.78348967549010280\text{E}{-01}$
b_2	$=$	$3.39749815581615160\text{E}{-01}$	$\hat{b}_2$	$=$	$9.86113252374122280\text{E}{-01}$
b_3	$=$	$0.90591169906993785\text{E}{-01}$	$\hat{b}_3$	$=$	$-0.44375878106397079\text{E}{-01}$
b_4	$=$	$0.15544010405384820\text{E}{-01}$	$\hat{b}_4$	$=$	$0.75424755826637435\text{E}{-01}$
b_5	$=$	$-0.70415131184569629\text{E}{-01}$	$\hat{b}_5$	$=$	$-4.03922498424570730\text{E}{-01}$
b_6	$=$	$-0.57453336873230933\text{E}{-01}$	$\hat{b}_6$	$=$	$1.08411400781197930\text{E}{-01}$
b_7	$=$	$4.55341801261479610\text{E}{-01}$	$\hat{b}_7$	$=$	$0.00000000000000000\text{E}{+00}$

by GROW37nr, GROW37n and GROW37n2. All these schemes are stiffly accurate and satisfy the order conditions listed in Table 4.2. The major difference between GROW37nr and GROW37n as well as GROW37n2 is that GROW37n and GROW37n2 also satisfy the order conditions required to preserve full order $p = 3$ when including non-exact Jacobian expressions of the form $B_z = g_z + \mathcal{O}(h)$. Note that corresponding conditions have been derived realizing the given thesis. But they are not presented in detail due to their extensive theory. GROW37nr does not satisfy corresponding additional conditions. The letter 'r' in GROW37nr therefore is standing for 'reduced'. Note that the embedded methods ensure order $p = 2$ with respect to all ROW-type

Table 5.8: Set of Coefficients for GROW37n

$\gamma = 4.55341801261479050\mathrm{E}{-01}$					
α_{21}	$=$	$9.10683602522037510\mathrm{E}{-01}$	γ_{21}	$=$	$-9.10683602522319500\mathrm{E}{-01}$
α_{31}	$=$	$1.76555028814813290\mathrm{E}{+00}$	γ_{31}	$=$	$-1.35581710936781200\mathrm{E}{+00}$
α_{32}	$=$	$-3.33678543348817860\mathrm{E}{-01}$	γ_{32}	$=$	$7.43411722130622450\mathrm{E}{-01}$
α_{41}	$=$	$9.52006951763304850\mathrm{E}{-01}$	γ_{41}	$=$	$-1.29201964868383160\mathrm{E}{+00}$
α_{42}	$=$	$-1.13782286825624170\mathrm{E}{+00}$	γ_{42}	$=$	$6.59048457019262070\mathrm{E}{-01}$
α_{43}	$=$	$1.33926045914029500\mathrm{E}{+00}$	γ_{43}	$=$	$-9.36977469416560900\mathrm{E}{-01}$
α_{51}	$=$	$-1.29095155458942170\mathrm{E}{+00}$	γ_{51}	$=$	$-6.64638307460343360\mathrm{E}{-01}$
α_{52}	$=$	$1.51585500845095590\mathrm{E}{+00}$	γ_{52}	$=$	$-3.05817273777330990\mathrm{E}{+00}$
α_{53}	$=$	$-0.84245777675290553\mathrm{E}{-01}$	γ_{53}	$=$	$1.45799003697520900\mathrm{E}{+00}$
α_{54}	$=$	$-3.89744350884809410\mathrm{E}{-01}$	γ_{54}	$=$	$0.27333808235919389\mathrm{E}{-01}$
α_{61}	$=$	$1.07435890408946140\mathrm{E}{+00}$	γ_{61}	$=$	$-7.16626872923143090\mathrm{E}{-01}$
α_{62}	$=$	$1.86415531666235060\mathrm{E}{+00}$	γ_{62}	$=$	$-1.03836938896198050\mathrm{E}{+00}$
α_{63}	$=$	$-1.67949712212497880\mathrm{E}{+00}$	γ_{63}	$=$	$1.32047956420658630\mathrm{E}{+00}$
α_{64}	$=$	$-2.30171657580979240\mathrm{E}{-01}$	γ_{64}	$=$	$0.65791887224906476\mathrm{E}{-01}$
α_{65}	$=$	$-0.28845441045860110\mathrm{E}{-01}$	γ_{65}	$=$	$0.43705564362958652\mathrm{E}{-01}$
α_{71}	$=$	$2.69157431244354260\mathrm{E}{-01}$	γ_{71}	$=$	$-1.06590335903827700\mathrm{E}{-01}$
α_{72}	$=$	$5.46053615165564280\mathrm{E}{-01}$	γ_{72}	$=$	$1.27246898098887760\mathrm{E}{+00}$
α_{73}	$=$	$-0.83019223550177171\mathrm{E}{-01}$	γ_{73}	$=$	$2.76051104051963040\mathrm{E}{-01}$
α_{74}	$=$	$-2.04081858154657380\mathrm{E}{-01}$	γ_{74}	$=$	$-7.36742072463449800\mathrm{E}{-01}$
α_{75}	$=$	$0.91988910804203419\mathrm{E}{-01}$	γ_{75}	$=$	$-0.00714432229676680\mathrm{E}{-01}$
α_{76}	$=$	$3.79901124490714730\mathrm{E}{-01}$	γ_{76}	$=$	$-1.15981504570537220\mathrm{E}{+00}$
b_1	$=$	$1.62567095340526680\mathrm{E}{-01}$	$\hat{b}_1$	$=$	$2.57613752241026340\mathrm{E}{-01}$
b_2	$=$	$1.81852259615444130\mathrm{E}{+00}$	$\hat{b}_2$	$=$	$7.02315222688898570\mathrm{E}{-01}$
b_3	$=$	$1.93031880501786460\mathrm{E}{-01}$	$\hat{b}_3$	$=$	$-1.72527430916564220\mathrm{E}{-01}$
b_4	$=$	$-9.40823930618112870\mathrm{E}{-01}$	$\hat{b}_4$	$=$	$-2.08585453014037690\mathrm{E}{-01}$
b_5	$=$	$0.91274478574529361\mathrm{E}{-01}$	$\hat{b}_5$	$=$	$0.58556352893451687\mathrm{E}{-01}$
b_6	$=$	$-7.79913921214651200\mathrm{E}{-01}$	$\hat{b}_6$	$=$	$3.62627556107225300\mathrm{E}{-01}$
b_7	$=$	$4.55341801261479050\mathrm{E}{-01}$	$\hat{b}_7$	$=$	$0.00000000000000000\mathrm{E}{+00}$

methods involved regarding $s = 6$ internal stages. Moreover, the embedded schemes of GROW37n and GROW37n2 also satisfy the conditions by Lubich and Ostermann [42] as well as those introduced by Rang [59, 60] up to order $p = 2$.

Another important difference between the schemes considered below is given by the value of γ used. With respect to GROW37nr and GROW37n the prescribed parameter $\gamma \approx 0.45$ is considered by including the additional condition

$$\gamma = \frac{1 + \sqrt{3}}{6}$$

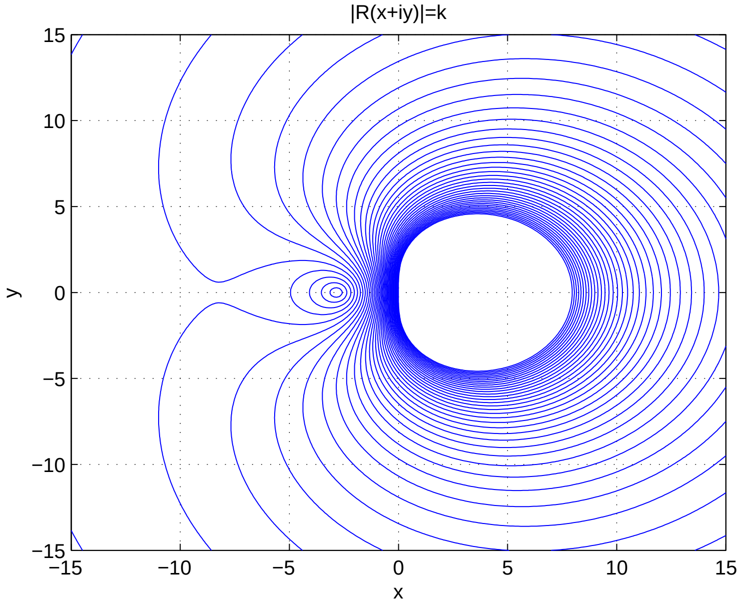


Figure 5.7: Stability region of GROW37nr regarding `k=0:0.025:1`.

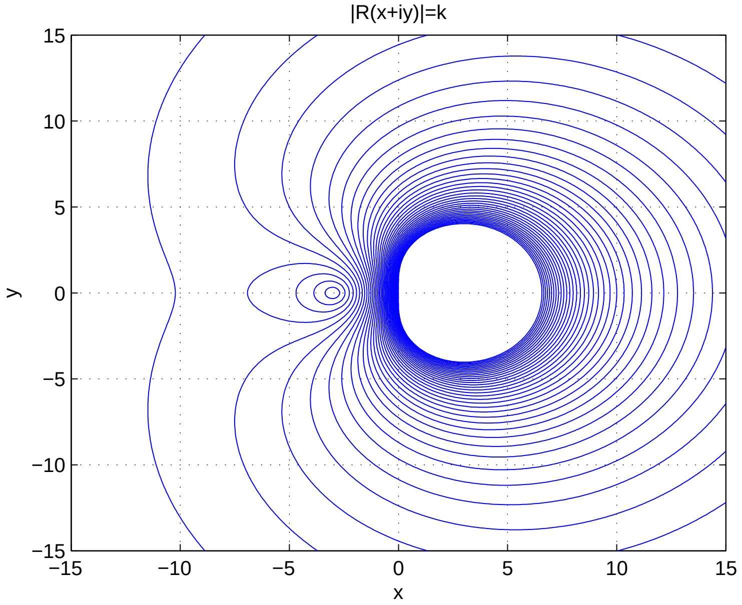


Figure 5.8: Stability region of GROW37n regarding `k=0:0.025:1`.

Table 5.9: Set of Coefficients for GROW37n2

$\gamma = 2.10537184485118880\text{E}{-01}$			
α_{21}	$= 4.21074368968140600\text{E}{-01}$	γ_{21}	$= -4.21074368968948400\text{E}{-01}$
α_{31}	$= 2.79917422879864820\text{E}{-01}$	γ_{31}	$= -1.84958978473185100\text{E}{-01}$
α_{32}	$= 2.83174778822673390\text{E}{-01}$	γ_{32}	$= -1.88216334415097250\text{E}{-01}$
α_{41}	$= 2.94412032869762920\text{E}{-01}$	γ_{41}	$= -0.76907157582717905\text{E}{-01}$
α_{42}	$= 2.84868609445784850\text{E}{-01}$	γ_{42}	$= -1.73691471529902030\text{E}{-01}$
α_{43}	$= 1.72518580805914280\text{E}{-01}$	γ_{43}	$= -0.34875934037295372\text{E}{-01}$
α_{51}	$= 2.65442828751577760\text{E}{-01}$	γ_{51}	$= -0.03161731203265310\text{E}{-01}$
α_{52}	$= -4.08963317964040120\text{E}{-01}$	γ_{52}	$= 1.51570666360029800\text{E}{-01}$
α_{53}	$= 7.96634514342330570\text{E}{-01}$	γ_{53}	$= 0.67741307897286968\text{E}{-01}$
α_{54}	$= -0.23535644633709309\text{E}{-01}$	γ_{54}	$= -3.92409221925114240\text{E}{-01}$
α_{61}	$= 2.14920375331621390\text{E}{-01}$	γ_{61}	$= 1.45388102958351280\text{E}{-01}$
α_{62}	$= 0.05358020624551900\text{E}{-01}$	γ_{62}	$= 1.53925393948013470\text{E}{-01}$
α_{63}	$= 2.52523457152398210\text{E}{-01}$	γ_{63}	$= -2.34299540127013540\text{E}{-01}$
α_{64}	$= 1.93398518195710250\text{E}{-01}$	γ_{64}	$= 2.45965106390625550\text{E}{-01}$
α_{65}	$= 3.33799628695717110\text{E}{-01}$	γ_{65}	$= -5.21516247655095520\text{E}{-01}$
α_{71}	$= 1.79669939392207960\text{E}{-01}$	γ_{71}	$= 0.48127245090637989\text{E}{-01}$
α_{72}	$= -1.34416560236480460\text{E}{-01}$	γ_{72}	$= 1.18123375465308660\text{E}{-01}$
α_{73}	$= 6.04712167158606980\text{E}{-01}$	γ_{73}	$= -2.71214165874376860\text{E}{-01}$
α_{74}	$= -2.02516064603826870\text{E}{-01}$	γ_{74}	$= 3.01888584184993200\text{E}{-01}$
α_{75}	$= 4.97293964155389350\text{E}{-01}$	γ_{75}	$= -1.96925038866290010\text{E}{-01}$
α_{76}	$= 0.55256554134096809\text{E}{-01}$	γ_{76}	$= -2.10537184485375890\text{E}{-01}$
b_1	$= 2.27797184482845920\text{E}{-01}$	$\hat{b}_1$	$= 3.60308478289972700\text{E}{-01}$
b_2	$= -0.16293184771178960\text{E}{-01}$	$\hat{b}_2$	$= 1.59283414572565150\text{E}{-01}$
b_3	$= 3.33498001284230410\text{E}{-01}$	$\hat{b}_3$	$= 0.18223917025384881\text{E}{-01}$
b_4	$= 0.99372519581170660\text{E}{-01}$	$\hat{b}_4$	$= 4.39363624586336490\text{E}{-01}$
b_5	$= 3.00368925289095420\text{E}{-01}$	$\hat{b}_5$	$= -1.87716618959377720\text{E}{-01}$
b_6	$= -1.55280630351278840\text{E}{-01}$	$\hat{b}_6$	$= 2.10537184485118880\text{E}{-01}$
b_7	$= 2.10537184485118880\text{E}{-01}$	$\hat{b}_7$	$= 0.00000000000000000\text{E}{+00}$

when defining the equations to solve by the numerical method. The given equation follows from [79]. Here, a strongly A-stable W method with $s = 7$ stages is mentioned that is characterized by classical order $p = 3$ and the value of γ given. In contrast to GROW37nr and GROW37n the set of coefficients denoted by GROW37n2 considers $\gamma \approx 0.21$. It corresponds to a value that is entirely determined by the numerical method for finding an appropriate set of coefficients. Note that this parameter, contrary to $\gamma \approx 0.45$ lies within the regions considered by Table 6.4 in Section IV of [24]. However, the stability region resulting for this scheme seems less reliable.

Remark 5.8 *In [79] the set of coefficients for realizing the W method with* $s = 7$

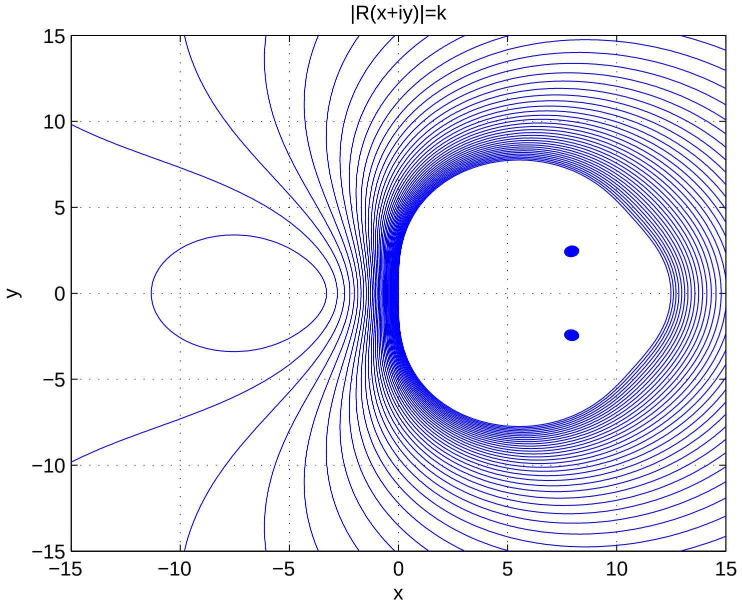


Figure 5.9: Stability region of GROW37n2 regarding `k=0:0.025:1`.

stages mentioned is not given explicitly. Hence, there was no opportunity of using its parameters as an initial guess of possible solutions for the numerical method applied. For that reason, initial guesses for finding a solution were chosen randomly.

The set of coefficients with respect to GROW37nr is summarized in Table 5.7. Its stability region is shown in Figure 5.7. Note that the embedded method of GROW37nr satisfies $|\tilde{R}(\infty)| \approx 0.62$. The set of coefficients with respect to GROW37n is summarized in Table 5.8. Its stability region is shown in Figure 5.8. Note that the embedded method of GROW37n satisfies $|\tilde{R}(\infty)| \approx 0.46$. The set of coefficients with respect to GROW37n2 is summarized in Table 5.9. Its stability region is shown in Figure 5.9. Note that the embedded method of GROW37n2 satisfies $|\tilde{R}(\infty)| = 0$.

5.3.3 Order Three with Eight Stages

In [79] Strehmel et al. introduce a W method of classical order $p = 3$ with $s = 8$ internal stages that also allows for non-exact Jacobian entries when solving singularly perturbed systems. In this context, it is constructed for considering $A_y = f_y + \mathcal{O}(h)$, $A_z = f_z + \mathcal{O}(h)$, $B_y = g_y + \mathcal{O}(h)$, $B_z = g_z + \mathcal{O}(h)$ as well as $A_y = 0$, $A_z = 0$, $B_y = g_y + \mathcal{O}(h)$, $B_z = g_z + \mathcal{O}(h)$. However, the method is not intended to preserve full order $p = 3$ when solving differential components explicitly (i.e. regarding $A_y = 0$ and $A_z = 0$). Instead, it should achieve order $p = 2$ in this case only [79]. The given method satisfies the additional conditions by Lubich [42] and is suitable to be applied

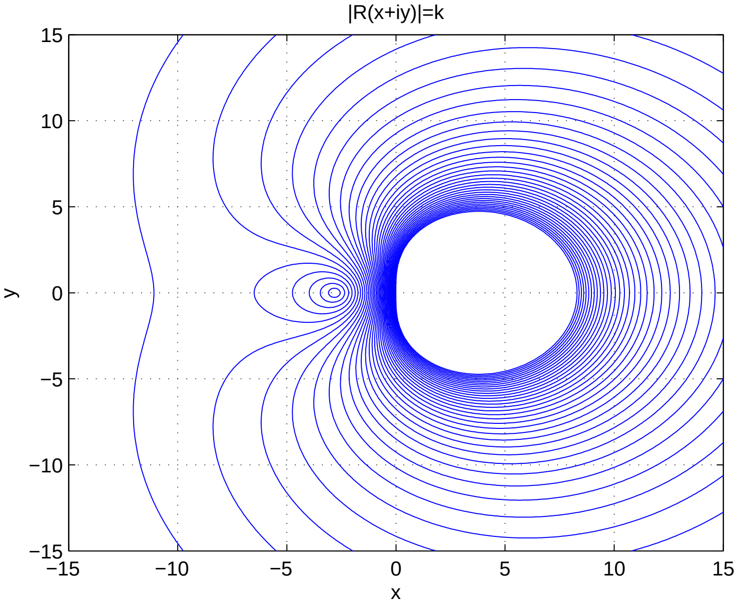


Figure 5.10: Stability region of GROW38n.

to DAEs of index one as well. Note that it satisfies the order conditions for considering $B_z = g_z + \mathcal{O}(h)$, too.

Below, the method by Strehmel et al. [79] is denoted by ROS38. Its set of coefficients violates order conditions (3D5.0), (3D6.0), (3D8.0) - (3D10.0) as well as (2A4.0). In addition, it does not satisfy all the conditions introduced by Rang in [59, 60] and considers no appropriate embedded method for step-size control. Because it is possible to realize corresponding schemes using $s = 7$ stages there is an attempt to extend the method so that it preserves order $p = 3$ with respect to all the Jacobian approximations allowed by ROW-type scheme (2.3) in the following. For that purpose, the Levenberg-Marquardt approach is applied to solve all the order conditions given using parameters of ROS38 as initial values. The resulting method is not intended to be stiffly accurate. Nevertheless, it satisfies $|R(\infty)| = 0$ while its embedded scheme satisfies $|\tilde{R}(\infty)| \approx 0.98$.

The resulting method is called GROW38n below. The corresponding set of coefficients is summarized in Table 5.10. Its stability region is shown in Figure 5.10.

Table 5.10: Set of Coefficients for GROW38n

γ = 4.35866521508458840E−01					
α_{21}	=	5.74499645467156820E−01	γ_{21}	=	−4.07383313415718480E−01
α_{31}	=	3.70667286022747180E−01	γ_{31}	=	−4.48083069303915560E−01
α_{32}	=	−0.00000000000028440E−01	γ_{32}	=	−5.82397595504767660E−01
α_{41}	=	1.89069678866929490E−01	γ_{41}	=	−2.23093486991461450E−01
α_{42}	=	−0.50495716743559897E−01	γ_{42}	=	−1.89397159790451410E−01
α_{43}	=	1.60674437190776200E−01	γ_{43}	=	−1.99487232299956380E−01
α_{51}	=	2.40285389918223320E−01	γ_{51}	=	−0.02572528296770120E−01
α_{52}	=	−0.09439202477800800E−01	γ_{52}	=	1.21751849922096130E−01
α_{53}	=	2.58850800828379690E−01	γ_{53}	=	2.37152916341893230E−01
α_{54}	=	−2.94438905508506270E−01	γ_{54}	=	−6.43187535044199790E−01
α_{61}	=	0.95859252942120762E−01	γ_{61}	=	3.27089886132997830E−01
α_{62}	=	1.52510534091371640E−01	γ_{62}	=	1.23932184835040450E−01
α_{63}	=	−9.96784664127383540E−01	γ_{63}	=	1.16925851607248550E+00
α_{64}	=	0.67654221694437910E−01	γ_{64}	=	−1.13099394061565260E−01
α_{65}	=	2.15412035904427720E+00	γ_{65}	=	−2.15412035905130720E+00
α_{71}	=	2.82938589018596920E−01	γ_{71}	=	0.47035619212783382E−01
α_{72}	=	0.73581910626216995E−01	γ_{72}	=	1.71683090676622490E−01
α_{73}	=	−8.40167841389465650E−01	γ_{73}	=	9.08261715571086480E−01
α_{74}	=	−1.57664016558605460E−01	γ_{74}	=	−1.07406741328926280E−01
α_{75}	=	1.98723090598668370E+00	γ_{75}	=	−1.97909821855923870E+00
α_{76}	=	−0.82477285751160867E−01	γ_{76}	=	0.20066617068320389E−01
α_{81}	=	1.19544410512615480E−01	γ_{81}	=	−1.40124102806272380E−01
α_{82}	=	−1.98465435608939890E−01	γ_{82}	=	−0.99118871185485180E−01
α_{83}	=	−1.33929979896610860E−00	γ_{83}	=	1.07670149356397890E+00
α_{84}	=	0.55475800593467912E−01	γ_{84}	=	2.05986246939261750E+00
α_{85}	=	2.01758363165588190E+00	γ_{85}	=	−1.74838979023465480E+00
α_{86}	=	−0.47794188653445778E−01	γ_{86}	=	−1.70463424954769960E+00
α_{87}	=	6.00047104876716640E−01	γ_{87}	=	1.24487282055110170E+00
b_1	=	4.37528037047676620E−01	$\hat{b}_1$	=	1.42990598059708930E−01
b_2	=	−0.02198680471591410E−01	$\hat{b}_2$	=	0.39053303055184091E−01
b_3	=	−0.22945058688240601E−01	$\hat{b}_3$	=	0.10338827192217509E−01
b_4	=	−1.70158991218053810E−01	$\hat{b}_4$	=	3.34804002842223610E−01
b_5	=	6.86992907932635570E−01	$\hat{b}_5$	=	4.55039511853233820E−01
b_6	=	−2.92430241839366330E−01	$\hat{b}_6$	=	−0.21022453330550368E−01
b_7	=	2.55842374016068630E−01	$\hat{b}_7$	=	0.15479990859941960E−01
b_8	=	1.07369653220871360E−01	$\hat{b}_8$	=	0.23316219468040449E−01

Table 5.11: Properties of given methods.

Method	s	p (GROW)	p (Roche)	p (W)	p (Rentrop)	$\tilde{p}$	$\|R(\infty)\|$	$\|\tilde{R}(\infty)\|$
ROS2	2	1	2	2	2	1	0	-
ROS2S	3	1	2	2	2	2	0	0.33
ROS3P	3	1	3	1	1	2	0.73	0.73
ROS34PW2	4	2	3	3	3	2	0	0.48
ROS34PRW	4	2	3	3	3	2	0	0.25
ROS34PRw	4	1	3	2	2	3	0	0.25
ROS3PRL2	4	1	3	1	1	3	0	0.25
ROS38	8	2	3	3	2	2	0	-
GROW2	3	2	2	2	2	1	0	0.88
GROW2S	3	2	2	2	2	2	0	0.33
GROW3P	3	2	3	2	2	2	0.73	0.73
GROW34PRw	4	2	3	2	2	3	0	0.25
GROW3PRL2	4	2	3	2	2	3	0	0.25
GROW35n	5	2	3	2	3	3	0	0.0023
GROW37nr	7	3	3	3	3	3	0	0.62
GROW37n	7	3	3	3	3	3	0	0.46
GROW37n2	7	3	3	3	3	3	0	0
GROW38n	8	3	3	3	3	3	0	0.98

Table 5.12: B_{PR}-consistency of given methods.

Method	Lubich	Condition 5.3		Condition 5.4					
		$k = 2$	3	$k = 3$	4	5	4	5	6
				$l = 1$	2	3	1	2	3
ROS2	-	-	-	-	-	-	-	-	-
ROS2S	x	x	x	x	x	x	-	x	-
ROS3P	x	x	x	x	x	x	-	-	-
ROS34PW2	x	x	x	x	x	x	-	-	-
ROS34PRW	x	x	x	x	x	x	x	-	-
ROS34PRw	x	x	x	x	x	x	x	x	x
ROS3PRL2	x	x	x	x	x	x	x	x	x
ROS38	x	x	x	x	x	x	-	-	-
GROW2	-	x	x	-	-	-	-	-	-
GROW2S	x	x	x	x	x	x	-	x	-
GROW3P	x	x	x	x	x	x	-	-	-
GROW34PRw	x	x	x	x	x	x	x	x	x
GROW3PRL2	x	x	x	x	x	x	x	x	x
GROW35n	x	x	x	x	x	x	x	x	x
GROW37nr	x	x	x	x	x	x	x	x	x
GROW37n	x	x	x	x	x	x	x	x	x
GROW37n2	x	x	x	x	x	x	x	x	x
GROW38n	x	x	x	x	x	x	x	x	x

6 Convergence Tests

In this section characteristics of the different sets of coefficients derived are analyzed. The major objective is to prove that the properties theoretically satisfied by the order conditions taken into account are present with respect to some practical application. For this purpose, convergence tests are considered that study the behavior of the given methods when solving DAE systems by means of non-exact Jacobian entries. In addition, the convergence behavior regarding the Prothero-Robinson problem with respect to different degrees of stiffness is shown.

6.1 Dependence on Different Jacobian Expressions

Applications within this section deal with the convergence behavior of the schemes previously introduced when computing academic test problems with known analytical solution. For this purpose, three problems are considered that describe semi-explicit DAE systems of index one. In order to examine the reliability of the new order conditions derived, i.e. testing if the orders of convergence predicted can be preserved by the schemes applied, effects of using non-exact Jacobian expressions are analyzed by solving the given test problems by means of time-lagged or completely neglected Jacobian entries.

Remark 6.1 *Note that the sets of coefficients were implemented using a formulation of the ROW-type method (2.3) that allows to avoid matrix-vector multiplications. Corresponding transformed versions of ROW-type schemes are generally considered in practice. For further details see, for example, [35] and [81].*

All test problems within this section are solved by means of fixed step-sizes h. The different Jacobian expressions used are determined analytically. If not mentioned otherwise, resulting global errors are computed in discrete L_2-norm by

$$err = ||\tilde{y}_{num}(x_{end}) - \tilde{y}_{ana}(x_{end})||_2 \tag{6.1}$$

where $\tilde{y} = (y, z)^T$. Given terms $\tilde{y}_{num}$ and $\tilde{y}_{ana}$ denote the numerical and the analytical solution, respectively. The resulting order of convergence then is given by [56]:

$$q = log_2 \left(\frac{err_{2h}}{err_h} \right). \tag{6.2}$$

6.1.1 Test 1

The first test problem considered was originally introduced by Rentrop et al. in [62]. It reads:

$$\begin{aligned} y_1' &= \frac{1}{2} y_2^3 z \\ y_2' &= \frac{1}{6} y_2 z \\ 0 &= z + 6 \frac{y_1}{y_2^3}. \end{aligned} \tag{6.3}$$

The corresponding analytical solution is given by:

$$y_1(x) = \exp(-3x), \qquad y_2(x) = \exp(-x), \qquad z(x) = -6.$$

Note that this test problem was constructed for analyzing partitioned schemes that are characterized by solving the differential parts of a given DAE system by means of underling explicit Runge-Kutta strategies. Hence, the differential parts of (6.3) are assumed to be not stiff [62]. As well, there is $g_z = 1$. So, the index-1 condition is always satisfied [62]. The test problem is solved regarding step-sizes $h = 1/(100 \cdot 2^k)$ with $k = 0, 1, ..., 5$ in the time interval $x = [0, 0.5]$ below. Consistent initial values are given by [62]:

$$y_1(0) = 1, \qquad y_2(0) = 1, \qquad z(0) = -6.$$

Exact Jacobian

In a first analysis given system (6.3) is solved by means of exact Jacobian expressions that are updated with every time-step. So, there is $A_y = (f_y)_0$, $A_z = (f_z)_0$, $B_y = (g_y)_0$ and $B_z = (g_z)_0$. As a consequence, the schemes applied are used in accordance with the original formulation of ROW methods for DAEs introduced by Roche in [64]. The sets of coefficients listed in Table 5.11 are considered. In addition, method RODASP [72] is taken into account. RODASP corresponds to a ROW method for solving DAEs with exact Jacobian up to order $p = 4$.

Results are shown in Figure 6.1. Note that the dotted lines depict reference slopes with respect to orders $p = 1$, $p = 2$ and $p = 3$. All the methods reach the order of convergence they were originally constructed for. So, there is order $p = 2$ for ROS2 and ROS2s as well as GROW2 and GROW2s, order $p = 4$ for RODASP and order $p = 3$ for all further schemes considered. Hence, the schemes are in accordance with the expected orders of convergence according to Table 5.11 (see column denoted by: p (Roche)).

Explicit Solution of Differential Functions

In a second analysis system (6.3) is solved using a Jacobian expression whose entries with respect to given differential functions are all set to zero. Jacobian entries with respect to remaining algebraic functions remain exact and are updated with every time-step. So, there is $A_y = 0$, $A_z = 0$, $B_y = (g_y)_0$ and $B_z = (g_z)_0$. As a consequence, the given schemes reduce to the formulation of underlying explicit Runge-Kutta methods with respect to given differential parts while they preserve the formulation of classical linearly implicit ROW methods with respect to given algebraic parts. So, the strategy introduced by Rentrop et al. in [62] is considered. Note that this strategy is only advisable when given differential functions are known to be not stiff.

Results are shown in Figure 6.2. As expected most of the schemes known from literature show effects of order reduction. Only ROS2 and ROS2s are able to preserve their original order $p = 2$. Also, ROS34PW2 and ROS34PRW are able to preserve their original order $p = 3$. However, RODASP decreases order from $p = 4$ to $p = 1$.

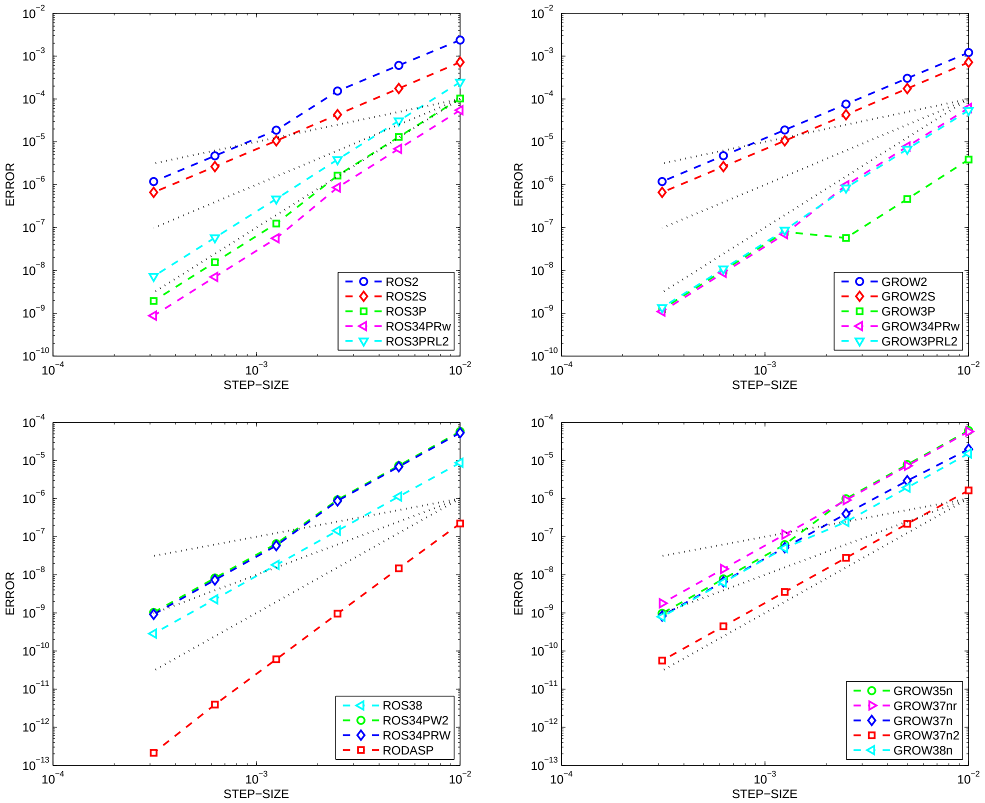


Figure 6.1: Step-size versus error for (6.3) using exact Jacobian.

With respect to schemes of third order ROS34PRw and ROS38 decrease from $p = 3$ to $p = 2$, ROS3P and ROS3PRL2 decrease from $p = 3$ to $p = 1$. There are no effects of order reduction regarding the new schemes introduced. The methods analytically derived preserve order $p = 2$ while the methods numerically derived preserve order $p = 3$. In this context, especially GROW3P and GROW3PRL2 are able to prevent the order reduction that can be observed for ROS3P and ROS3PRL2. Note that all these results are in accordance with the expected orders of convergence according to Table 5.11 (see column denoted by: p (Rentrop)).

Time-Lagged Jacobian Information

In a third test effects of using time-lagged Jacobian information are analyzed. In this context, approximated Jacobian entries satisfying $A_y = f_y + \mathcal{O}(h)$, $A_z = f_z + \mathcal{O}(h)$ and $B_y = g_y + \mathcal{O}(h)$ are considered. For this purpose, computing these entries is given just after a predefined number of time-steps. Note that in this analysis Jacobian components with respect to sub-matrix B_z are updated with every time-step still, i.e.

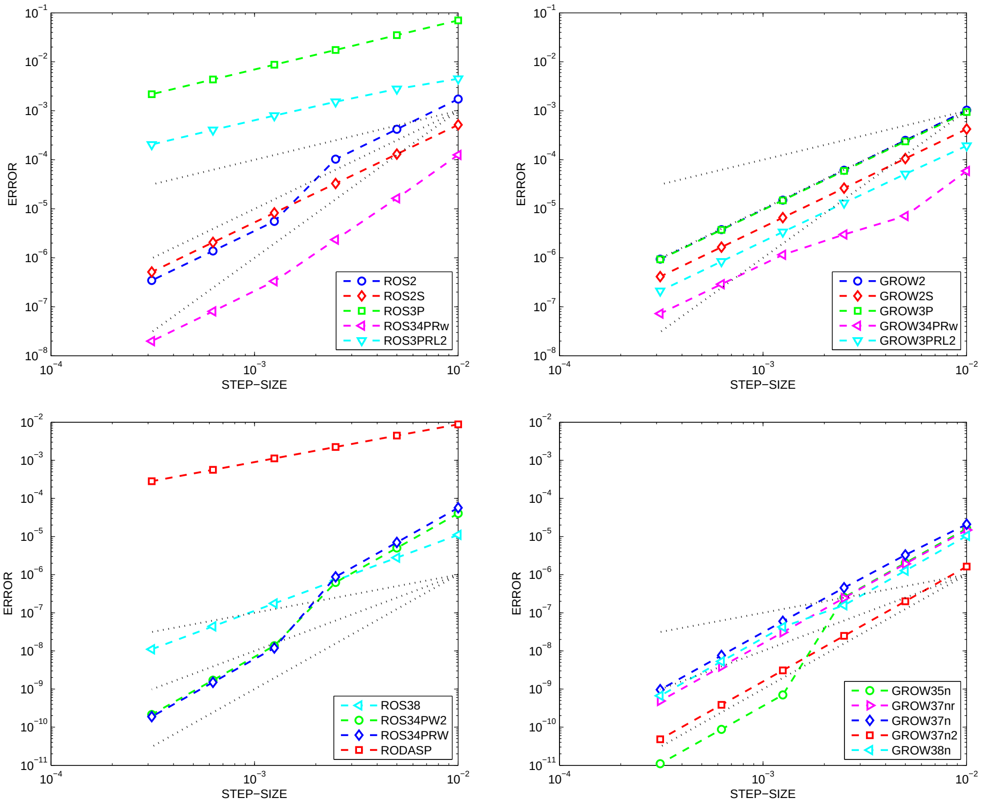


Figure 6.2: Step-size versus error for (6.3) after setting Jacobian entries with respect to A_y and A_z to zero.

$B_z = (g_z)_0$. Regarding efficiency, this strategy might range in between the two analyses considered previously. As computing differential parts by linearly implicit strategies is not omitted completely it is more costly than solving them explicitly with every time-step. However, it is less costly than using linearly implicit strategies completely with every time-step. Moreover, contrary to the second analysis considered before, this strategy might be applicable regarding DAE problems characterized by stiff differential functions, too.

The analysis is realized by considering an update of the Jacobian entries after 5, 10 and 20 time-steps. Results for updates after 5 time-steps are given in Figure 6.3. Results for updates after 10 time-steps are given in Figure 6.4. Results for updates after 20 time-steps are given in Figure 6.5. The pictures show that ROS2 and ROS2s are able to preserve their order $p = 2$ for all the three cases of Jacobian updates considered. Interestingly, ROS2 shows a convergence of orders higher than $p = 3$ for smaller step-sizes when the update is given just after 20 time-steps. RODASP decreases its order from $p = 4$ to $p = 2$. ROS3P decreases its order from $p = 3$ to

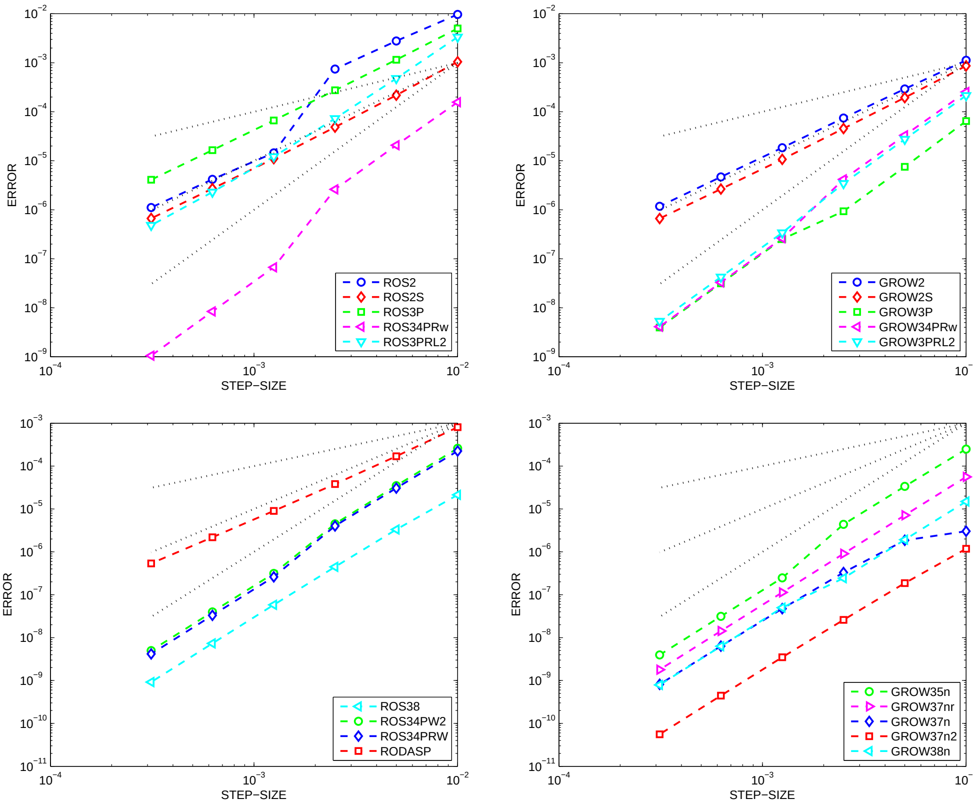


Figure 6.3: Step-size versus error for (6.3) using Jacobian updates after 5 steps.

$p = 2$. ROS34PW2, ROS34PRW, ROS34PRw, ROS3PRL2 and ROS38 are able to preserve their original order $p = 3$ for all the three cases considered. However, their efficiency is decreasing the less frequently Jacobian updates are performed. In fact, it seem reasonable that most classical schemes do not show significant effects of order reduction for the test scenario considered. This is because the given approximation with respect to the Jacobian entries is quite special and requires just a reduced number of additional conditions to preserve the order of convergence.

Among the new schemes introduced, GROW2 and GROW2S preserve order $p = 2$ and even show no significant loss of efficiency when realizing the Jacobian updates less frequently. GROW3P, GROW34PRw and GROW3PRL2 reach order $p = 3$. But they show some decrease of efficiency when realizing Jacobian updates after a larger number of time-steps, too. The same behavior is given for GROW35n. GROW37nr, GROW37n, GROW37n2 and GROW38n are able to preserve order $p = 3$ without significant loss of efficiency. Note that these are the schemes with the largest numbers of internals stages. Also, they satisfy the order conditions for additional approximations with respect to Jacobian entries B_z.

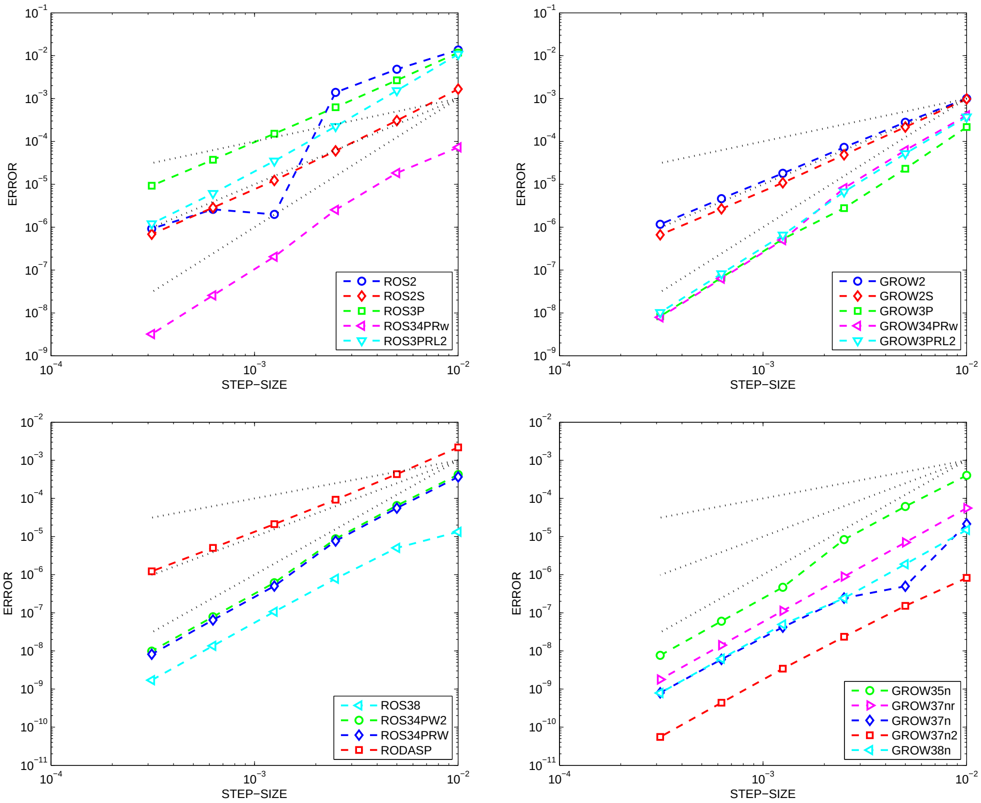


Figure 6.4: Step-size versus error for (6.3) using Jacobian updates after 10 steps.

The loss of efficiency given for several sets of coefficients when using time-lagged Jacobian information proves to be quite a drawback. In fact, to meet prescribed error tolerances when solving a DAE system in practice, it might cause an increased number of required time-steps and therefore lead to disadvantages regarding computational efficiency. This drawback might be no longer relevant when the computational effort for time-steps that reuse old Jacobian information is significantly smaller than for time-steps computing a complete new Jacobian. However, regarding the given test problem that is characterized by a small number of differential and algebraic equations corresponding differences considering computing times cannot be shown in detail. This means, there is no significant difference regarding computing times when solving the given problem with regular or time-lagged Jacobian information for the sets of coefficients applied. Nevertheless, in this context, Figure 6.6 exemplarily depicts error versus computing time with respect to different strategies of Jacobian updates for some of the sets of coefficients introduced: GROW2S, GROW3PRL2, GROW37nr and GROW37n.

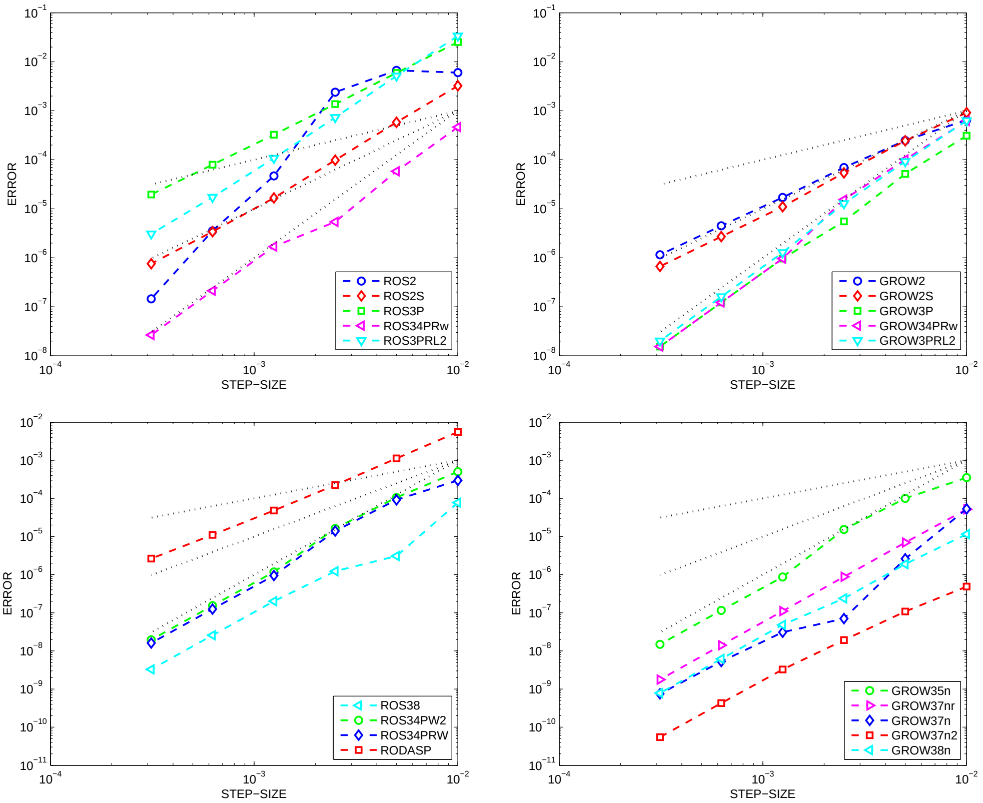


Figure 6.5: Step-size versus error for (6.3) using Jacobian updates after 20 steps.

Additionally Neglecting Algebraic Components

In a fourth test the effects of neglecting all Jacobian entries with respect to differential equations as well as additional components considering given algebraic equations are analyzed. For this purpose, test problem (6.3) is solved using $A_y = 0$, $A_z = 0$ and $B_y = 0$. The remaining Jacobian entries are updated regularly, i.e. $B_z = (g_z)_0$ is considered. This analysis can be interpreted as an extension of the case investigated by Rentrop et al. in [62]. It corresponds to the most critical test scenario for the sets of coefficients applied because it checks their reliability when taking into account arbitrary Jacobian entries with respect to given algebraic components. Note that this strategy allows to avoid the computation with respect to columns that correspond to derivatives with respect to differential components y completely. Thus, it is quite promising when having to apply algorithms to compute the Jacobian numerically, especially when solving DAEs that are characterized by extensive differential parts.

Results are shown in Figure 6.7. They show that RODASP reduces its order of convergence from $p = 4$ to $p = 1$. ROS3P and ROS3PRL2 reduce their order of

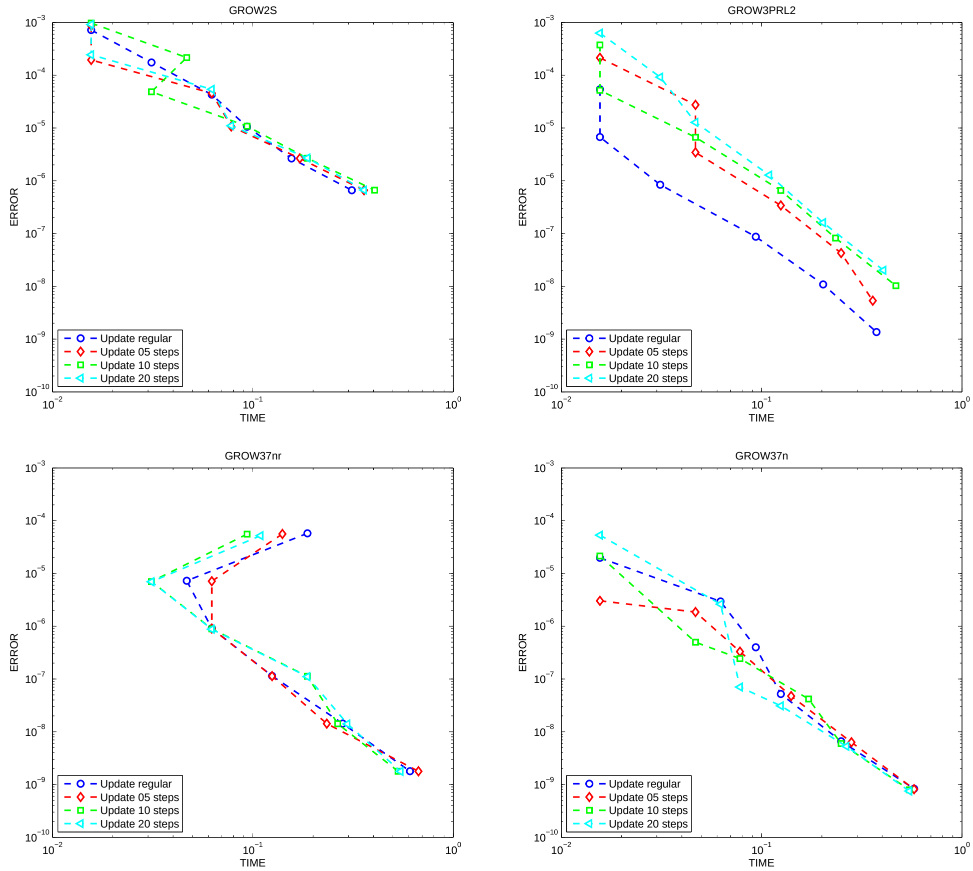


Figure 6.6: Computing times versus error for different strategies of Jacobian updates.

convergence from $p = 3$ to $p = 1$. All the other schemes known from literature generally reduce their order from $p = 3$ to $p = 2$. An exception is ROS38 that preserves its original order $p = 3$. Also, ROS2 and ROS2S are able to preserve their original order $p = 2$. The new schemes introduced preserve their order of convergence, i.e. order $p = 2$ for the schemes determined analytically as well as GROW35n and order $p = 3$ for all the remaining schemes determined numerically. In this context, especially GROW3P and GROW3PRL2 are able to prevent the order reduction that can be observed for ROS3P and ROS3PRL2 as it was the case when using Jacobian approximations $A_y = 0$ and $A_z = 0$ together with $B_y = (g_y)_0$ and $B_z = (g_z)_0$ already.

Note that the schemes ROS2, ROS2S, ROS34PRw and ROS38 perform much better than suggested by Table 5.11 (see column denoted by: p (GROW)). In fact, ROS2, ROS2S and ROS34PRw should be of order $p = 1$ and ROS38 should be of order $p = 2$. The higher orders reached seem to be a consequence of the given test equation. When

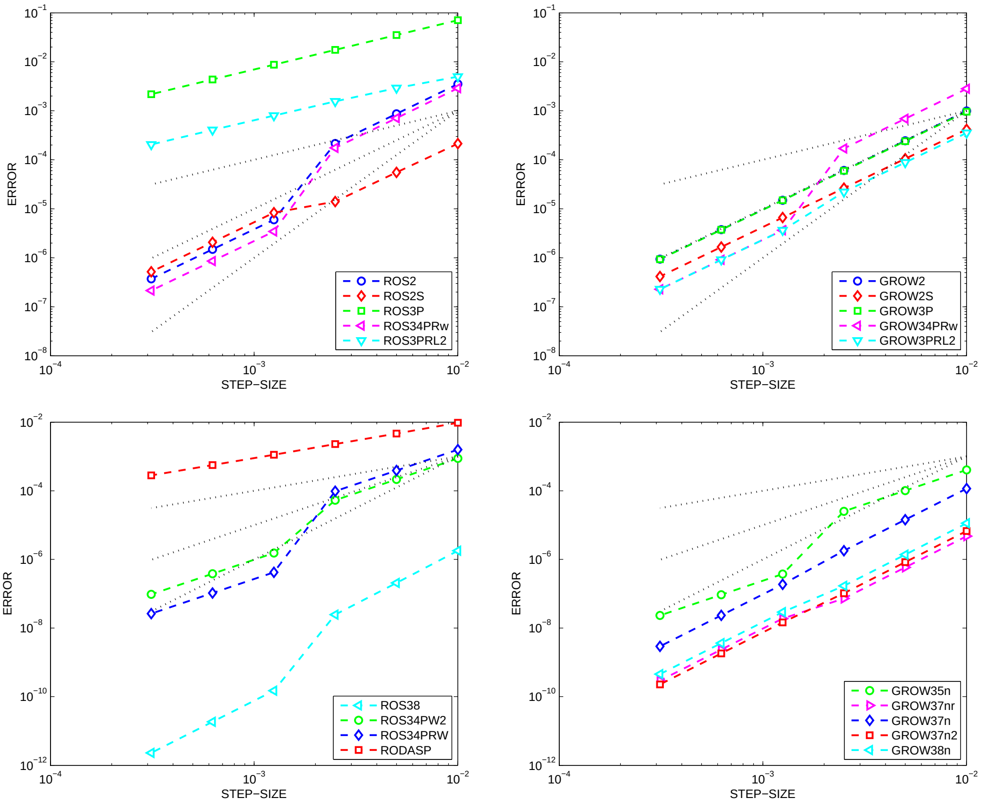


Figure 6.7: Step-size versus error for (6.3) using $A_y = 0$, $A_z = 0$ and $B_y = 0$.

regarding the additional test equations given below the same analysis leads to the orders of convergence expected for these schemes as well. All the other schemes show the order of convergence listed in Table 5.11 directly.

6.1.2 Test 2

Because not all the sets of coefficients considered showed the behavior suggested according to Table 5.11 when being applied to previous test problem (6.3) another DAE system is solved below. In this context, for further analyzing their orders of convergence the four scenarios with respect to given Jacobian entries are repeated. The second test problem also originates from [62]. It reads:

$$\begin{aligned} y_1' &= z_1 \\ y_2' &= -\frac{1}{2} z_2^{1/4} \qquad (6.4) \end{aligned}$$

$$
\begin{aligned}
0 &= y_1^2 + z_1^2 - \frac{y_2^4}{z_2} \\
0 &= z_2 - y_2^4.
\end{aligned}
$$

The analytical solution is given by:

$$y_1(x) = \sin(x), \qquad y_2(x) = \exp(-0.5x), \qquad z_1(x) = \cos(x), \qquad z_2(x) = \exp(-2x).$$

Note that the given test problem violates the index-1 assumption for $x = \pi/2$, i.e. g_z becomes a singular matrix. Hence, integrators constructed for index-1 problems might show some irregular behavior in the vicinity of this point in time [62]. For that reason, the problem is solved regarding step-sizes $h = 1/(100 \cdot 2^k)$ with $k = 0, 1, ..., 5$ in the time interval $x = [0, 1.5]$ below. Consistent initial values are given by [62]:

$$y_1(0) = 0, \qquad y_2(0) = 1, \qquad z_1(0) = 1, \qquad z_2(0) = 1.$$

The results for solving this test problem by regarding the exact and time-lagged Jacobian information as well as Jacobian entries reduced to algebraic equations are shown in Appendix B. They are not detailed here because the sets of coefficients applied show generally the same behavior as described with respect to test problem (6.3). Some differences are given by GROW37nr which shows order $p = 2$ instead of $p = 3$ when neglecting the Jacobian entries with respect to given differential components. However, this is not the case regarding the same scenario with respect to test (6.3) and test (6.5). Also, GROW37n2 shows some less consistent behavior regarding the orders of convergence reached. Applying time-lagged Jacobian information, especially GROW38n decreases its order of convergence from $p = 3$ regarding an update of the Jacobian entries after 5 time-steps to $p = 2$ regarding an update of the Jacobian entries after 20 time-steps. Note that this behavior is not observed regarding the same scenario with respect to test (6.3) and test (6.5), too. Also, the loss of efficiency given for the new sets of coefficients derived seems to be more present when using time-lagged Jacobian information.

Below, only the results given when using $A_y = 0$, $A_z = 0$ and $B_y = 0$ together with $B_z = (g_z)_0$ are considered in detail. They are shown in Figure 6.8. Again, schemes known from literature show a significant decrease regarding the order of convergence given. In fact, except for ROS34PW2, ROS34PRW and ROS38 that reduce to order $p = 2$ all the other schemes reduce to $p = 1$. So, sets of coefficients ROS2, ROS2s, ROS34PRw and ROS38 that were able to reach higher orders with respect to test (6.3) now show the behavior suggested in Table 5.11 (see column denoted by: p (GROW)). The new schemes introduced are able to preserve order. There is order $p = 2$ for the schemes determined analytically as well as GROW35n. So, the extension of the schemes known from literature avoids a decrease with respect to the order of convergence effectively. The remaining schemes determined numerically are able to preserve order $p = 3$. An exception is given for GROW34PRw that shows an order of convergence that is nearly $p = 1$ instead of $p = 2$. This negative effect, however, is not present when applying the given scenario with respect to Jacobian entries to test (6.3) and test (6.4).

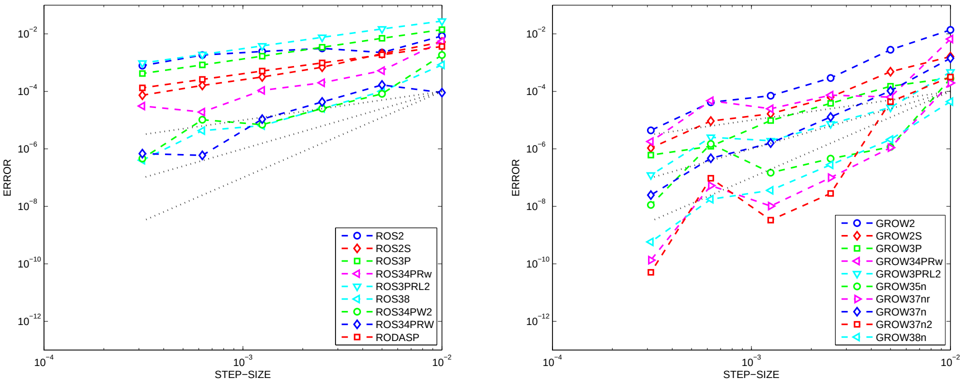


Figure 6.8: Step-size versus error for (6.4) using $A_y = 0$, $A_z = 0$ and $B_y = 0$.

6.1.3 Test 3

Test problems (6.3) and (6.4) yield no non-linear terms with respect to sub-matrix g_z given within the Jacobian. Hence, the third test problem was designed in order to have an appropriate DAE system for testing schemes that allow for approximations of the form $B_z = g_z + \mathcal{O}(h)$ in addition. The third test problem does not originate from literature. It reads:

$$\begin{aligned} y_1' &= 3y_2^2 y_3 - 3z_1^3 \\ y_2' &= y_3 \\ y_3' &= -y_2 \\ 0 &= y_1 - y_2^3 - z_1^3 \\ 0 &= z_1 - z_2^2. \end{aligned} \tag{6.5}$$

The analytical solution is given by:

$$y_1(x) = \exp(-3x) + \sin^3(x), \qquad y_2(x) = \sin(x), \qquad y_3(x) = \cos(x)$$

$$z_1(x) = \exp(-x), \qquad z_2(x) = \exp(-0.5x)$$

Note that the given system ensures a regular matrix g_z. Hence, the index-1 assumption is generally satisfied. The problem is solved regarding step-sizes $h = 1/(100 \cdot 2^k)$ with $k = 0, 1, ..., 5$ in the time interval $x = [0, 1.5]$ below. Consistent initial values are given by:

$$y_1(0) = 1, \qquad y_2(0) = 0, \qquad y_3(0) = 1, \qquad z_1(0) = 1, \qquad z_2(0) = 1.$$

The results for solving the test problem given by means of exact and time-lagged Jacobian information as well as Jacobian entries reduced to algebraic equations are shown in Appendix C. As for the previous test case (6.4) they are not considered in

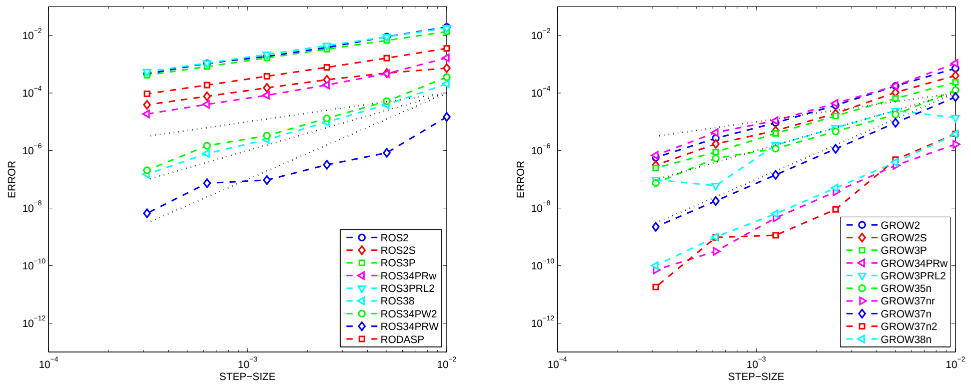


Figure 6.9: Step-size versus error for (6.5) using $A_y = 0$, $A_z = 0$ and $B_y = 0$.

detail here because the sets of coefficients show generally the same behavior described with respect to test equation (6.3). There are no mentionable differences.

Results with respect to the test scenario which regards Jacobian entries characterized by $A_y = 0$, $A_z = 0$, $B_y = 0$ and $B_z = (g_z)_0$ are shown in Figure 6.9. Here, the sets of coefficients generally show the behavior suggested by Table 5.11 (see column denoted by: p (GROW)) as well. There is a significant decrease regarding the orders of convergence reached by the methods known from literature. ROS34PW2 and ROS38 reduce their order from $p = 3$ to $p = 2$. All other schemes reduce their order to $p = 1$. Note that there is an exception with respect to ROS34PRW. Instead of reaching order $p = 2$ as it was the case for test (6.3) and test (6.4) it shows a tendency to order $p = 1$. The new schemes introduced, however, are able to preserve their order. There is order $p = 2$ for the sets of coefficients determined analytically and GROW35n. There is order $p = 3$ for the remaining sets of coefficients determined numerically.

6.2 Dependence on Stiffness Parameters

The new sets of coefficients were constructed such that they satisfy additional order conditions introduced by Rang [60, 60]. Intention of these order conditions is to reduce effects of order reduction that might occur when having to solve stiff problems. In order to check how the new methods derived perform with respect to this aspect, the test problem of Prothero and Robinson [54] is analyzed below. For this purpose, as considered in works by Scholz [68] and Rang [59, 60] equation (5.1) is solved together with

$$\varphi(x) = 10 - (10 + x)\exp(-x) \tag{6.6}$$

regarding the time interval $[0, 2]$. Step-sizes considered are $h = 0.1 \cdot 2^{-k}$ with $k = 0, ..., 5$. Stiffness parameter λ is chosen to be equal to $\lambda = -10^i$ with $i = 0, ..., 6$. The order of convergence is determined using the errors that result by comparing the numerical and the analytical solution. The errors are measured in the discrete L_2-norm.

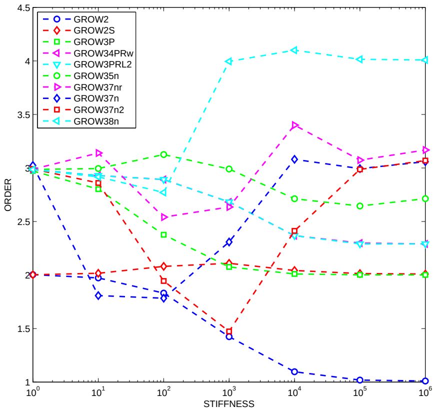


Figure 6.10: Stiffness versus order of convergence for the Prothero-Robinson example.

Note that only the new sets of coefficients introduced are applied. Corresponding results with respect to most of the schemes known from literature can be found in [59, 60].

An overview of orders of convergence reached for the different stiffness parameters considered is shown in Figure 6.10. For realizing the visualization, the mean values with respect to the different orders of convergence that are given for the different stepsizes considered are used. In this context, orders given with respect to error values close to machine precision are omitted as they falsify the given orders significantly (see [60]).

Results show that only few of the schemes derived preserve the order of convergence constantly. Some of the methods tend to decrease their order while some of the methods tend to increase their order when the stiffness given within the system rises. For example, GROW2 decreases order from $\tilde{p} = 2$ to $\tilde{p} = 1$. However, it is the only scheme that was not constructed satisfying the additional order condition. Hence, this behavior is less astonishing. Nevertheless, its enhanced version given by GROW2S preserves order $\tilde{p} = 2$ quite well. All the sets of coefficients determined analytically were constructed to preserve order $\tilde{p} = 2$ at least. Hence, a decrease of the order form $\tilde{p} = 3$ to $\tilde{p} = 2$ can be observed for GROW3P and a decrease from $\tilde{p} = 3$ to $\tilde{p} = 2.5$ can be observed for GROW34PRw and GROW34PRL2. In this context, it is interesting that GROW34PRw and GROW34PRL2 show exactly the same behavior without being directly related. The sets of coefficients determined numerically were constructed to preserve order $\tilde{p} = 3$ at least. However, only GROW35n and GROW37nr are able to preserve this order more or less constantly. GROW38n increases its order from $\tilde{p} = 3$ to $\tilde{p} = 4$ with increasing stiffness. So, additional conditions to reach order $\tilde{p} = 4$ seem to be satisfied by chance. GROW37n and GROW37n2 show a significant drop of their order from $\tilde{p} = 3$ to less than $\tilde{p} = 2$ with respect to low and medium stiffness parameters. Nevertheless, their order increases back to $\tilde{p} = 3$ when the stiffness gets

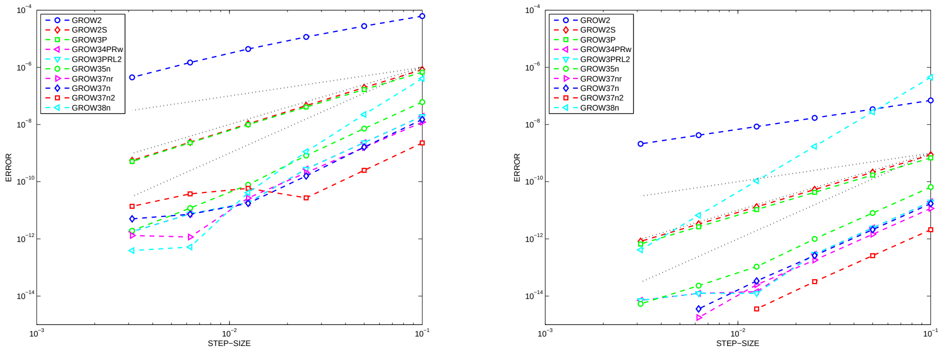


Figure 6.11: Step-size versus error with respect to the Prothero-Robinson example regarding $\lambda = -10^3$ (left) and $\lambda = -10^6$ (right).

higher.

A more detailed comparison regarding the performance of the different sets of coefficients that is given with respect to medium stiffness $\lambda = -10^3$ and high stiffness $\lambda = -10^6$ is shown in Figure 6.11. Here, the decrease of GROW2 from order $\tilde{p} = 2$ to $\tilde{p} = 1$ once more becomes obvious. GROW2S and GROW3P are of order $\tilde{p} = 2$. In contrast to the observations given regarding the mean values, GROW34PRw and GROW3PRL2 show order $\tilde{p} = 3$. The reducing of the order considering the mean values is a consequence of the fact that their order drops regarding small step sizes. A similar behavior can be observed at least for $\lambda = -10^6$ when regarding results of the related original sets of coefficients ROS34PRw and ROS3PRL2 presented in [59]. Among the set of coefficients determined numerically, GROW38n shows order $\tilde{p} = 4$. GROW37nr, GROW37n and GROW37n2 are of order $\tilde{p} = 3$. However, especially GROW37n2 drops its order significantly.

7 Conclusion and Outlook

Theory

A theory for deriving an enhanced class of linearly implicit ROW-type methods for solving semi-explicit systems of DAEs assumed to be of index one was introduced. Corresponding schemes are characterized by allowing for arbitrary Jacobian approximations with respect to its original entries f_y, f_z and g_y. Hence, the given theory in principle corresponds to a combination of the ideas introduced by Roche [64], i.e. ROW-type methods with exact Jacobian applied to DAEs, as well as Steihaug and Wolfbrandt [71], i.e. ROW-type methods with non-exact Jacobian applied to ODEs (i.e. W methods).

The theory derived is based on graphical representation via rooted trees as originally introduced by Butcher [6, 7, 8] with respect to explicit one-step Runge-Kutta schemes for ODEs. In fact, it combines the extensions of this strategy introduced by Roche [64] with respect to ROW methods for DAEs as well as described by Hairer and Wanner [24] with respect to W methods for ODEs. By taking into account new types of vertices for describing additionally occurring non-exact elementary differential components and coefficients it was possible to create the theory for deriving the schemes considered in a way that preserves corresponding theories with respect to classical schemes given in literature. In this context, especially the strategies known for constructing rooted trees and thus elementary differentials and coefficients could be directly covered. In fact, by regarding special Jacobian approximations, resulting order conditions combine each other appropriately in order to form the order conditions of different ROW-type schemes that were presented in literature already. While corresponding methods were considered separately so far, the extended theory introduced now allows for their unification and to recognize their close relationship to each other.

The theory introduced allows to determine the order conditions for realizing the new class of ROW-type schemes by means of rooted trees in a quite straight forward way. In this context, conditions were identified that cannot be found in common literature and that are necessarily required to preserve the order of convergence when taking into account non-exact Jacobian expressions. However, as usual for ROW-type schemes that are based on solution strategies considering non-exact Jacobian expressions, the number of order conditions drastically increases with respect to schemes of higher order. Nevertheless, not all the order conditions that are given with respect to resulting tree structures must be taken into account explicitly. In fact, a series of resulting conditions turn out to be redundant. Moreover, corresponding redundant conditions can be identified directly by means of special structures regarding rooted trees used for their description.

Methods

Methods derived show the relevance of the order conditions derived with respect to the tests applied for analyzing resulting orders of convergence. Only schemes satisfying the order conditions introduced were able to preserve their order with respect to all the test scenarios considered. In this context, schemes of order $p = 2$ were determined analytically with a minimum of $s = 3$ internal stages. Schemes of order $p = 3$ were determined numerically with a minimum of $s = 7$ internal stages. The sets of coefficients introduced were constructed such that they generally reach corresponding orders of B_{PR}-consistency according to Rang [59, 60]. Also, additional order conditions were taken into account to allow for higher orders of convergence when considering special Jacobian approximations. Note that the schemes determined analytically correspond to extensions of methods known from literature that usually suffer from order reduction when solving DAEs by means of non-exact Jacobian information. In fact, many schemes derived allow to preserve original properties of their predecessors and extend them for the purpose of using arbitrary Jacobian entries without having to increase the number of stages. This means, the properties of the schemes considered in literature can be increased significantly just by changing some of the coefficients given. In this context, especially the properties of ROS3P [35] could be enhanced: While preserving order $p = 3$ when solving DAEs by means of exact Jacobian expressions its extension to GROW3P allows for order $p = 2$ instead of $p = 1$ when solving DAEs by means of non-exact Jacobian expressions.

By allowing for arbitrary approximations with respect to Jacobian components mentioned the resulting ROW-type methods allow for realizing a wide range of different solution strategies in order to compute semi-explicit DAE systems of index-1. In this context, different strategies are generally characterized by using different extents of partial explicit integration via underlying one-step Runge-Kutta schemes or time-lagged Jacobian information. For that reason, together with the unifying character of the theory considered for their derivation, different schemes and their characteristic solution behavior presented in literature can be easily realized. In fact, different schemes do not have to be coded separately in order to combine and apply ROW-type schemes. They can be implemented just by adapting Jacobian entries appropriately. In this context, the adaptive character of the resulting generalized ROW-type schemes for computing given DAE systems holds much potential for reducing computational efforts. Computing the Jacobian is one of the most expensive operations when applying ROW-type schemes. When exploiting extended explicit integration by neglecting Jacobian entries, i.e. making use of much sparser Jacobian structures, as well as time-lagged Jacobian information corresponding costs can be reduced drastically. However, this demands for sets of coefficients that are able to realize corresponding strategies without showing a significant loss of efficiency. A loss of efficiency would cause an increase of time-steps and thus might counteract advantages that are given by the strategies mentioned.

Outlook

With respect to further investigations it would be promising to find schemes of order $p = 3$ with $s = 5$ or $s = 6$ internal stages in order to further reduce the stage number and, thus, corresponding computational effort. Theoretically it is possible to derive methods of order $p = 3$ with a minimum of $s = 5$ stages. However, it was not possible to realize such schemes so far. There is the possibility that conflicts with respect to resulting order conditions might prevent the realization of such methods. In this case it would be helpful to identify such conflicts and to document them for the purpose of more reliable strategies for realizing corresponding schemes. Also, with respect to the schemes introduced, it would be useful to identify the reasons why some of the given sets of coefficients show a significant decrease with respect to their overall efficiency when using time-lagged Jacobian information while others do not. Schemes that will allow for applying these strategies without decreasing their efficiency are more promising because they will not require a disadvantageous increase of time-steps in order to reach predefined tolerances. Maybe there are additional conditions that can be formulated and that are satisfied by some of the schemes considered already.

Considering given applications, the efficiency of the methods introduced must be further studied regarding large DAE systems that require for numerical Jacobian computations. In general the computation of the Jacobian is the most costly aspect when applying ROW-type methods. In fact, by allowing for the use of non-exact Jacobian entries much of this effort can be saved. When considering systems that consist of large numbers with respect to differential components using approximations such as $A_y = 0$ and $B_y = 0$ enables to save the computation of whole Jacobian columns. Corresponding systems especially occur in method of lines applications. Examples are network structures for simulating transport of gases and fluids. These are cases where the computational effort is expected to be reduced by the schemes introduced. Applications regarded within this thesis are way to small in order to show advantageous effects with respect to computing times, i.e. there is no significant difference in using a reduced non-exact Jacobian compared to using a complete exact Jacobian. Hence, implementing and analyzing corresponding test problems in detail is relevant.

In this context, routines for computing Jacobian expressions with respect to extended DAE systems require to be optimized. The properties of the schemes introduced cannot be fully exploited by the numerical methods present (e.g. MATLAB routine `numjac`). This is because corresponding methods are based on determining a complete, usually square Jacobian [9, 66]. As a consequence, it is difficult to realize an effective computation with respect to a special choice of reduced Jacobian entries, in particular a special choice of given partial derivatives. Besides, corresponding strategies must be further developed in order to exploit possibilities for saving computational efforts more effectively. This might include strategies for some automatic rearranging of given components and equations for the purpose of ending up with more advantageous Jacobian structures after having neglected several entries.

A familiar aspect that needs to be addressed is realizing automatic and dynamic routines for sparing Jacobian entries effectively in combination with reliable monitoring with respect to stiffness detection. It is one aspect to have an effective numerical approach for computing the Jacobian numerically after neglecting several entries. However, it is another aspect to determine corresponding elements not manually and to ensure that the components neglected will not affect the integration process negatively. So, it must be determined if neglecting given Jacobian entries is useful and reliable with respect to stiffness properties of the system to solve. To avoid accidental explicit integration of stiff elements by corresponding stiffness detection is advantageous especially regarding the solution of problems whose stiffness properties are not known in advance. There must be appropriate adaptive strategies developed for realizing adaptive methods. In [48] methods for reducing the number of Jacobian entries have been introduced. Therein, a strategy is used that eliminates Jacobian entries considered to be small by means of a sparsing criterion. This criterion is dynamically adapted to ensure stability and to avoid step-size restriction. In combination with the ROW-type schemes introduced within this thesis such strategies can be used to realize an automatic switching of methods applied with respect to given stiffness properties of a given DAE system.

The property of realizing different solution strategies by applying different approximations to given Jacobian entries holds the potential for implementing type-switching schemes that are able to adapt their solution behavior with respect to given stiffness properties. For example, regarding the simulation of large network structures using a monolithic approach, it is possible to realize different integration strategies for different network regions. Moreover, automatic adaption with respect to given dynamical or latent behavior within single regions is possible. However, the effectiveness of combining explicit and linearly implicit strategies within one single approach for numerical integration is limited by aspects of stability. A significant problem with the ROW-type methods introduced is given by the fact that they allow to apply several extends of explicit and implicit integration simultaneously. However, the approach is realized just by one single step-size that is used during the integration process. Naturally, explicit strategies require for smaller step-sizes due to stability aspects. Hence, the efficiency of possibly included implicit strategies that could work with larger step-sizes cannot be exploited because they are limited by the small step-size of the explicit strategies considered.

For that reason, a combination with multirate strategies seems reasonable. Multirate schemes are based on regarding one single integration strategy. But they allow for using different step-sizes with respect to given active and latent components, i.e. active elements are solved using small step-sizes while latent elements are solved using large step-sizes. Multirate methods have been considered in several works that focus on the numerical integration of electrical networks especially [19, 2, 21]. An extension to W methods for ODEs that allow for non-exact Jacobian entries is presented in [3]. An approach for DAEs of index one with respect to ROW-type schemes using exact Jacobian entries is introduced by Striebel in [82, 83]. Therein, order conditions required are derived introducing a graphical representation based on rooted trees as well. A combination of multirating with ROW-type schemes for solving DAEs by means of non-exact Jacobian matrices have not been considered in literature so far. Hence, corresponding combinations of multirate with the schemes introduced are a promising field of further research.

In order to exploit the possibilities of saving computational efforts by means of non-exact Jacobian entries more effectively, it would be helpful to consider schemes that allow for additional approximations with respect to Jacobian entries g_z. In this context, regarding time-lagged Jacobian information by means of $B_z = g_z + \mathcal{O}(h)$ is the only promising approximation that can be considered. The order conditions for realizing corresponding schemes have been derived in the present PhD phase already. They can be determined by using an enhanced theory based on graphical representation via rooted trees as well. Note that the resulting order conditions correspond to a direct extension of the order conditions presented in this thesis. Their correctness was confirmed by means of appropriate tests which analyze resulting orders of convergence. However, the theory for describing the derivation of these order conditions in detail proved to be too extensive for including it in the thesis given.

Another problem that needs to be addressed are given concepts for analyzing stability aspects. So far, there seems to be a shortage regarding corresponding approaches with respect to DAE problems and linearly implicit strategies that are based on non-exact Jacobian expressions. In general, stability analysis for corresponding methods is reduced to familiar ODE test equations assuming exact Jacobian entries [24]. Examples for alternative ODE test equations when using schemes that are characterized by non-exact Jacobian matrices can be found in [28, 18]. Also, there are recent approaches with respect to ROW-type schemes that use non-exact Jacobian information in the context of approximate-matrix factorization for ODEs [15, 17]. However, the strategies introduced are quite specific and therefore seem to be less applicable to DAE systems at the moment.

Finally, it should be mentioned that there is some potential of further research regarding the derivation of order conditions to ensure B_{PR}-convergence with respect to ROW-type schemes that are based on using non-exact Jacobian expressions. In this thesis, approaches introduced by Rang [59, 60] have been considered to improve characteristics of the schemes developed. However, corresponding order conditions are generally restricted to schemes based on applying exact Jacobian expressions. In [42] there are hints given that the theory to ensure B_{PR}-consistency and B_{PR}-convergence must be adapted when considering schemes that use non-exact Jacobian expressions. Hence, it should be checked if the approach introduced by Rang in [59, 60] can be further enhanced with respect to the ROW-type schemes that are characterized by arbitrary Jacobian approximations.

References

[1] U. M. Ascher, L. R. Petzold: *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, SIAM, Philadelphia (1998)

[2] A. Bartel, M. Günther, A. Kværnø: *Multirate Methods in Electrical Circuit Simulation*, in: A. M. Anile, V. Capasso. A. Greco (Eds.), *Progress in Industrial Mathematics at ECMI 2000*, Mathematics in Industry, 1, Springer, 2002

[3] A. Bartel, M. Günther: *A Multirate W-Method for Electrical Networks in State-Space Formulation*, in: *J. Comput. Appl. Math.*, 147, 411 - 425 (2002)

[4] S. Beck, S. González-Pinto, S. Pérez-Rodríguez, R. Weiner: *A Comparison of AMF- and Krylov-Methods in MATLAB for Large Stiff ODE Systems*, in: *J. Comput. Appl. Math.*, 262, 292 - 303 (2014)

[5] K. E. Brenan, S. L. Campbell, L. R. Petzold: *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, Classics in Applied Mathematics, Vol. 14, SIAM, Philadelphia (1996)

[6] J. C. Butcher: *Coefficients for the Study of Runge-Kutta Integration Processes*, in: *J. Aust. Math. Soc.*, 3, 185 - 201 (1963)

[7] J. C. Butcher: *An Algebraic Theory of Integration Methods*, in: *Math. Comp.*, 26, 79 - 106 (1972)

[8] J.C. Butcher: *Trees and Numerical Methods for Ordinary Differential Equations*, in: *Numer. Algor.*, 53, 153 - 170 (2010)

[9] T. F. Coleman, B.S. Garbow, J. J. Moré: *Software for Estimating Sparse Jacobian Matrices*, in: *AMS Trans. Math. Software*, 11, 329 - 345 (1984)

[10] P. Deuflhard, E. Hairer, J. Zugck: *One-step and Extrapolation Methods for Differential-Algebraic Systems*, in: *Numer. Math.*, 51, 501 - 516 (1987)

[11] R. Frank, J. Schneid, C. W. Ueberhuber: *The Concept of B-convergence*, in: *SIAM J. Numer. Anal.*, 18, 753 - 780 (1981)

[12] C. W. Gear: *Simultaneous Numerical Solution of Differential-Algebraic Equations*, in: *IEEE Transactions on Circuit Theory*, 18, 89 - 95 (1971)

[13] C. W. Gear: *ODE Methods for the Solution of Differential/Algebraic Systems*, in: *SIAM J. Numer. Anal.*, 21, 716 - 728 (1984)

[14] S. González-Pinto, D. Hernández-Abreu, S. Pérez-Rodríguez: *Rosenbrock-Type Methods with Inexact AMF for the Time Integration of Advection-Diffusion-Reaction PDEs*, in: *J. Comput. Appl. Math.*, 262, 304 - 321 (2014)

[15] S. González-Pinto, D. Hernández-Abreu, S. Pérez-Rodríguez, R. Weiner: *A Familiy of Three-Stage Third Order AMF-W-Methods for the Time Integration of Advection Diffusion Reaction PDEs*, in: *Appl. Math. Comp.*, 274, 565 - 584 (2016)

[16] S. González-Pinto, D. Hernández-Abreu, S. Pérez-Rodríguez: *W-Methods to Stabilize Standard Explicit Runge-Kutta Methods in the Time Integration of Advection-Diffusion-Reaction PDEs*, in: *J. Comput. Appl. Math.*, 316, 143 - 160 (2017)

[17] S. González-Pinto, E. Hairer, D. Hernández-Abreu, S. Pérez-Rodríguez: *PDE-W-Methods for Parabolic Problems with Mixed Derivatives*, in: *Numer. Algor.*, 78, 957 - 981 (2018)

[18] S. González-Pinto, E. Hairer, D. Hernández-Abreu, S. Pérez-Rodríguez: *AMF-Type W-Methods for Parabolic Problems with Mixed Derivatives*, in: *SIAM J. Sci. Comput.*, 40, A2905 - A2929 (2018)

[19] M. Günther, P. Rentrop: *Multirate ROW Methods and Latency of Electrical Circuits*, in: *Appl. Numer. Math.*, 13, 83 - 102 (1993)

[20] M. Günther: *Numerical Solution of Differential-Algebraic Equations in Electric Circuit Simulation*, in: H. Neunzert (Ed.), *Progress in Industrial Mathematics at ECMI 94*, Vieweg+Teubner, Wiesbaden (1996)

[21] M. Günther, A. Kværnø, P. Rentrop: *Multirate Partitioned Runge-Kutta Methods*, in: *BIT*, 41, 504 - 5014 (2001)

[22] E. Hairer, G. Bader, Ch. Lubich: *On the Stability of Semi-Implicit Methods for Ordinary Differential Equations*, in: *BIT*, 22, 211 - 232 (1982)

[23] E. Hairer, S.P. Nørset, G. Wanner: *Solving Ordinary Differential Equations I: Nonstiff Problems*, 2nd edition, Springer, Berlin, Heidelberg, New York (2000)

[24] E. Hairer, G. Wanner: *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, 2nd edition, Springer, Berlin, Heidelberg, New York (2002)

[25] A.-W. Hamkar, S. Hartmann, J. Rang: *A Stiffly Accurate Rosenbrock-Type Method of Order 2 Applied to FE-Analyses in Finite Strain Viscoelasticity*, in: *Appl. Numer. Math.*, 62, 1837 - 1848 (2012)

[26] M. Hilden, G. Steinebach: *ENO-Discretizations in MOL-Applications: Some Examples in River Hydraulics*, in: *Appl. Numer. Math.*, 28, 293 - 308 (1998)

[27] P. J. van der Houwen, B. P. Sommeijer: *Approximate Factorization for Time-Dependent Partial Differential Equations*, in: *J. Comput. Appl. Math*, 128, 447 - 466 (2001)

[28] W. Hundsdorfer, J. G. Verwer: *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, Springer, Berlin, Heidelberg, New York (2003)

[29] T. Jax, G. Steinebach: *ROW Methods Adapted to Network Simulation for Fluid Flow*, in: G. Russo et al. (Eds.), *Progress in Industrial Mathematics at ECMI 2014*, Mathematics in Industry, 22, Springer, 2016

[30] T. Jax, G. Steinebach: *Generalized ROW-Type Methods for Solving Semi-Explicit DAEs of Index-1*, in: *J. Comput. Appl. Math.*, 316, 213 - 228 (2017)

[31] T. Jax, G. Steinebach: *Generalized ROW-Type Methods for Simulating Water Supply Networks*, in: P. Quintela, P. Barral, D. Gómez, F. J. Pena, et al. (Eds.), *Progress in Industrial Mathematics at ECMI 2016*, Mathematics in Industry, Vol. 26, Springer, 2017

[32] T. Jax: *Network Aggregation*, in: A. Pirsing, A. Morsi, et al. (Eds.), *Decision Support Systems for Water Supply Systems*, EMS Series in Industrial and Applied Mathematics, European Mathematical Society Publishing House, Zürich, 2020 (submitted)

[33] P. Kaps, P. Rentrop: *Generalized Runge-Kutta Methods of Order Four with Stepsize Control for Stiff Ordinary Differential Equations*, in: *Numer. Math.*, 33, 55 - 68 (1979)

[34] P. Kaps, A. Ostermann: *Rosenbrock Methods using few LU-Decompositions*, in: *IMA J. Numer. Anal.*, 9, 15 - 27 (1989)

[35] J. Lang, J. Verwer: *ROS3P - An Accurate Third-Order Rosenbrock Solver Designed for Parabolic Problems*, in: *BIT*, 41, 730 - 737 (2001)

[36] J. Lang, D. Teleaga: *Towards a Fully Space-Time Adaptive FEM for Magnetoquasistatics*, in: *IEEE Trans. Magn.*, 44, 1238 - 1241 (2008)

[37] J. Lang, J. G. Verwer: *W-Methods in Optimal Control*, in: *Numer. Math.*, 124, 337 - 360 (2013)

[38] K. Levenberg: *A Method for the Solution of Certain Problems in Least-Squares*, in: *Quarterly Applied Mathematics 2*, 164 - 168 (1944)

[39] R. J. LeVeque, W. Jinghua: *A Linear Hyperbolic System with Stiff Source Terms*, in: A. Donato, F. Oliveri (Eds.), *Nonlinear Hyperbolic Problems: Theoretical, Applied, and Computational Aspects*, Notes on Numerical Fluid Mechanics (NNFM), 43, Vieweg+Teubner, Braunschweig (1993)

[40] P. Lötstedt: *Numerical Solution of Nonlinear Differential Equations with Algebraic Constraints I: Convergence Results for Backward Differentiation Formulas*, in: *Math. Comput.*, 46, 491 - 516 (1986)

[41] Ch. Lubich, M. Roche: *Rosenbrock Methods for Differential-Algebraic Systems with Solution-Dependent Singular Matrix Multiplying the Derivative*, in: *Computing*, 43, 325 - 342 (1990)

[42] Ch. Lubich, A. Ostermann: *Linearly Implicit Time Discretization of Non-Linear Parabolic Equations*, in: *IMA J. Numer. Anal.*, 15, 555 - 583 (1995)

[43] D. Marquardt: *An Algorithm for Least-Squares Estimation of Nonlinear Parameters*, in: *SIAM J. Appl. Math.*, 11, 431 - 441 (1963)

[44] R. I. McLachlan, K. Modin, H. Munthe-Kaas, O. Verdier: *Butcher Series: A Story of Rooted Trees and Numerical Methods for Evolution Equations*, in: *Asia Pacific Mathematics Newsletter*, 7, 1 - 11 (2017)

[45] R. H. Merson: *An Operational Method for the Study of Integration Processes*, in: *Proceedings of Conference on Data Processing and Automatic Computing Machines*, Vol. 1, 1 - 25, Weapons Research Establishment, Salisbury, South Australia (1957)

[46] J. J. Moré: *The Levenberg-Marquardt Algorithm: Implementation and Theory*, in: *Numerical Analysis*, G. A. Watson (Ed.), *Lecture Notes in Mathematics 630*, 105 - 116, Springer (1977)

[47] P. Novati: *Some Secant Approximations for Rosenbrock W-Methods*, in: *Appl. Numer. Math.*, 58, 195 - 211 (2008)

[48] U. Nowak: *Dynamic Sparsing in Stiff Extrapolation Methods*, Preprint SC 92-21, ZIB Konrad-Zuse-Zentrum für Informationstechnik Berlin, Berlin (1992)

[49] A. Ostermann, M. Roche: *Rosenbrock Methods for Partial Differential Equations and Fractional Orders of Convergence*, in: *SIAM J. Numer. Anal.*, 30, 1084 - 1098 (1993)

[50] L. Petzold: *Differential/Algebraic Equations are not ODE's* in: *SIAM J. Sci. Stat. Comput.*, 3, 367 - 384 (1982)

[51] L. Petzold: *A Description of DASSL: A Differential/Algebraic System Solver*, *Sandia Report*, SAND82-8637, Sandia National Laboratories (1982)

[52] L. Petzold, P. Lötstedt: *Numerical Solution of Nonlinear Differential Equations with Algebraic Constraints II: Practical Implications*, in: *SIAM J. Sci. Stat. Comput.*, 7, 720 - 733 (1986)

[53] A. Pirsing, A. Morsi, et al. (Eds.): *Decision Support Systems for Water Supply Systems*, EMS Series in Industrial and Applied Mathematics, European Mathematical Society Publishing House, Zürich, 2020 (submitted)

[54] A. Prothero, A. Robinson: *On the Stability and Accuracy of One-Step Methods for Solving Stiff Systems of Ordinary Differential Equations*, in: *Math. Comput.*, 28, 145 - 162 (1974)

[55] A. Rahunanthan, D. Stanescu: *High-Order W-Methods*, in: *J. Comput. Appl. Math.*, 233, 1798 - 1811 (2010)

[56] J. Rang, L. Angermann: *New Rosenbrock W-Methods of Order 3 for Partial Differential Algebraic Equations of Index 1*, in: *BIT Numerical Mathematics*, 45, 761 -787 (2005)

[57] J. Rang: *An Analysis of the Prothero-Robinson Example for Constructing New DIRK and ROW Methods*, in: *J. Comput. Appl. Math.*, 262, 105 - 114 (2014)

[58] J. Rang: *The Prothero and Robinson Example: Convergence Studies for Runge-Kutta and Rosenbrock-Wanner Methods*, in: *Informatik-Bericht*, No. 2014 - 05, TU Braunschweig (2014)

[59] J. Rang: *Improved Traditional Rosenbrock-Wanner Methods for Stiff ODEs and DAEs*, in: *J. Comput. Appl. Math.*, 286, 128 - 144 (2015)

[60] J. Rang: *The Prothero and Robinson Example: Convergence Studies for Runge-Kutta and Rosenbrock-Wanner Methods*, in: *Appl. Numer. Math.*, 108, 37 - 56 (2016)

[61] P. Rentrop: *Partitioned Runge-Kutta Methods with Stiffness Detection and Stepsize Control*, in: *Numer. Math.*, 47, 545 - 564 (1985)

[62] P. Rentrop, M. Roche, G. Steinebach: *The Application of Rosenbrock-Wanner Type Methods with Stepsize Control in Differential-Algebraic Equations*, in: *Numer. Math.*, 55, 545 - 563 (1989)

[63] P. Rentrop, G. Steinebach: *Model and Numerical Techniques for the Alarm System of River Rhine*, in: *Surv. Math. Ind.*, 6, 245 - 265 (1997)

[64] M. Roche: *Rosenbrock Methods for Differential Algebraic Equations*, in: *Numer. Math.*, 52, 45 - 63 (1988)

[65] H. H. Rosenbrock: *Some General Implicit Processes for the Numerical Solution of Differential Equations*, in: *Comp. J.* , 5, 329 - 331 (1963)

[66] D. E. Salane, *Adaptive Routines for Forming Jacobians Numerically*, in: *Sandia Report*, SAND86-1319, Sandia National Laboratories (1986)

[67] B. A. Schmitt, R. Weiner: *Matrix-Free W-Methods Using a Multiple Arnoldi Iteration*, in: *Appl. Numer. Math.*, 18, 307 - 320 (1995)

[68] S. Scholz: *Order Barriers for the B-Convergence of ROW Methods*, in: *Computing*, 41, 219 - 235 (1989)

[69] L. F. Shamoine: *Implementation of Rosenbrock Methods*, in: *ACM Transactions on Mathematical Software*, 8, 93 - 113 (1982)

[70] B. Simeon: *On the History of Differential-Algebraic Equations*, in: A. Ilchmann, T. Reis (Eds.): *Surveys in Differential-Algebraic Equations*, Vol. 4, pp. 1 - 39, Differential-Algebraic Equations Forum, Springer (2017)

[71] T. Steihaug, A. Wolfbrandt: *An Attempt to Avoid Exact Jacobian and Nonlinear Equations in the Numerical Solution of Stiff Differential Equations*, in: *Math. Comp.*, 33, 521 - 534 (1979)

[72] G. Steinebach: *Order-Reduction of ROW-Methods for DAEs and Method of Lines Applications*, Preprint 1741, Technische Universiẗt Darmstadt, Darmstadt (1995)

[73] G. Steinebach, P. Rentrop: *An Adaptive Method of Lines Approach for Modelling Flow and Transport in Rivers*, in: A. Vande Wouver, Ph. Sauces, W. E. Schiesser (Eds.), *Adaptive Method of Lines*, pp. 181 - 205, Chapman & Hall/CRC, Boca Raton, London New York, Washington D.C. (2001)

[74] G. Steinebach, S. Rademacher, P. Rentrop, M. Schulz: *Mechanisms of Coupling in River Flow Simulation Systems*, in: *J. Comput. Appl. Math.*, 168, 459 - 470 (2004)

[75] G. Steinebach, R. Rosen, A. Sohr: *Modeling and Numerical Simulation of Pipe Flow Problems in Water Supply Systems*, in: A. Martin, K. Klamroth, J. Lang, et al. (Eds.), *Mathematical Optimization of Water Networks*, International Series of Numerical Mathematics, 162, pp. 3 - 15, Birkhäuser, Basel (2012)

[76] G. Steinebach, O. Kolb: *Hydraulic Modeling and Energy View*, in: A. Pirsing, A. Morsi, et al. (Eds.), *Decision Support Systems for Water Supply Systems*, EMS Series in Industrial and Applied Mathematics, European Mathematical Society Publishing House, Zürich, 2020 (submitted)

[77] K. Strehmel, R. Weiner, I. Dannehl: *On Error Behaviour of Partitioned Linearly Implicit Runge-Kutta Methods for Stiff and Differential Algebraic Systems*, in: *BIT*, 30, 358 - 375 (1990)

[78] K. Strehmel, R. Weiner: *Linearly-Implicit Runge-Kutta Methods for Singularly Perturbed Problems and Index-1 DAE's*, in: *Numerical Treatment of Differential Equations*,*Teubner-Texte zur Mathematik*, 121, Vieweg+Teubner (1991)

[79] K. Strehmel, R. Weiner, M. Büttner: *Order Results for Rosenbrock Type Methods on Classes of Stiff Equations*, in: *Numer. Math.*, 59, 723 - 737 (1991)

[80] K. Strehmel, R. Weiner: *Linear-implizite Runge-Kutta Methoden und ihre Anwendung*, *Teubner-Texte zur Mathematik*, 127, Vieweg+Teubner (1992)

[81] K. Strehmel, R. Weiner, H. Podhaisky: *Numerik gewöhnlicher Differentialgleichungen: Nichtsteife, steife und differential-algebraische Gleichungen*, 2nd edition, Vieweg+Teubner, Wiesbaden (2012)

[82] M. Striebel, M. Günther: *A Charge Oriented Mixed Multirate Method for a Special Class of Index-1 Network Equations in Chip Design*, in: *Appl. Numer. Math.*, 53, 489 - 507 (2005)

[83] M. Striebel, A. Bartel, M. Guünther: *A Multirate ROW-Scheme for Index-1 Network Equations*, in *Appl. Numer. Math.*, 59, 800 - 814 (2009)

[84] P. Tranquilli, A. Sandu: *Rosenbrock-Krylov Methods for Large Systems of Differential Equations*, in: *SIAM J. Sci. Comput.*, 36, A1313 - A1338 (2014)

[85] P. Tranquilli, R. Glandon, A. Sandu: *Adaptive Krylov-Type Time Integration Methods*, Preprint submitted to Elsevier, arXiv:1910.02514v1 [math.NA] (2019)

[86] J. G. Verwer, S. Scholz, J. G. Blom, M. Louter-Nool: *A Class of Runge-Kutta-Rosenbrock Methods for Solving Stiff Differential Equations*, in: *ZAMM*, 63, 13 - 20 (1983)

[87] J. G. Verwer, S. Scholz: *Rosenbrock Methods with Time-Lagged Jacobian Matrices*, in: *Beitäge zur Numerischen Mathematik*, 11, 173 - 183 (1983)

[88] J. G. Verwer, E. J. Spee, J. G. Bloom, W. Hundsdorfer: *A Second-Order Rosenbrock Method Applied to Photochemical Dispersion Problems*, in: *SIAM J. Sci. Comput.*, 20, 1456 - 1480 (1999)

[89] G. Wanner: *On the Integration of Stiff Differential Equations*, in: J. Descloux, J. Marti (Eds.), *Numerical Analysis*, International Series of Numerical Mathematics, 37, pp. 209 - 226, Birkhäuser, Basel (1977)

[90] R. Weiner, B. A. Schmitt, H. Podhaisky: *ROWMAP - A ROW-Code with Krylov Techniques for Large Stiff ODEs*, in: *Appl. Numer. Math.*, 25, 303 - 319 (1997)

A Appedix: Conditions up to Order 3

Table A.1: All Conditions for Order 3 (1 of 5).

Differential Part

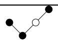
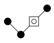
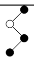
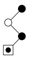
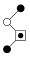
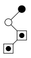
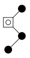
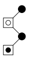
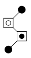
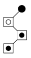
$\rho(t)$	Req.	No.	DAT_y	$ADAT_y^{DA}$	Differential	Condition
1	X	1D1.0			f	$\sum b_i = 1$
2	X	2D1.0			$f_y f$	$\sum b_i \alpha_{ij} = 1/2$
	X	2D1.1			$A_y f$	$\sum b_i \gamma_{ij} = 0$
	X	2D2.0			$f_z(-g_z)^{-1} g_y f$	$\sum b_i \alpha_{ij} \omega_{jk} \alpha_{kl} = 1/2$
		2D2.1			$A_z(-g_z)^{-1} g_y f$	$\sum b_i \gamma_{ij} \omega_{jk} \alpha_{kl} = 0$
		2D2.2			$f_z(-g_z)^{-1} B_y f$	$\sum b_i \alpha_{ij} \omega_{jk} \gamma_{kl} = 0$
		2D2.3			$A_z(-g_z)^{-1} B_y f$	$\sum b_i \gamma_{ij} \omega_{jk} \gamma_{kl} = 0$
3	X	3D1.0			$f_{yy}(f,f)$	$\sum b_i \alpha_{ij} \alpha_{ik} = 1/3$
	X	3D2.0			$f_y f_y f$	$\sum b_i \alpha_{ij} \alpha_{jk} = 1/6$
	X	3D2.1			$A_y f_y f$	$\sum b_i \gamma_{ij} \alpha_{jk} = 0$
	X	3D2.2			$f_y A_y f$	$\sum b_i \alpha_{ij} \gamma_{jk} = 0$
	X	3D2.3			$A_y A_y f$	$\sum b_i \gamma_{ij} \gamma_{jk} = 0$
	X	3D3.0			$f_{yz}(f,(-g_z)^{-1} g_y f)$	$\sum b_i \alpha_{ij} \alpha_{ik} \omega_{kl} \alpha_{lm} = 1/3$
		3D3.1			$f_{yz}(f,(-g_z)^{-1} B_y f)$	$\sum b_i \alpha_{ij} \alpha_{ik} \omega_{kl} \gamma_{lm} = 0$
	X	3D4.0			$f_y f_z(-g_z)^{-1} g_y f$	$\sum b_i \alpha_{ij} \alpha_{jk} \omega_{kl} \alpha_{lm} = 1/6$
	X	3D4.1			$A_y f_z(-g_z)^{-1} g_y f$	$\sum b_i \gamma_{ij} \alpha_{jk} \omega_{kl} \alpha_{lm} = 0$
		3D4.2			$f_y A_z(-g_z)^{-1} g_y f$	$\sum b_i \alpha_{ij} \gamma_{jk} \omega_{kl} \alpha_{lm} = 0$
		3D4.3			$A_y A_z(-g_z)^{-1} g_y f$	$\sum b_i \gamma_{ij} \gamma_{jk} \omega_{kl} \alpha_{lm} = 0$
		3D4.4			$f_y f_z(-g_z)^{-1} B_y f$	$\sum b_i \alpha_{ij} \alpha_{jk} \omega_{kl} \gamma_{lm} = 0$
		3D4.5			$A_y f_z(-g_z)^{-1} B_y f$	$\sum b_i \gamma_{ij} \alpha_{jk} \omega_{kl} \gamma_{lm} = 0$
		3D4.6			$f_y A_z(-g_z)^{-1} B_y f$	$\sum b_i \alpha_{ij} \gamma_{jk} \omega_{kl} \gamma_{lm} = 0$
		3D4.7			$A_y A_z(-g_z)^{-1} B_y f$	$\sum b_i \gamma_{ij} \gamma_{jk} \omega_{kl} \gamma_{lm} = 0$

Table A.1: All Conditions of Order 3 (2 of 5).

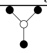
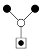
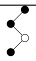
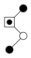
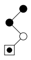
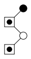
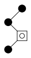
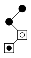
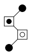
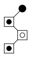
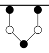
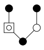
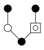
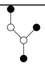
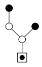
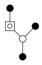
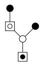
				Differential Part		
$\rho(t)$	Req.	No.	DAT_y	$ADAT_y^{DA}$	Differential	Condition
3	X	3D5.0			$f_z(-g_z)^{-1}g_{yy}(f,f)$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\alpha_{km} = 1/3$
		3D5.1			$A_z(-g_z)^{-1}g_{yy}(f,f)$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\alpha_{km} = 0$
	X	3D6.0			$f_z(-g_z)^{-1}g_yf_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\alpha_{lm} = 1/6$
	X	3D6.1			$A_z(-g_z)^{-1}g_yf_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\alpha_{lm} = 0$
		3D6.2			$f_z(-g_z)^{-1}g_yA_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\gamma_{lm} = 0$
		3D6.3			$A_z(-g_z)^{-1}g_yA_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\gamma_{lm} = 0$
		3D6.4			$f_z(-g_z)^{-1}B_yf_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\gamma_{kl}\alpha_{lm} = 0$
		3D6.5			$A_z(-g_z)^{-1}B_yf_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\gamma_{kl}\alpha_{lm} = 0$
		3D6.6			$f_z(-g_z)^{-1}B_yA_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\gamma_{kl}\gamma_{lm} = 0$
		3D6.7			$A_z(-g_z)^{-1}B_yA_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\gamma_{kl}\gamma_{lm} = 0$
	X	3D7.0			$f_{zz}((-g_z)^{-1}g_yf,(-g_z)^{-1}g_yf)$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\alpha_{im}\omega_{mn}\alpha_{np} = 1/3$
		3D7.1			$f_{zz}((-g_z)^{-1}B_yf,(-g_z)^{-1}g_yf)$	$\sum b_i\alpha_{ij}\omega_{jk}\gamma_{kl}\alpha_{im}\omega_{mn}\alpha_{np} = 0$
		3D7.2			$f_{zz}((-g_z)^{-1}g_yf,(-g_z)^{-1}B_yf)$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\alpha_{im}\omega_{mn}\gamma_{np} = 0$
		3D7.3			$f_{zz}((-g_z)^{-1}B_yf,(-g_z)^{-1}B_yf)$	$\sum b_i\alpha_{ij}\omega_{jk}\gamma_{kl}\alpha_{im}\omega_{mn}\gamma_{np} = 0$
	X	3D8.0			$f_z(-g_z)^{-1}g_{zy}((-g_z)^{-1}g_yf,f)$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\alpha_{mn}\alpha_{kp} = 1/3$
		3D8.1			$A_z(-g_z)^{-1}g_{zy}((-g_z)^{-1}g_yf,f)$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\alpha_{mn}\alpha_{kp} = 0$
		3D8.2			$f_z(-g_z)^{-1}g_{zy}((-g_z)^{-1}B_yf,f)$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\gamma_{mn}\alpha_{kp} = 0$
		3D8.3			$A_z(-g_z)^{-1}g_{zy}((-g_z)^{-1}B_yf,f)$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\gamma_{mn}\alpha_{kp} = 0$

Table A.1: All Conditions of Order 3 (3 of 5).

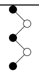
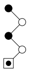
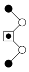
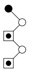
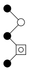
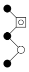
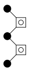
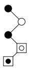
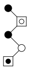
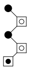
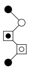
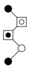
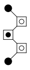
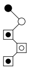
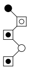
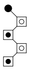
					Differential Part	
$\rho(t)$	Req.	No.	DAT_y	$ADAT_y^{DA}$	Differential	Condition
3	X	3D9.0			$f_z(-g_z)^{-1}g_yf_z(-g_z)^{-1}g_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\alpha_{lm}\omega_{mn}\alpha_{np} = 1/6$
		3D9.1			$A_z(-g_z)^{-1}g_yf_z(-g_z)^{-1}g_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\alpha_{lm}\omega_{mn}\alpha_{np} = 0$
		3D9.2			$f_z(-g_z)^{-1}g_yA_z(-g_z)^{-1}g_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\gamma_{lm}\omega_{mn}\alpha_{np} = 0$
		3D9.3			$A_z(-g_z)^{-1}g_yA_z(-g_z)^{-1}g_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\gamma_{lm}\omega_{mn}\alpha_{np} = 0$
		3D9.4			$f_z(-g_z)^{-1}B_yf_z(-g_z)^{-1}g_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\gamma_{kl}\alpha_{lm}\omega_{mn}\alpha_{np} = 0$
		3D9.5			$f_z(-g_z)^{-1}g_yf_z(-g_z)^{-1}B_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\alpha_{lm}\omega_{mn}\gamma_{np} = 0$
		3D9.6			$f_z(-g_z)^{-1}B_yf_z(-g_z)^{-1}B_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\gamma_{kl}\alpha_{lm}\omega_{mn}\gamma_{np} = 0$
		3D9.7			$A_z(-g_z)^{-1}B_yf_z(-g_z)^{-1}g_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\gamma_{kl}\alpha_{lm}\omega_{mn}\alpha_{np} = 0$
		3D9.8			$A_z(-g_z)^{-1}g_yf_z(-g_z)^{-1}B_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\alpha_{lm}\omega_{mn}\gamma_{np} = 0$
		3D9.9			$A_z(-g_z)^{-1}B_yf_z(-g_z)^{-1}B_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\gamma_{kl}\alpha_{lm}\omega_{mn}\gamma_{np} = 0$
		3D9.10			$f_z(-g_z)^{-1}B_yA_z(-g_z)^{-1}g_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\gamma_{kl}\gamma_{lm}\omega_{mn}\alpha_{np} = 0$
		3D9.11			$f_z(-g_z)^{-1}g_yA_z(-g_z)^{-1}B_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\gamma_{lm}\omega_{mn}\gamma_{np} = 0$
		3D9.12			$f_z(-g_z)^{-1}B_yA_z(-g_z)^{-1}B_yf$	$\sum b_i\alpha_{ij}\omega_{jk}\gamma_{kl}\gamma_{lm}\omega_{mn}\gamma_{np} = 0$
		3D9.13			$A_z(-g_z)^{-1}B_yA_z(-g_z)^{-1}g_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\gamma_{kl}\gamma_{lm}\omega_{mn}\alpha_{np} = 0$
		3D9.14			$A_z(-g_z)^{-1}g_yA_z(-g_z)^{-1}B_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\gamma_{lm}\omega_{mn}\gamma_{np} = 0$
		3D9.15			$A_z(-g_z)^{-1}B_yA_z(-g_z)^{-1}B_yf$	$\sum b_i\gamma_{ij}\omega_{jk}\gamma_{kl}\gamma_{lm}\omega_{mn}\gamma_{np} = 0$

Table A.1: All Conditions of Order 3 (4 of 5).

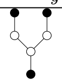
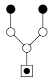
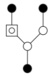
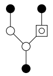
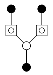
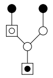
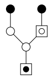
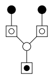
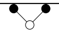
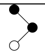
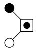
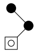
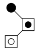
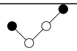
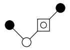
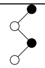
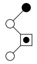
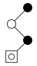
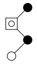
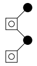
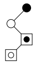
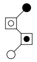
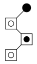
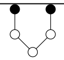
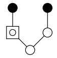
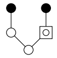
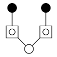
					Differential Part	
$\rho(t)$	Req.	No.	DAT_y	$ADAT_y^{DA}$	Differential	Condition
3	X	3D10.0			$f_z(-g_z)^{-1}g_{zz}((-g_z)^{-1}g_yf,(-g_z)^{-1}g_yf)$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\alpha_{mn}\alpha_{kp}\omega_{pq}\alpha_{qr}=1/3$
		3D10.1			$A_z(-g_z)^{-1}g_{zz}((-g_z)^{-1}g_yf,(-g_z)^{-1}g_yf)$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\alpha_{mn}\alpha_{kp}\omega_{pq}\alpha_{qr}=0$
		3D10.2			$f_z(-g_z)^{-1}g_{zz}((-g_z)^{-1}B_yf,(-g_z)^{-1}g_yf)$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\gamma_{mn}\alpha_{kp}\omega_{pq}\alpha_{qr}=0$
		3D10.3			$f_z(-g_z)^{-1}g_{zz}((-g_z)^{-1}g_yf,(-g_z)^{-1}B_yf)$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\alpha_{mn}\alpha_{kp}\omega_{pq}\gamma_{qr}=0$
		3D10.4			$f_z(-g_z)^{-1}g_{zz}((-g_z)^{-1}B_yf,(-g_z)^{-1}B_yf)$	$\sum b_i\alpha_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\gamma_{mn}\alpha_{kp}\omega_{pq}\gamma_{qr}=0$
		3D10.5			$A_z(-g_z)^{-1}g_{zz}((-g_z)^{-1}B_yf,(-g_z)^{-1}g_yf)$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\gamma_{mn}\alpha_{kp}\omega_{pq}\alpha_{qr}=0$
		3D10.6			$A_z(-g_z)^{-1}g_{zz}((-g_z)^{-1}g_yf,(-g_z)^{-1}B_yf)$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\alpha_{mn}\alpha_{kp}\omega_{pq}\gamma_{qr}=0$
		3D10.7			$A_z(-g_z)^{-1}g_{zz}((-g_z)^{-1}B_yf,(-g_z)^{-1}B_yf)$	$\sum b_i\gamma_{ij}\omega_{jk}\alpha_{kl}\omega_{lm}\gamma_{mn}\alpha_{kp}\omega_{pq}\gamma_{qr}=0$

Table A.1: All Conditions of Order 3 (5 of 5).

Algebraic Part

$\rho(t)$	Req.	No.	DAT_y	$ADAT_y^{DA}$	Differential	Condition
1	X	1A1.0			$(-g_z)^{-1}g_yf$	$\sum b_i\omega_{ij}\alpha_{jk} = 1$
		1A1.1			$(-g_z)^{-1}B_yf$	$\sum b_i\omega_{ij}\gamma_{jk} = 0$
2	X	2A1.0			$(-g_z)^{-1}g_{yy}(f,f)$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{jl} = 1$
	X	2A2.0			$(-g_z)^{-1}g_yf_yf$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{kl} = 1/2$
	X	2A2.1			$(-g_z)^{-1}g_yA_yf$	$\sum b_i\omega_{ij}\alpha_{jk}\gamma_{kl} = 0$
		2A2.2			$(-g_z)^{-1}B_yf_yf$	$\sum b_i\omega_{ij}\gamma_{jk}\alpha_{kl} = 0$
		2A2.3			$(-g_z)^{-1}B_yA_yf$	$\sum b_i\omega_{ij}\gamma_{jk}\gamma_{kl} = 0$
	X	2A3.0			$(-g_z)^{-1}g_{yz}(f,(-g_z)^{-1}g_yf)$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{jl}\omega_{lm}\alpha_{mn} = 1$
		2A3.1			$(-g_z)^{-1}g_{yz}(f,(-g_z)^{-1}B_yf)$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{jl}\omega_{lm}\gamma_{mn} = 0$
	X	2A4.0			$(-g_z)^{-1}g_yf_z(-g_z)^{-1}g_yf$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{kl}\omega_{lm}\alpha_{mn} = 1/2$
		2A4.1			$(-g_z)^{-1}g_yA_z(-g_z)^{-1}g_yf$	$\sum b_i\omega_{ij}\alpha_{jk}\gamma_{kl}\omega_{lm}\alpha_{mn} = 0$
		2A4.2			$(-g_z)^{-1}B_yf_z(-g_z)^{-1}g_yf$	$\sum b_i\omega_{ij}\gamma_{jk}\alpha_{kl}\omega_{lm}\alpha_{mn} = 0$
		2A4.3			$(-g_z)^{-1}g_yf_z(-g_z)^{-1}B_yf$	$\sum b_i\omega_{ij}\alpha_{jk}\alpha_{kl}\omega_{lm}\gamma_{mn} = 0$
		2A4.4			$(-g_z)^{-1}B_yf_z(-g_z)^{-1}B_yf$	$\sum b_i\omega_{ij}\gamma_{jk}\alpha_{kl}\omega_{lm}\gamma_{mn} = 0$
		2A4.5			$(-g_z)^{-1}B_yA_z(-g_z)^{-1}g_yf$	$\sum b_i\omega_{ij}\gamma_{jk}\gamma_{kl}\omega_{lm}\alpha_{mn} = 0$
		2A4.6			$(-g_z)^{-1}g_yA_z(-g_z)^{-1}B_yf$	$\sum b_i\omega_{ij}\alpha_{jk}\gamma_{kl}\omega_{lm}\gamma_{mn} = 0$
		2A4.7			$(-g_z)^{-1}B_yA_z(-g_z)^{-1}B_yf$	$\sum b_i\omega_{ij}\gamma_{jk}\gamma_{kl}\omega_{lm}\gamma_{mn} = 0$
	X	2A5.0			$(-g_z)^{-1}g_{zz}((-g_z)^{-1}g_yf,(-g_z)^{-1}g_yf)$	$\sum b_i\omega_{ij}\alpha_{jk}\omega_{kl}\alpha_{lm}\alpha_{jn}\omega_{np}\alpha_{pq} = 1$
		2A5.1			$(-g_z)^{-1}g_{zz}((-g_z)^{-1}B_yf,(-g_z)^{-1}g_yf)$	$\sum b_i\omega_{ij}\alpha_{jk}\omega_{kl}\gamma_{lm}\alpha_{jn}\omega_{np}\alpha_{pq} = 0$
		2A5.2			$(-g_z)^{-1}g_{zz}((-g_z)^{-1}g_yf,(-g_z)^{-1}B_yf)$	$\sum b_i\omega_{ij}\alpha_{jk}\omega_{kl}\alpha_{lm}\alpha_{jn}\omega_{np}\gamma_{pq} = 0$
		2A5.3			$(-g_z)^{-1}g_{zz}((-g_z)^{-1}B_yf,(-g_z)^{-1}B_yf)$	$\sum b_i\omega_{ij}\alpha_{jk}\omega_{kl}\gamma_{lm}\alpha_{jn}\omega_{np}\gamma_{pq} = 0$

B Appedix: Convergence Results of Test 2

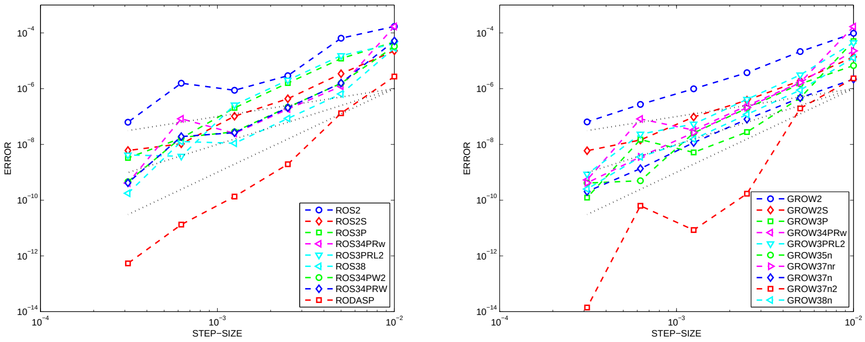


Figure B.1: Step-size versus error for (6.4) using exact Jacobian.

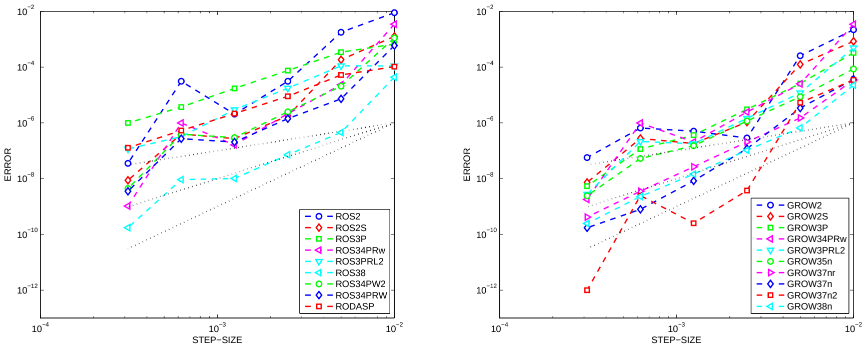


Figure B.2: Step-size versus error for (6.4) using Jacobian updates after 5 steps.

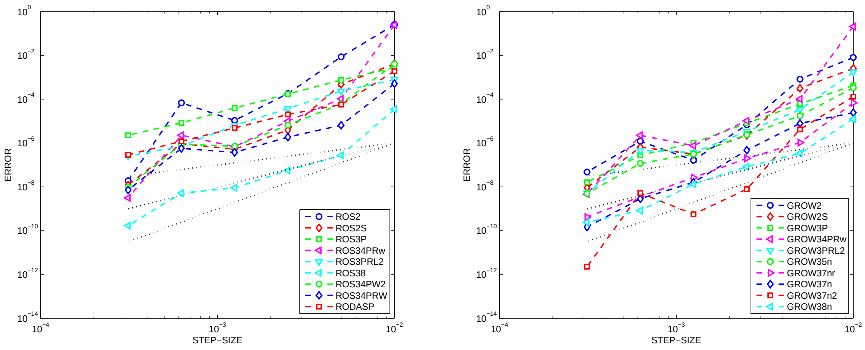


Figure B.3: Step-size versus error for (6.4) using Jacobian updates after 10 steps.

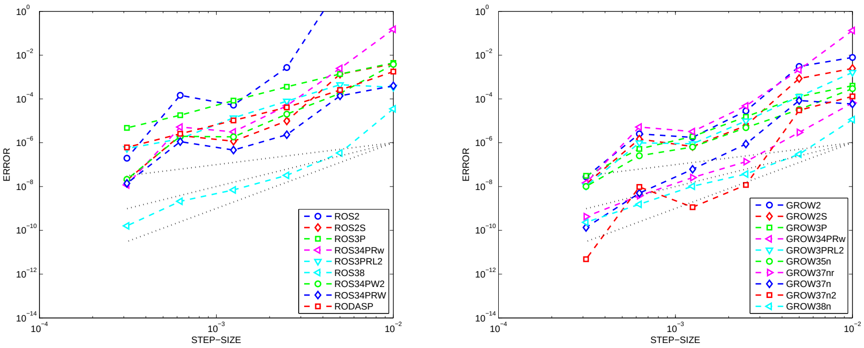


Figure B.4: Step-size versus error for (6.4) using Jacobian updates after 20 steps.

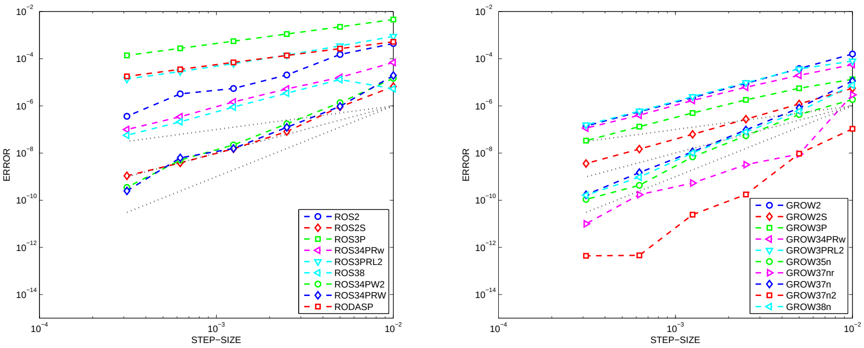


Figure B.5: Step-size versus error for (6.4) after setting Jacobian entries with respect to A_y and A_z to zero.

C Appedix: Convergence Results of Test 3

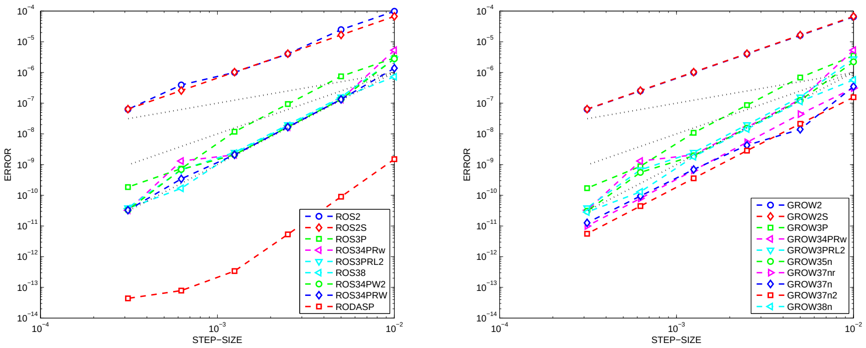


Figure C.1: Step-size versus error for (6.5) using exact Jacobian.

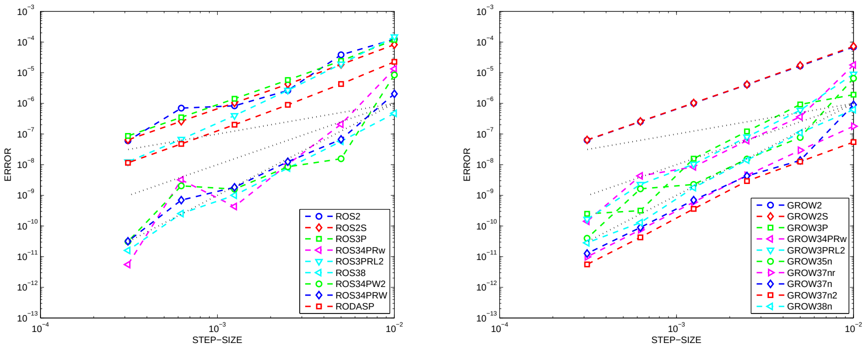


Figure C.2: Step-size versus error for (6.5) using Jacobian updates after 5 steps.

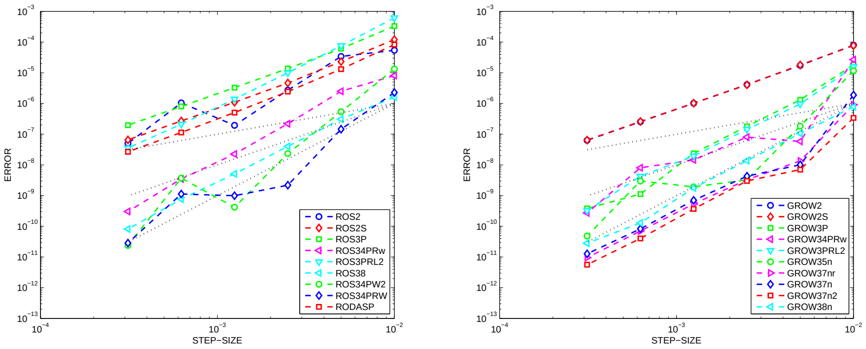


Figure C.3: Step-size versus error for (6.5) using Jacobian updates after 10 steps.

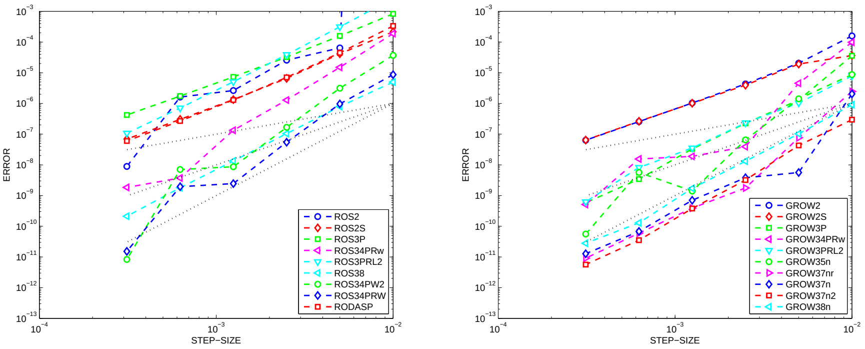


Figure C.4: Step-size versus error for (6.5) using Jacobian updates after 20 steps.

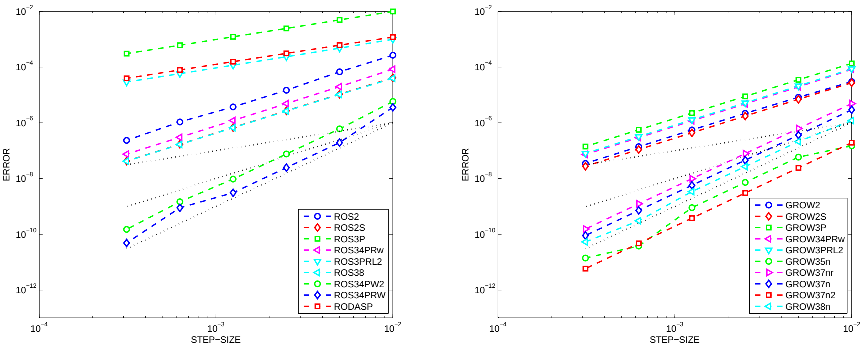


Figure C.5: Step-size versus error for (6.5) after setting Jacobian entries with respect to A_y and A_z to zero.

D Appedix: Some Results Using Non-Exact Entries $\mathbf{B_z}$

Within the thesis it is mentioned that order conditions for non-exact Jacobian entries B_z have also been derived. In this context, the case $B_z = (g_z)_0 + \mathcal{O}(h)$ has been considered in particular. In fact, it proves to be the only reasonable approximation with respect to these Jacobian entries so far. Among the sets of coefficients introduced in Section 5 GROW37n, GROW37n2 and GROW38n are numerically determined schemes that satisfy the conditions to achieve order $p = 3$. Note that there are no additional conditions required to preserve order $p = 2$.

Without describing the resulting order conditions or the theory used to find them in detail, some exemplary results shall be shown below that seem to indicate the correctness of the additional conditions derived. For this purpose, GROW37n and GROW37nr are applied to problem (6.5) that was originally constructed to analyze effects using non-exact Jacobian entries B_z. The test is applied regarding approximations of the form $A_y = 0$, $A_z = 0$ and $B_y = 0$ in combination with $B_z = (g_z)_0$ and $B_z = (g_z)_0 + \mathcal{O}(h)$. In this context, the approximation $B_z = (g_z)_0 + \mathcal{O}(h)$ is realized by regarding an update of corresponding Jacobian entries after ten time-steps.

Findings are depicted in Figure D.1 showing results for $B_z = (g_z)_0$ to the left and $B_z = (g_z)_0 + \mathcal{O}(h)$ to the right. They show that GROW37n which satisfies the additional conditions determined for considering non-exact components B_z is able to preserve order $p = 3$. However, GROW37nr that does not satisfy the additional conditions reduces its order from $p = 3$ regarding exact components B_z to $p = 2$ regarding non-exact components B_z. In this context, note that GROW37nr performed quite well with respect to the tests considered in Section 6. Hence, one can assume that the order reduction shown is a consequence of not satisfying the additional conditions determined for the case $B_z = (g_z)_0 + \mathcal{O}(h)$.

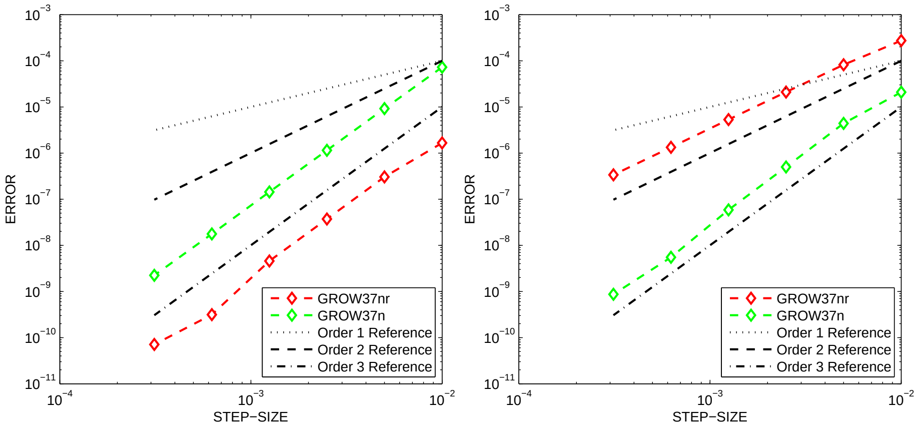


Figure D.1: Comparison of GROW37nr and GROW37n solving problem (6.5) using $A_y = 0$, $A_z = 0$ and $B_y = 0$ together with exact B_z (left) and non-exact B_z (right).