COMPUTATIONAL COMPLEXITY OF NUMERICAL SOLUTIONS OF INITIAL VALUE PROBLEMS FOR DIFFERENTIAL ALGEBRAIC EQUATIONS

(Spine title: Computational Complexity of
Numerical Solutions of IVP for DAE)

(Thesis format: Monograph)

by

Silvana Ilie

Graduate Program in Applied Mathematics

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

Faculty of Graduate Studies
The University of Western Ontario
London, Ontario
November, 2005

THE UNIVERSITY OF WESTERN ONTARIO
FACULTY OF GRADUATE STUDIES

CERTIFICATE OF EXAMINATION

Chief Advisor	Examiners
Prof. Robert M. Corless	Prof. Gustaf Söderlind
Co-Supervisor	Prof. David Jeffrey
Prof. Greg Reid	Prof. Pei Yu
	Prof. Roy Eagleson

The thesis by

Silvana Ilie

entitled

COMPUTATIONAL COMPLEXITY OF NUMERICAL SOLUTIONS OF INITIAL VALUE PROBLEMS FOR DIFFERENTIAL ALGEBRAIC EQUATIONS

is accepted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Date ______________________ ______________________
Chair of the Thesis Examination Board

Abstract

Computational Complexity of Numerical Solution
of Initial Value Problems for
Differential Algebraic Equations

Silvana Ilie
Doctor of Philosophy
Graduate Department of Applied Mathematics
The University of Western Ontario
2005

We investigate the cost of solving initial value problems for differential algebraic equations depending on the number of digits of accuracy requested. A recent result showed that the cost of solving initial value problems (IVP) for ordinary differential equations (ODE) is polynomial in the number of digits of accuracy. This improves on the classical result of information-based complexity, which predicts exponential cost. The new theory is based on more realistic assumptions.

The algorithm analysed in this thesis is based on a previously published Taylor series method for solving a general class of differential algebraic equations. We consider DAE of constant index to which the method applies. The DAE is allowed to be of arbitrary index, fully implicit and have derivatives of order higher than one.

Similarly, by considering a realistic model, we show that the cost of computing the solution of IVP for DAE with the algorithm adopted and by using automatic differentiation is polynomial in the number of digits of accuracy. We also show that nonadaption is more expensive than adaption, giving thus a theoretical justification of the success of adaptivity in practice.

A particular case frequently arising in practical applications, the index-1 DAE, is treated separately, in more depth. On the other hand, an analysis of the higher-index DAE is significantly more complicated and applies to a wider class of problems. In both cases, continuous output is also given.

These results apply to many important problems arising in practice. We present an interesting theoretical application to polynomial system solving.

Keywords: Differential Algebraic Equations, Initial Value Problem, Adaptive Step-size Control, Taylor Series, Structural Analysis, Computational Complexity, Automatic Differentiation.

To my parents,

To Teo.

Acknowledgements

I wish to express my deepest gratitude to my supervisor, Professor Rob Corless. I would like to thank him for proposing me an exciting and challenging problem and for introducing me to the subject of differential algebraic equation solving and to numerical analysis. His expert advice, careful and kind guidance, and extraordinary support have been invaluable in all stages of completing this work. I am very fortunate to have worked under your supervision! I cannot thank you enough.

I would like to express my gratitude to Professor David Jeffrey. His generous support and encouragements during my PhD studies made such a difference. I am grateful for many stimulating discussions, for his wonderful advice, scientifically and beyond. Your writing style sets high standards. I tried to improve mine under your expert eye.

I deeply thank both of you for your friendship and great support.

I would like to thank Professor Greg Reid, my co-supervisor, for many inspiring discussions and for his enthusiasm for my work. My understanding of homotopy continuation methods for polynomial solving and the geometrical insight on DAE solving are due to you. Thank you for careful reading of my thesis, which helped greatly improve its rigor.

I am very grateful to Professor Gustaf Söderlind for agreeing to be external examiner of my thesis. I am honored.

This work has benefit from suggestions of several people to whom I am very grateful: Steve Campbell, George Corliss, Wayne Enright, Ned Nedialkov, John Pryce, Gustaf Söderlind and Jan Verschelde.

The Department of Applied Mathematics and the Ontario Research Centre for

Computer Algebra have offered an excellent working environment.

Gayle McKenzie, Pat Malone and Audrey Krager have helped me so much with all the administrative problems and I thank them.

I would like to thank my father for cultivating my love for mathematics and my mother for making everything possible. I thank my son, Teo, your effort and patience were biggest among all, and my husband, Lucian, for great support and understanding.

Finally, the financial support of the Ontario Graduate Scholarship is gratefully acknowledged.

Contents

Chapter 1

Introduction

The study of numerical solution of differential algebraic equations has attracted much attention due to its many important practical applications. These applications include robotics, electrical circuits, mechanical systems, fluid mechanics and many others. Intensive work has been done so far for numerically solving these problems in an efficient way.

Initially, the DAE problems studied were those with a special structure and typically this structure was exploited by the numerical method used to solve them. The first method of numerically solving a DAE (a backward differentiation formula method - see [8]) is attributed to Gear in 1971 [33], for problems from computer aided design of electrical networks.

Later on, more complex structures were considered and many advances made in both the theory and numerical solution of DAE, see Gear [34, 35], Gear and Petzold [36], Brenan, Campbell and Petzold [36], Hairer, Lubich and Roche [39], Hairer and Wanner [41], Mattsson and Söderlind [55], Rabier and Rheinboldt [68], Campbell and Griepentrog [14], and others.

Taylor series methods for DAE, which are the object of analysis in this thesis, have proved to be quite successful. The idea of automatically generating recursion formulae for efficient computation of Taylor series goes back at least to Moore (see [59], Chapter 11). The general method he describes is applicable to functions satisfying rational differential equations. He also discusses the low cost of this method. Full recurrence formulae for Taylor coefficients appeared in an earlier publication by Miller [57]. Rall [69] discusses in detail algorithms for automatic differentiation and also the generation of Taylor coefficients. He also gives applications of Taylor series methods. A recent comprehensive study of automatic differentiation and applications is given in Griewank [38].

Corliss and Chang [25] discuss the solution of IVP for ODE using Taylor series and automatic differentiation. Efficient methods for series analysis and singularity detection are also given (see also [15]).

Pantelides [64] gives a method for DAE solving which iteratively identifies constraints for problems of the form $F(y', y, t) = 0$. His structural analysis is generalized by Pryce [65, 66] to systems allowing higher derivatives. Unlike Pantelides' approach, in Pryce's analysis, the "offsets" of the DAE (which give a systematic way of solving the equations) are obtained directly as solution of an assignment problem (which generalizes Pantelides' transversal method).

Pryce's structural analysis may be used also for solving the DAE with Taylor series. This Taylor series method (using automatic differentiation) gave good results in practice for mildly stiff problems and high accuracy. For stiff problems, Hermite-Obreschkoff methods [49] offer an efficient alternative.

The first paper we are aware of showing how to organize the equations to solve a DAE by Taylor series method is due to Corliss and Lodwick [26].

Another approach for DAE solving is the derivative array method of Campbell [13]. That method consists in differentiating all equations in the system the same number of times, until one can solve for the first derivative of the dependent variable as function of dependent and independent variables. This method is more expensive, but more general than previous methods.

Nedialkov and Pryce [62] show how to implement Pryce's structural analysis for solving the DAE using Taylor series in the general form of the system, without applying index reduction.

Hoefkens [42] exploits another application of Pryce's structural analysis, the reduction of a DAE to the underlying ODE.

Another important practical aspect of numerically solving differential equations is finding optimal meshes, which, as seen in practice for many problems, are adaptive meshes.

Adaptive time-stepping in numerical solution of differential equations is mainly based on heuristics. The important task of constructing theoretically rigorous and practically efficient techniques for adaptivity is a difficult one. A key contribution in achieving optimal meshes is due to Söderlind in [75, 76]. New efficient techniques of adaptive time-stepping are introduced there and given theoretical justification. This thesis requires and assumes such good adaption techniques throughout.

1.1 Differential algebraic equations

A general class of implicit ordinary differential equation is

$$F(y', y, t) = 0 \tag{1.1}$$

where $y : I \subset \mathbb{R} \to \mathbb{R}^m$, $F : D \times I \subset \mathbb{R}^{2m+1} \to \mathbb{R}^m$ and the Jacobian

$$\frac{\partial F}{\partial y'}(y', y, t) \tag{1.2}$$

is non-singular at points $(y', y, t) \in \mathbb{R}^{2m+1}$ satisfying $F(y', y, t) = 0$. According to the Implicit Function Theorem, it is therefore possible in principle to solve (1.1) for the derivative of the unknown, y', and the problem becomes equivalent to solving an explicit ordinary differential equation:

$$y' = f(y, t) \ .$$

But for many physical problems, the Jacobian (1.2) is singular and it is not straightforward how to solve for y' as function of y and t (i.e. how to reduce the problem to an explicit ordinary differential equation).

Systems of general form (1.1) for which the Jacobian may be singular are called *differential algebraic equations* (DAE). In general form, a DAE may also contain higher order derivatives of the unknown function and the problem can be written as

$$F(y^{(n)}, \ldots, y', y, t) = 0 \ .$$

We note that an ODE is a particular case of a DAE. While many numerical solvers are available for ODE and many theoretical results of existence, uniqueness and regularity are known, for DAE the picture is not as complete.

We note that many methods of numerically solving DAE exist in the literature. Among them: differentiation to get an ODE (a disadvantage is that it may produce instability in the numerical integration - the 'drift off' phenomenon, when error in constraints accumulates rapidly in time), adaptations of standard numerical methods [41, 8], geometric (jet space) methods [68, 70], Taylor series methods, and so on.

DAEs are generally harder to solve than ODEs. Various concepts of 'index' have been developed to attempt to quantify the numerical difficulty of swolving DAEs. DAE of index, in terms of these concepts, higher than 2 are generally considered hard to solve numerically. These DAEs are often transformed through index reduction techniques, to problems of index-1 or to ODEs. In this thesis, we shall not adopt these strategies, but solve systems of higher index in their original form, by using the Taylor series method developed in [65, 66]. We show that this can be done efficiently. The index-1 case is treated separately, both for ease of understanding of the general (higher index) case and because index-1 problems are common and of practical interest themselves.

In the next section, we discuss various notions of index.

1.2 Index

The complexity results in this thesis apply to systems of DAE for which Pryce's method [66] can be successfully applied. When this method succeeds, a certain integer ν_s is determined, which depends on the offsets of the problem (see the definition (4.2) and also [65, 66]).

This integer ν_s, called the *structural index*, is related to other notions of index for DAE occuring in the literature. We note however that, regardless of the relation to other such notions, the validity of the computational cost results of this thesis only depends on the successful application of Pryce's method.

We will now describe some of these related concepts of index. Generally, these concepts of index are measures of how singular a DAE is. Also, the higher the particular index is, generally the more difficult it is to solve the DAE.

A survey on several definitions of index can be found in [13]. At early stages, the 'differential index' was defined for systems with a certain structure. Relations were derived for the available definitions and various refinements have been made.

Two definitions of the index are widely used: *differential index* (ν_d) and *perturbation index* (ν_p).

The perturbation index, introduced in [39], measures the sensitivity of the solutions with respect to perturbations to the given problem. According to [41], the equation (1.1) has *perturbation index* $\nu_p = m$ along a solution y on $[0, T]$ if m is the smallest integer such that, for all $\hat{y}$ with

$$F(\hat{y}', \hat{y}, t) = \delta(t)$$

and there exists some constant C such that, on $[0, T]$,

$$\|\hat{y}(t) - y(t)\| \leq C \left(\|\hat{y}(0) - y(0)\| + \max_{0 \leq s \leq t} \|\delta(s)\| + \ldots + \max_{0 \leq s \leq t} \|\delta^{(m-1)}(s)\| \right)$$

for small right hand side.

According to [35], the following relation is valid when both indices exist: $\nu_p \leq \nu_d + 1$.

This thesis employs the structural index defined by (4.2). For a discussion of relations between this index and the structural index of Pantelides [64] or the *uniform index* of Campbell and Gear [13] see [66].

In [66], it is stated that, under certain assumptions, the structural index of (1.1) relates to the differential index by $\nu_d \leq \nu_s$. Equality does not hold in general. An example of a family of DAEs for which differential index is 1 and structural index can be arbitrary is given in the literature.

Example. The following DAE has differential index 1 (see, e.g., [66] for a similar example):

$$\begin{aligned}
y_2' + y_3' + y_1 &= f_1(t) \\
y_2' + y_3' + y_2 &= f_2(t) \\
y_4' + y_5' + y_3 &= f_3(t) \\
y_4' + y_5' + y_4 &= f_4(t) \\
y_6' + y_7' + y_5 &= f_5(t) \\
y_6' + y_7' + y_6 &= f_6(t) \\
y_7 &= f_7(t)\ .
\end{aligned}$$

With the notations from Chapter 4, the offsets of the problem are $c = (0, 0, 1, 1, 2, 2, 3)$ and $d = (0, 1, 1, 2, 2, 3, 3)$ and therefore the structural index is 4.

In general, the index of a DAE may depend on the particular solution computed. Moreover, geometrical approaches to DAE [68, 70] describe indices which may vary from one component to another of the DAE. Such a dependence is not considered in our analysis; we assume that the index is constant along neighbouring solution paths.

DAEs with a restricted structure have been widely studied and specific numerical methods have been developed for them.

Example. The following DAE

$$\begin{aligned}
y' &= f(t, y, z) \\
0 &= g(t, y, z)
\end{aligned}$$

is in *Hessenberg Index*-1 form if the Jacobian g_z is invertible along the solution path.

A DAE is in *Hessenberg Index*-2 form if

$$\begin{aligned}
y' &= f(t, y, z) \\
0 &= g(t, y)
\end{aligned}$$

and the matrix $g_y f_z$ is invertible along the solution path. In many practical applications, the variable z is in fact a Lagrange multiplier.

Other systems with a special structure include DAE in *Hessenberg form* of higher index, constrained mechanical systems, and triangular chains of systems. For all these particular cases of DAE, the structural method developed by Pryce works. Consequently, the general results of this thesis apply to them also.

1.3 Initial conditions

Unlike index-1 DAE, higher index DAE have hidden constraints. These constraints can be revealed by analytic differentiation of the prescribed constraints.

For initial value problems, in the case of a size m ODE, exactly m initial conditions have to be given and these conditions may be chosen arbitrarily over a wide range. In the case of a size m DAE, the number of initial conditions which have to be given is not obvious. The number of conditions needed for initialization depends on the degrees of freedom of the system, which has to be determined.

Moreover, the initial conditions of a DAE have to be *consistent* with the constraints, that is they should satisfy the given constraints and all the hidden constraints of the systems (uncovered by differentiation of the prescribed constraints). It is not known a priori what other restrictions are imposed on the initial conditions by the new constraints, obtained by differentiation of the given constraints, and if further variables should be initialized also. Thus the problem of prescribing a set of consistent initial conditions is a difficult one.

Fortunately, one of the features of Pryce's structural analysis—on which the method analysed in this thesis is based—is that it provides also a systematic way of consistently initializing the solution of DAE.

1.4 Automatic differentiation and generation of Taylor coefficients

Taylor series methods for solving differential equations have many advantages, from elegant and easy formulation to the fact that they allow variable (and, in particular, high) order and variable stepsize. As a consequence, these methods allow larger stepsizes than low order methods when high precision is required, being thus less costly as we shall see quantitatively. They are a good choice for high-precision computations. They also provide dense outputs (the Taylor polynomials), and they may to be used for other purposes such as interval arithmetics [69].

Initially, the theoretical beauty of the method was shadowed in practice by the fact that to obtain higher order derivatives, done with usual differentiation, was very costly for large number of terms in the series.

Due to a powerful numerical tool, Automatic Differentiation (AD) [69, 38], this problem was overcome. AD represents a set of procedures that take as input a computer program which evaluates a function and generates a program which computes the derivative of that function. It is based on the application of the rules for differentiation of the elementary calculus (chain rule, Leibnitz's rule) to a set of codable functions for which the corresponding derivatives are also codable functions. Derivatives up to arbitrary order may be generated automatically, using these rules.

The key idea is that the functions to which AD apply, no matter how complicated, are in fact obtained by applying only the elementary arithmetic operations to elementary functions such as sin, exp, ln and/or to functions such as Bessel functions, which can be represented through their defining differential equations [69].

By using recurrence formulae, to compute the solution of an ODE with p terms in the Taylor series is merely quadratic cost with respect to p [25, 38]. For problems where no products of functions appear, the cost drops to linear growth with respect to the number of terms. Due to this low cost, Taylor series methods can be quite competitive on large sets of problems.

1.5 Practical applications

We discuss below a few of the numerous important practical applications where the systems are naturally modelled by DAE. We also give two examples, from many practical problems, which are modelled as IVP for DAE (problems to which Pryce's method and hence our complexity results apply).

Among other applications of DAE not discussed here, we only mention trajectory prescribed path control, semiconductor device simulation, biological processes, such as pulse transmition across a nerve synapse, and chemical reaction kinetics.

It is important to mention that solvers using Pryce's structural analysis (see [61]) behave very well also on standard test problem sets [56]. The polynomial complexity results predicted by our model are thus in good agreement with what is obtained in practice [4, 60, 67].

1.5.1 Constrained mechanical systems

The motion of a wide class of mechanical systems with constraints are modelled as differential algebraic equations of the special structure [41, 8]

$$\begin{aligned} M(p)p'' &= f(t,p,p') - G(p)^T\lambda \\ \phi(p) &= 0\ , \end{aligned}$$

where $G(p) = \dfrac{\partial\phi(p)}{\partial p}$ and is assumed full row rank. The matrix M represents the mass matrix which is in general assumed positive-definite and symmetric, p gives the position of the mass points, f the external force, λ is the Lagrange multiplier and the term $G(p)^T\lambda$ denotes the constraint forces. Such forces appear due to the presence of the constraint $\phi(p) = 0$. Note that $p \in \mathbb{R}^n$, $\lambda \in \mathbb{R}^m$ with $m \leq n$.

Many multibody mechanics problems consist of less than several hundred equations. Also many of these problems are modelled as DAE with high index (up to six). For this type of problems we expect our complexity analysis to give good predictions when tight tolerance is required.

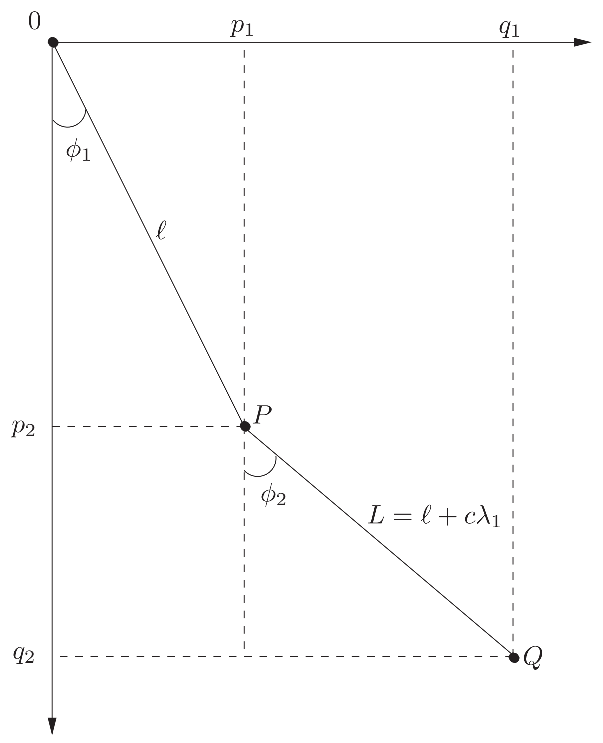

Figure 1.1: Double pendulum connected by strings of length ℓ and L

Example One example of constrained mechanical system is the system of equations for the double pendulum [66] (see Figure 1.1). The problem is modelled by an index-5

DAE, governed by the equations

$$\begin{aligned} p_1'' + p_1\lambda_1 &= 0 \\ p_2'' + p_2\lambda_1 - g &= 0 \\ p_1^2 + p_2^2 - \ell^2 &= 0 \\ q_1'' + q_1\lambda_2 &= 0 \\ q_2'' + q_2\lambda_2 - g &= 0 \\ q_1^2 + q_2^2 - (\ell + c\lambda_1)^2 &= 0 \end{aligned}$$

where (p_1, p_2), (q_1, q_2) are the Cartesian coordinates of the two pendulae; $\lambda_1,\ \lambda_2$ are the Lagrange multipliers and denote the tensions in the strings; g is the acceleration due to gravity. The length of the second pendulum is controlled by the tension in the first pendulum.

For the initial value problem, $p_1,\ p_2,\ q_1,\ q_2$ and their first derivatives should be prescribed at the initial time.

1.5.2 Electrical circuits

The study of the numerical solution of DAEs has long been stimulated by the problems encountered in computer aided design of electrical networks. The sparsity which characterizes these systems makes them good candidates for directly solving the DAE, rather than reducing the problem to the underlying ODE, a technique which can destroy its structure.

The problems are often of the form

$$M\frac{dy}{dt} = f(y) \tag{1.3}$$

where M is a singular matrix. By a few obvious linear transformations these problems often preserve their sparse structure and are transformed to DAEs to which the structural analysis of Pryce can be successfully applied.

These systems are derived from applying Ohm's law for a current through a resistor: $I = U/R$; the current through a capacitor satisfies $\left(I = C\dfrac{dU}{dt}\right)$, where U is the voltage and R and C are constants (representing the resistance and the capacitance, respectively); the current through a transistor satisfies a law of form $I_E = F(U)$, $I_c = -\alpha I_E$, $I_B = (\alpha - 1)I_E$, where α is a characteristic quantity of the transistor. For each node in a circuit, Kirchoff's law applies: the sum of all currents entering the node is zero at all times.

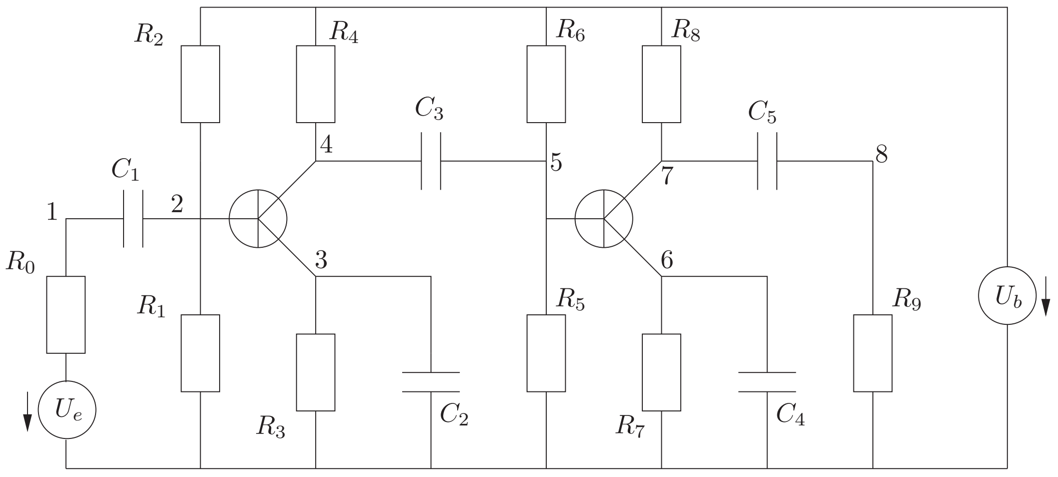

Figure 1.2: The transistor amplifier, according to [56].

Example Consider the transistor amplifier problem [56] (see Figure 1.2), which is an index-1 DAE with 8 equations. The system is of the form (1.3) with y, y' prescribed at the initial time.

The matrix M is of rank 5 and is given by

$$M = \begin{pmatrix} -C_1 & C_1 & & & & & & \\ C_1 & -C_1 & & & & & & \\ & & -C_2 & & & & & \\ & & & -C_3 & C_3 & & & \\ & & & C_3 & -C_3 & & & \\ & & & & & -C_4 & & \\ & & & & & & -C_5 & C_5 \\ & & & & & & C_5 & -C_5 \end{pmatrix}$$

and the function f is given by

$$f(y) = \begin{pmatrix} -\frac{U_e(t)}{R_0} + \frac{y_1}{R_)} \\ -\frac{U_b}{R_2} + y_2(\frac{1}{R_1} + \frac{1}{R_2} - (\alpha - 1)g(y_2 - y_3) \\ -g(y_2 - y_3) + \frac{y_3}{R_3} \\ -\frac{U_b}{R_6} + y_5(\frac{1}{R_5} + \frac{1}{R_6} - (\alpha - 1)g(y_5 - y_6) \\ -g(y_5 - y_6) + \frac{y_6}{R_7} \\ -\frac{U_b}{R_8} + \frac{y_7}{R_8} + \alpha g(y_5 - y_6) \\ \frac{y_8}{R_9} \end{pmatrix}$$

and g, U_e are auxiliary functions. U_e represents the input signal and y_8 the amplified output voltage. The solution has been obtained using, for example, HIDAES [61].

1.6 Thesis Outline

In Chapter 2 the background material on information-based computational complexity is presented. The basic terminology and the assumptions of the standard theory are given. A standard result, given by [83], that adaptive step size strategies are no better than nonadaptive ones in the worst case setting is briefly discussed.

This same chapter continues with a survey of the existing models and results of the computational complexity of initial value problems (IVP) for ordinary differential equations (ODE). A result on optimal mesh selection, previously published, is also given, with a new straightforward proof. This result is important for the subsequent chapters.

Chapter 3 is dedicated to the study of computational complexity for IVP for index-1 DAE. This is original work. It gives a full treatment of details of the model used in [47], but omitted in that paper. For example it gives a general result on bounds of forward error in terms of residuals for the case of piecewise continuous functions in the DAE. It also gives the construction of a continuous dense output and the derivation of residual error estimates for the algorithm analysed.

Chapter 4 deals with the computational complexity of IVP for higher-index DAE. It covers the much larger class of all DAE with index higher than 1, not covered in the previous chapter, and to which Pryce's method of DAE solving applies.

The theoretical results on which Pryce's method is based are surveyed. The main result of the chapter concerns the polynomial cost of solving IVP for DAE. We detail the intermediate steps in building and analysing our model. For example, we give the construction of continuous dense output for arbitrary index DAE to which Pryce's method applies, the derivation of cost of automatic differentiation in solving the problem, and a full error analysis, which is more complex for the high-index case than for the index-1 case. Examples are also included. This is also original work and is a detailed version of our paper [21].

Chapter 5 contains an application of the complexity result in Chapter 3 to an important problem in polynomial system solving, that of finding all isolated roots. Homotopy continuation methods are applied to give a polynomial cost method to solve the problem.

Conclusions and directions for future research are the object of Chapter 6.

The Appendix contains further useful theory on ODE and DAE solving.

Chapter 2

Computational Complexity

Many problems arising in science and engineering are modelled as differential equations. These problems can be solved exactly in only a few cases. In the general case, solutions can be obtained only approximately, by numerical computation. The information available in numerically solving such problems is usually only partial. Since a computer can only deal with finite sets of numbers, objects of infinite-dimensional spaces, such as real valued functions, need to be replaced in the computation by finite sets of numbers, a correspondence which is obviously many-to-one. Furthermore, for such computations, the information is contaminated, by measurement errors when data is obtained experimentally and by round-off errors when data is generated by computer codes.

Such information comes at a price which can be measured, for example, by the cost of running experiments to obtain the necessary data or by the number of operations performed during function evaluations in the solution process.

Examples of problems modelled by differential equations for which the information is partial, contaminated and priced occur in most practical applications of numerical differential equation solving.

The theory which measures the minimal computational resources to solve a mathematical problem and searches for optimal algorithms for solving it is called *computational complexity.* The complexity of a problem can be regarded as the minimal cost among all algorithms for solving it. This complexity is obviously independent of any algorithm, it is specific to the problem, and gives the limitations of solving it numerically.

The branch of computational complexity which studies the intrinsic difficulty of approximating the solution of problems for which the information is partial, noisy and priced is called *information-based complexity.*

Given a certain problem, we wish to construct algorithms to find an approximate solution with a specified tolerance.

If the number of digits of the tolerance, B, is considered as a parameter of the problem, one key question is how the cost of the solution changes when B is varied? In particular, it is interesting to study the asymptotic behaviour of the cost of the algorithm as B becomes large. This behaviour provides a useful model for practical computation. The goal of the theory is to approximately solve the mathematical problem as cheaply as possible. Therefore, a cost (time or space) that grows exponentially with the parameter is considered 'bad'. We call the problem *tractable* if its complexity grows polynomially, and *intractable* if the complexity grows super-polynomially (e.g. exponentially).

Furthermore, given a positive tolerance ε, we wish to find the ε-*complexity* for our problem, that is the minimal cost to compute an approximation with a tolerance at most ε for the solution. We note that for a tolerance ε the corresponding number of digits of accuracy is

$$B = \lceil \ln(1/\varepsilon) \rceil \ .$$

We also wish to find an algorithm with optimal complexity, one which guarantees at most the desired error at the minimal cost. Also, for practical reasons, it is very important to know under which conditions the standard algorithms perform well.

In order to find the computational complexity of a problem, we need to define the error and the cost of the algorithms. In other words, we need to specify the *setting* for our problem. We shall be particularly interested in the *worst case* setting, for which the cost and the error are defined on the hardest problem. Other settings are possible, such as average case setting, probabilistic setting, but they are not considered in this thesis. Finally, it is important to note that the computational complexity of the problem will depend on the particular setting considered. The aim is of course to construct a model which reflects what is observed in practice. If the theoretical results are obtained in an environment which is not similar to practical computation, they may be misleading for writing efficient codes.

In this chapter we shall review the existing models and the corresponding results for the computational complexity of IVP for ODEs. After an introduction to the general terminology and the assumptions of the standard theory of computational complexity [79, 84], we shall restrict our attention to the worst case setting and we shall revisit a controversial standard result: "adaption no better than nonadaption" for solving linear problems. We shall describe the setting and the results of the standard theory for the complexity of numerically solving of IVP for ODEs. The main standard result states that the minimal cost of obtaining the solution for an initial value problem for ordinary differential equations to accuracy ε is exponential in B (i.e. the problem is intractable in the standard setting). We also review results showing that some standard algorithms such as Taylor series methods are almost optimal for the model under consideration.

A more recent model, introduced by Corless in [18], is then presented. The main result for the new model shows that the cost of solving initial value problems for ordinary differential equations is polynomial in the number of digits of accuracy (i.e. the problem is tractable), improving thus on the results of the standard theory. How does the new theory break intractability? The new results are based on different, more realistic assumptions. Starting from a recent result of equidistribution [17], the theory also shows that adaptive meshes perform no worse than non-adaptive ones when solving IVP for ODEs, in agreement with the practical experience. A new, straightforward proof of the equidistribution result is also presented.

While interesting theoretically, the standard point of view is not a realistic model for practical computation. Good practical codes are efficient enough not to be of exponential cost and it has been observed in practice that nonadaption is more expensive than adaption. Hence the new setting seems more useful.

2.1 Standard theory

We begin this section by describing the general terminology and assumptions of the theory of information-based complexity, before restricting our attention to some specific setting. In doing this, we follow the standard textbooks [79, 84].

Before starting the presentation of the standard theory, we must mention that the notations and results of this section will not be used elsewhere in this thesis. If the reader is not interested in the standard results, he/she may skip this introduction and go directly to section 2.2.

In order to build the model for analysing the complexity of a certain problem, we need to give the formulation of the problem, the set of information which is permissible in solving the problem and the model of computation. We pause and take a closer look at these concepts.

- *Formulation of the problem.* In the general case, we consider a mapping $S : F \to V$ from a set F to a normed linear space V. The operator S is called a *solution operator* and F is the set of *problem elements.* The set F is given by a linear surjective restriction operator $T : F_1 \to X$ from a linear space F_1 to a normed linear space X by

$$F = \{f \in F_1 : \|Tf\| \leq 1\}.$$

- *Information.* In order to approximate the solution element Sf for a given problem element $f \in F$, we assume that partial information about the problem element f is available, by means of some functionals $\lambda : F \to \mathbb{R}$. Let Λ be the class of permissible information operations, which may belong to one of the following classes of linear information:

  – Standard information Λ^{std}: we assume that F is a space of real valued functions defined over some domain and that for every $f \in F$ and every x in the given domain, the evaluation of f and its derivatives at x are permissible.

  – Continuous linear information Λ^*: we assume that F_1 is a normed linear space and any linear continuous functional on F_1 is permissible.

  – Arbitrary linear information Λ': we assume that any linear functional on F_1 is permissible.

  The information Υ is said to be *non-adaptive* (or *parallel*) if there exists $N \in \mathbb{N}$ and $\lambda_1, \cdots, \lambda_N \in \Lambda$ such that

$$\Upsilon f = \begin{bmatrix} \lambda_1(f) \\ \vdots \\ \lambda_N(f) \end{bmatrix} \quad \text{for any } f \in F.$$

Thus neither N (the cardinality of Υ) nor the linear functionals $\lambda_1, \cdots, \lambda_N$ depend on f.

The information Υ is called *adaptive* (or *sequential*) if for each $f \in F$ there exists $N(f) \in \mathbb{N}$ (the cardinality of Υ at f) and a set of functionals $\lambda_1, \cdots, \lambda_{N(f)}$ so that

$$\Upsilon^{ad} f = \begin{bmatrix} \lambda_1(f) \\ \lambda_2(f; y_1) \\ \vdots \\ \lambda_{N(f)}(f; y_1, \cdots, y_{N(f)-1}) \end{bmatrix} \quad \text{for any } f \in F. \tag{2.1}$$

where

$$\begin{cases} y_1 = \lambda_1(f) \\ y_k = \lambda_k(f; y_1, \cdots, y_{k-1}) \end{cases}$$

for $2 \leq k \leq N(f) - 1$. Notice that each functional λ_k depends on the previous computed values.

- *Model of computation* (the abstract model of the computer). We assume that the cost of a function evaluation does not depend on the function. We denote it by the positive constant

$$c(\lambda, f) = c$$

(independent of $\lambda \in \Lambda, f \in F$).

We also assume that the combinatory operations over the space V (including scalar multiplication with real numbers and the sum of two elements in V) as well as the arithmetic operations with real numbers can be performed exactly at unit cost. A function evaluation is usually much more expensive than an arithmetic operation, so we assume that $c \gg 1$.

> We remark that the non-adaptive information has the advantage that it can be computed in parallel, which is very efficient, but for the remainder of this dissertation we consider that the computation is sequential, that is we assume that the cost of a set of operations is the sum of each cost.

Once the information Υ is specified, we model an algorithm using Υ as a mapping $\alpha : \Upsilon(F) \to V$, which takes as input some admissible information Υf on a problem element $f \in F$ and produces as output an approximation of the solution $Sf \in V$.

We are interested in computing the cost of the algorithm, $\mathrm{cost}(\alpha, \Upsilon, f)$. This cost is composed of the *informational cost* which is the cost of evaluating the information Υf and the *combinatory cost*, which is the cost of obtaining the approximation $\alpha(\Upsilon f)$, once the information Υf is known.

Notice that the information cost corresponding to a non-adaptive information Υ of cardinality N is cN. Also, if k is the number of permissible combinatory operations needed to calculate $\alpha(\Upsilon f)$, knowing Υf, then the combinatory cost is k. In practice, though, these costs depend on precision.

The error and the cost of an algorithm α using the information Υ in the worst case setting are defined by their worst values

$$\begin{aligned} e(\alpha, \Upsilon) &= \sup\nolimits_{f \in F} \|Sf - \alpha(\Upsilon f)\| \\ \mathrm{cost}(\alpha, \Upsilon) &= \sup\nolimits_{f \in F} \mathrm{cost}(\alpha, \Upsilon, f) \ . \end{aligned}$$

The ε-*complexity* for a problem is defined by

$$\mathrm{comp}(\varepsilon) = \inf\{\mathrm{cost}(\alpha, \Upsilon) : \text{ with } e(\alpha, \Upsilon) \leq \varepsilon\}.$$

An algorithm α_ε using information Υ_ε is called ε-*optimal complexity algorithm* if

$$e(\alpha_\varepsilon, \Upsilon_\varepsilon) \leq \varepsilon \text{ and } \mathrm{cost}(\alpha_\varepsilon, \Upsilon_\varepsilon) = \mathrm{comp}(\varepsilon)$$

and is called *almost optimal complexity algorithm* if there exists a constant d of the same order of magnitude as 1 so that

$$e(\alpha_\varepsilon, \Upsilon_\varepsilon) \leq \varepsilon \text{ and } cost(\alpha_\varepsilon, \Upsilon_\varepsilon) \leq d \cdot comp(\varepsilon).$$

2.1.1 Adaption vs. nonadaption

"This result (adaption is no better than nonadaption) has not been enthusiastically accepted by many practitioners of scientific computation, since the conventional wisdom is that for 'practical' problems, adaption is better than nonadaption." - Werschulz [84], p.124

We review below a result which states that adaption is no better than non-adaption for linear problems in the worst case setting.

A problem is considered *linear* if, in addition to the assumptions above, T is a linear operator, $S : F \to V$ is the restriction of a linear operator $S : F_1 \to V$ and the permissible information operations are all linear functionals.

Let Υ^{ad} be an adaptive linear information given by (2.1). We construct a non-adaptive information of the particular form

$$\Upsilon^{non} f = \begin{bmatrix} \lambda_1(f) \\ \lambda_2(f; 0) \\ \vdots \\ \lambda_{n(0)}(f; 0, \cdots, 0) \end{bmatrix} \quad \text{for any } f \in F.$$

Theorem 1 *For any linear problem and any linear adaptive information Υ^{ad}, consider the non-adaptive information (of lower cardinality) Υ^{non} defined above. Then the minimal error among all algorithms using the information Υ^{non} is less than double the minimal error among all algorithms using the information Υ^{ad}.*

Consequently, adaption does not help, or it gives an improvement by a factor at most two for linear problems in the worst case-setting.

The result for adaptive information of constant cardinality ($n(f) = n(0)$ for all $f \in F$) is given in [80].

The result for adaptive information of variable cardinality has been shown by Wasilkowski [83].

2.1.2 Ordinary differential equations

We shall present in this section the results of the standard theory on the complexity of initial value problems for ordinary differential equations. Most of the results in this section are due to Kacewicz [53].

The following assumptions are made: D is a bounded open convex subset of $\mathbb{R}^{d+1}$,

$$F_1 = \{f : \mathbb{R}^{d+1} \to \mathbb{R}^d,\ \text{supp}(f) \subseteq D,\ D^\alpha(f) \text{ continuous for all } |\alpha| \le r\}$$

which is a Banach space with the norm

$$\|f\|_r = \sum_{|\alpha| \le r} \|D^\alpha f\|_{\text{supp}},$$

F is the unit ball of F_1, and $V = C([0,1])$ with the supremum norm. The solution operator is defined by $Sf = u$, where u is the solution of the following problem

$$\begin{aligned} u'(t) &= f(t, u(t)) \\ u(0) &= U_0 \end{aligned} \tag{2.2}$$

for $t \in (0,1)$. Note we assume that the solution of the IVP exists for all $0 \le t \le 1$.

Consider a partition of $[0,1]$ given by

$$0 = t_0 < t_1 < \cdots < t_m = 1$$

with $t_k = k/m$ and corresponding constant step-size $h = 1/m$. We define

$$u_j(t) = U_j + \sum_{k=0}^{r-1} \frac{(t-t_j)^{k+1}}{(k+1)!} \frac{d^k}{dt^k} f(t, u(t))|_{t=t_j, u(t)=U_j} \tag{2.3}$$

for $0 \le j \le m-1$ and $t \in [t_j, t_{j+1}]$. Consider $U_{j+1} = u_j(t_{j+1})$.

The Taylor information is given by

$$\Upsilon_{r,h}^T f = \{D^\beta f(t_j, U_j) : |\beta| \le r, 0 \le j \le m-1\}$$

and the Taylor series algorithm by

$$\alpha_{r,h}^T(t) = u_j(t) \text{ for } t_j \le t \le t_{j+1} \ .$$

Theorem 2 *Under the assumptions above and $\Lambda = \Lambda^{std}$:*

- *the complexity of the problem (2.2) is $\Theta(\varepsilon^{-1/r})$ as $\varepsilon \to 0$.*
- *the Taylor series algorithm using stepsize $h = \Theta(\varepsilon^{1/r})$ as $\varepsilon \to 0$ is an almost optimal complexity algorithm, i.e.*

$$e(\alpha_{r,h}^T, \Upsilon_{r,h}^T) \le \varepsilon$$

and

$$cost(\alpha_{r,h}^T, \Upsilon_{r,h}^T) = \Theta(\varepsilon^{-1/r}).$$

The main steps of the proof are as follows. First it is shown that the Taylor series algorithm gives a global error of order h^r. Then it is shown that solving the problem (2.2) with a Taylor series algorithm $\phi_{r,h}^T$ using information $\Upsilon_{r,h}^T$ with $h = \Theta(\varepsilon^{1/r})$ is achieved with cost $\Theta(\varepsilon^{-1/r})$. Finally, using a result of Kacewicz [51], it is proved that the bound $\Theta(\varepsilon^{-1/r})$ for the computational cost of solving (2.2) with the Taylor series algorithm can not be improved in the class of algorithms using standard information of the same cardinality.

Other examples of almost optimal complexity algorithms are the explicit Runge-Kutta algorithms of order r. This result is due to Werschulz [84]. Let us consider the r-th order Runge-Kutta method on a grid over $[0, 1]$ of fixed step-size h. Consider $t_{j+1} = t_j + h$ for $0 \leq j \leq m-1$. We approximate $u(t_j)$ by U_j for $0 \leq j \leq m-1$ using the following intermediate steps:

$$\begin{aligned}
k_{1,j} &= f(t_j, U_j) \\
k_{2,j} &= f(t_j + c_2 h, U_j + h \cdot a_{21} k_{1,j}) \\
\cdots & \\
k_{s,j} &= f(t_j + c_s h, U_j + h \cdot (a_{s1} k_{1,j} + \cdots + a_{s,s-1} k_{s-1,j})) \\
U_{j+1} &= U_j + h \cdot (b_1 k_{1,j} + \cdots + b_s k_{s,j}).
\end{aligned}$$

This is an *s-stage explicit Runge-Kutta method* for (2.2). The coefficients c_i satisfy the condition

$$c_i = \sum_{k=1}^{i-1} a_{ik}$$

for $i \leq s$. The constant coefficients $a_{k,i}, b_k, c_k$ are chosen so that the method is of order r.

The Runge-Kutta information is given by

$$\Upsilon_{r,h}^{RK} = \{k_{i,j} : 1 \leq i \leq s,\ 0 \leq j \leq m-1\}$$

and the Runge-Kutta algorithm is given, on each interval $[x_{ir}, x_{(i+1)r}]$, by the Lagrange interpolant of degree r of the points $(t_{ir}, z_{ir}), (t_{ir+1}, z_{ir+1}), \ldots, (t_{(i+1)r}, z_{(i+1)r})$ for $0 \leq i \leq m/r - 1$.

Theorem 3 *Under the assumptions of Theorem 2:*

- *There exists a constant C_r, independent of h such that*

$$e(\alpha_{r,h}^{RK}, \Upsilon_{r,h}^{RK}) \leq C_r \cdot h^r.$$

- *The r-th order Runge-Kutta algorithm $\alpha_{r,h}^{RK}$ using the information $\Upsilon_{r,h}^{RK}$ with stepsize $h = (\varepsilon/C_r)^{1/r}$ is an almost optimal complexity algorithm for solving (2.2).*

Remark. We observe that the exponent of ε in the expression of the complexity does not depend on the dimension d of the problem. This comes from the fact that the solution u (consisting of d scalar functions, $u_1, \cdots, u_d$) depends only on one variable, t. The overall cost depends therefore only mildly (polynomially) on the dimension of the problem.

Finally, we mention that a small improvement of the complexity is obtained if the class of admissible information is that of continuous linear information (Λ^*). In the frame of the standard theory, though, the complexity remains exponential in the number of digits of accuracy, with a slightly smaller constant of proportionality of the exponent. The results for this type of information are due to Kacewicz [53, 52].

2.2 Corless's setting for ordinary differential equations

Exponential cost in the number of digits of accuracy for solving IVP for ODEs is too pessimistic a result. These problems can be solved even on slow computers. For obtaining a good, realistic model of practical computation, further refinement of the theory is needed. A key observation was made by Corless [18], who noticed that, for the problems arising in practice, the functions involved in the ODE are typically piecewise analytic, rather than functions with a fixed order of continuity.

The problem under investigation is, without loss of generality, an autonomous ordinary differential equation:

$$\begin{aligned} u'(t) &= f(u(t)) \\ u(a) &= U_0 \end{aligned} \tag{2.4}$$

where $a \leq t \leq b$.

We assume that the function f in (2.4) is piecewise analytic, with isolated breaking points. Also, the solution is assumed to exist and be unique.

The analysis is done in terms of residual error, rather than local error. However, similar results may be obtained if the residual error is replaced by forward error or local error (because asymptotically as $\varepsilon \to 0$ local error per unit step is equivalent to residual error).

For the remainder of this thesis, we shall adopt the residual error criterion, since it is both cheap to implement and easy to analyse. Let us give below a short introduction to the residual error approach in the ODE case.

Residual control of IVP for ODE

Once the approximate solution at the mesh points is obtained, it is possible to generate a proper ($\mathcal{C}^1$) extension of the numerical solution by interpolation, and therefore it is possible to compute the residual,

$$\delta(t) = u'(t) - f(u(t)) \ .$$

One useful tool in obtaining appropriate global error bounds in terms of residual is the Gröbner-Alekseev nonlinear variation-of-constants formula. Details on this and on an alternative mathematical tool for obtaining realistic error bounds can be found in [48].

Theorem 4 (Gröbner-Alekseev nonlinear variation-of-constants formula) *Let y and z be the solutions of*

$$\begin{aligned} y' &= f(t,y) & y(t_0) &= y_0 \ , \\ z' &= f(t,z) + \delta(t,z), & z(t_0) &= z_0 \ , \end{aligned} \tag{2.5}$$

respectively, and suppose that $\partial f/\partial y$ exists and is continuous. Then the solutions of the equations (2.5) are connected by

$$z(t) - y(t) = \int_{t_0}^{t} \frac{\partial y}{\partial y_0}(t, s, z(s)) \cdot \delta(s, z(s)) ds \ .$$

The proof is given in [40].

As [28] emphasizes, the use of residual error has one important practical advantage: it separates the concept of numerical stability of the approximating method from the concept of the conditioning of the problem. For a small residual and a well-conditioned problem, the corresponding global error is small.

Reliable strategies of controlling the residual are given, for example, in [30].

Consider now the local residual error coefficients, $\phi_{i,r}$, for a method of order r and for a stepsize $h_i = t_{i+1} - t_i$ to be

$$\delta_i = \phi_{i,r} h_i^r \ .$$

Definition 1 *For an arbitrary vector with positive coefficients, $\Psi = [\psi_1, \cdots, \psi_N]$, we define by*

$$\|\Psi\|_s = \left(\sum_{i=1}^{N} \psi_i^s \right)^{1/s} \tag{2.6}$$

the s-norm of a vector and by

$$\mathcal{M}_s(\Psi) = \left(\frac{1}{N} \sum_{i=1}^{N} \psi_i^s \right)^{1/s} . \tag{2.7}$$

its Hölder s-mean.

We note that $\mathcal{M}_s(\Psi) \le \|\Psi\|_\infty$ holds.

Some regularity conditions for the vector of local error coefficients, $\Phi_N = [\phi_1, \cdots, \phi_N]$ are assumed: we assume that once the mesh is sufficiently fine, the Hölder mean never increases, and the same property holds for the supremum norm of the vector of local error coefficients.

We give below a general result which is critical for optimal mesh selection.

Minimax Theorem 1 *Given $p, N \in \mathbb{N}$ and a vector with positive coefficients $[\phi_i]_{1\le i\le N}$, the following inequality holds:*

$$\max\{\phi_i h_i^p : \sum_1^N h_i = b-a\} \ge (b-a)^p \|\Phi\|_{-1/p} = \bar{h}^p \mathcal{M}_{-1/p}(\Phi)$$

where $\bar{h} = (b-a)/N$ is the average stepsize. Equality holds iff

$$\phi_i h_i^p = \bar{h}^p \mathcal{M}_{-1/p}(\Phi) \text{ for all } 1 \le i \le N.$$

Since this general result is important also for the selection of the optimal mesh for solving IVP for DAEs, we analyse it in more detail. We give below a new proof, by linearization. A proof based on Hölder's inequality can be found in [17].

Proof. The following inequality is valid for all $a_j > 0$ (see e.g. [11])

$$\max_{1\le j\le N} b_j \ge \frac{\sum_{j=1}^N b_j a_j}{\sum_{j=1}^N a_j}$$

and equality holds iff the b_j are all equal. By choosing $a_j = \phi_j^{-1/p}$ and $b_j = \phi_j^{1/p} h_j$, we derive

$$\left(\max_{1\le j\le N} \phi_j h_j^p\right)^{1/p} = \max_{1\le j\le N} \phi_j^{1/p} h_j \ge \frac{b-a}{\sum_{j=1}^N \phi_j^{-1/p}} \tag{2.8}$$

with equality iff $\phi_j h_j^p$ are all equal. We conclude by applying the power p to (2.8).

A consequence of the Minimax Theorem and the regularity conditions on the vector of local error coefficients is the generation of the optimal mesh (the one on which the tolerance is satisfied at the minimal cost). This mesh is obtained by equidistributing the (residual) error:

$$\phi_i h_i^r \approx \varepsilon \qquad \text{for all } 1 \leq i \leq N \ .$$

Theorem 5 *Under the above assumptions:*

(i) The cost of computing the solution of (2.4) to accuracy ε on an equidistributing mesh is equal or less than the cost of computing the solution with the same accuracy on a non-equidistributing mesh.

(ii) If the cost of one step with an order r method with precision ε is bounded by

$$C_f \cdot r^k B^2 \ \text{as } \varepsilon \to 0^+ \tag{2.9}$$

then we may choose r as a function of ε so that the computational cost of solving (2.4) to error ε is

$$C_f(\frac{e}{k})^k(b-a)(\mathcal{M}_{-1/r}(\Phi))^{1/r}B^{k+2}. \tag{2.10}$$

It is important to mention that the order of the method depends on the tolerance, in fact

$$r = \lceil B/k \rceil \ ,$$

and thus a numerical method of variable order should be used. Examples of methods of variable order, for which the cost of one step is of the form (2.9), are available for ODEs: Taylor series methods and Runge-Kutta methods.

Further details on the results in this section can be found in [18].

2.3 Summary

In this chapter, we introduced the basic concepts of the standard theory of information-based complexity. We revisited a controversial result of the standard theory stating that adaptive meshes are no better than non-adaptive ones.

We focused on the computational complexity of solving initial value problems for ordinary differential equations, and we studied both the standard theory and the new, improved setting given by Corless [18]. While the standard theory predicts that the complexity is exponential in the number of digits of accuracy requested, the new theory proves that the cost of numerically solving the problem is polynomial in the number of digits of accuracy. The new model is in harmony with the observed behaviour of numerical solution algorithms for ODE: that such problems are tractable and not of exponential cost.

There are several different assumptions that separate one approach from the other, but the main difference is the set of problem elements, F. For problems lying in finite-dimensional spaces, one can always choose F to be the largest possible set. But for problems in infinite-dimensional spaces (e.g., differential equations), such a choice is not possible. Since the complexity of solving the problem depends strongly on the set F, its choice is critical in obtaining a good model :

> "...for infinite-dimensional problems one cannot obtain meaningful complexity results if F is too large" - Traub and Wozniakowski, [81].

The standard theory studies the class of $\mathcal{C}^r$-continuous functions. It is this assumption about the finite order of smoothness which is responsible for the failure of the standard theory in finding a complexity in agreement with what has been observed in practical applications.

Usually, in practice, functions have isolated singularities (at a priori unknown points), and between these breaking points the function is smooth. This is exactly the aspect that has been exploited by the new theory.

Another difference is that while the standard theory is concerned with bad problems for a given mesh, the new theory finds a good mesh for a given problem.

Also, a different error criterion is used: forward error in the standard theory as opposed to residual error in the new setting. This is not a fundamental difference since these two errors are of the same order by the Gröbner-Alekseev nonlinear variation-of-constants formula.

In the new setting it can be proved that nonadaption is never less expensive than adaption in solving ordinary differential equations, which is exactly what is experienced in practical application. There are many applications for which an adaptive mesh is much better than a non-adaptive one. For example, if we wish to compute the orbit of a planet around the sun then it is less expensive if we choose the stepsize smaller when the planet is closer to the sun and larger when the planet is far from it. In order to maintain the same accuracy of the solution with a fixed step-size mesh, we need a small stepsize everywhere (to ensure accuracy near the sun), and thus the method becomes more expensive.

These two models are different views of the same problem. Both analyses are mathematically interesting, elegant and rigorous. But the value of a computational complexity result, measured by the impact on the practical computation, is achieved only if the model is realistic.

Chapter 3

Index-1 DAEs

In Chapter 2 we reviewed the complexity of numerically solving initial value problems for ordinary differential equations and we emphasized the importance of having good models of practical computation.

We discussed the recent result of Corless [18] showing that the cost of solving initial value problems for ordinary differential equations in standard form is polynomial in the number of bits of accuracy. This result is based on more realistic assumptions, such as more smoothness in between breaking points assumed for the functions that characterize the differential equations.

We show in this chapter that the cost to solve initial value problems for semi-explicit index-1 differential algebraic equations is polynomial in the number of bits of accuracy, extending thus the result of [18].

The wide class of semi-explicit index-1 DAEs constitute a first generalization of ODEs. A good complexity result for this class of problems not only has a theoretical and practical importance for the study of DAE solving. It also gives a good complexity for solving important polynomial problems, such as factoring bivariate polynomials [20] and homotopy continuation methods for finding all isolated roots of polynomial systems [2] (also see Chapter 5).

Initial value problems for differential algebraic equations are harder to solve in practice than initial value problems for ordinary differential equations. However, the results of this chapter show that the difficulty is essentially a constant factor, and not a different exponent in the theoretical complexity.

We use the residual error, or 'defect', because it makes the analysis simpler. This is equivalent to local error methods (see e.g. [72]). It is also easier to explain to users and is quite practical to implement, given that continuous extensions are needed also for other purposes [28]. The residual error is also, in the IVP for ODE case, directly connected to the global (forward) error by the Gröbner-Alekseev nonlinear variation-of-constants formula (see e.g. Hairer, Nørsett and Wanner [40]). This separates the stability of the method (producing a small residual) from the conditioning of the problem [16, 28]. The Gröbner-Alekseev formula has been extended to IVP for certain classes of DAE [30, 63].

The analysis assumes that the functions defining the differential algebraic equation are piecewise analytic. This is the key assumption, allowing the use of arbitrary order methods (in fact the chosen order $p = \lceil B/2 \rceil$ depends on the tolerance ε for the residual). In practice, events and singularities of course occur; see e.g. Moler [58]. We assume that singularities can be located accurately at negligible cost: we also assume that events are $\mathcal{O}(1)$ apart as $\varepsilon \to 0$ and that accurate location adds at most an extra factor of $\ln B$ to the cost.

We also use the fact that order-p accurate solutions may be computed on an interval of width $h \ll 1$ in $\mathcal{O}(p^2)$ operations using $\mathcal{O}(B)$ bits and therefore at cost $\mathcal{O}(p^2B^2)$. This can be done for IVP, for example, by Taylor series methods (see e.g.Corliss and Chang [25], Butcher [12], Hairer, Norsett and Wanner[40]) or Hermite-Obreschkoff methods (Butcher [12], Jackson and Nedialkov [49]) and recently it has been shown to be possible for DAE (Nedialkov and Pryce [62]).

The analysis also relies on certain regularity assumptions that ensure that the error estimates are not fooled; without those assumptions, the problem is in fact undecidable, [54].

3.1 The problem

Consider the following semi-explicit index-1 DAE:

$$\begin{aligned} y_e'(t) &= f(y_e(t), z_e(t)) && (3.1) \\ 0 &= g(y_e(t), z_e(t)) && (3.2) \end{aligned}$$

for $t \in I = (a, b)$, $y_e : I \to \mathbb{R}^k$ and $z_e : I \to \mathbb{R}^m$. Here $f : D \subset \mathbb{R}^{m+k} \to \mathbb{R}^k$ and $g : D \subset \mathbb{R}^{m+k} \to \mathbb{R}^m$ are assumed to be piecewise analytic functions on an open subset D of $\mathbb{R}^{m+k}$. We assume that the solution path lies in D. We also assume that the Jacobian $g_z(y_e, z_e)$ is invertible along the solution path. The initial condition, given by

$$y_e(a) = y_0,\ z_e(a) = z_0\ ,$$

is assumed consistent with the constraints, so

$$g(y_0, z_0) = 0\ .$$

We assume that the solution $(y_e(t), z_e(t))$ exists and is unique on the interval of integration. We also assume that if we augment the DAE with extra ODEs that describe standard functions used in f and g, then the problem is converted to a larger DAE in which the new functions f and g involve only the four basic arithmetic operations. Thus f and g in (3.1)-(3.2) belong to the class of functions to which automatic differentiation applies. This includes most functions of practical interest, but excludes, for example, the Γ function.

Other assumptions (detailed later) ensure that the minimum stepsize is bounded away from zero, and thus the integration does not 'grind to a halt'.

We are interested in computing a numerical solution with a tolerance ε for both the residual error of the differential equation $\|\delta_1(t)\|$ and the residual error of the algebraic equation $\|\delta_2(t)\|$, with

$$\delta_1(t) = y'(t) - f(y(t), z(t)) \tag{3.3}$$

$$\delta_2(t) = g(y(t), z(t)) \ . \tag{3.4}$$

3.1.1 Residual error control vs. forward error control

By controlling the magnitude of the residual we also control the forward error. We will describe how the relationship between residual and forward error can be obtained by using Gronwall's lemma for the underlying ODE (see also [41]). For an alternative approach using the Gröbner-Alekseev formula see [63].

We first assume that the functions f and g are analytic in a neighbourhood of the exact solution path. Since g_z is invertible, by the implicit function theorem there exists a unique solution of $g(y(t), z(t)) = \delta_2(t)$, given by

$$z(t) = G(y(t), \delta_2(t)) \ .$$

Therefore, if we denote by

$$y_s(t) = y_e(t) + s(y(t) - y_e(t)) \ ,$$

we derive

$$\begin{aligned}
&z(t) - z_e(t) = \\
&-\int_0^1 g_z^{-1}(y_s(t), G(y_s(t), s\delta_2(t))g_y(y_s(t), G(y_s(t), s\delta_2(t))ds \cdot (y(t) - y_e(t)) \\
&+\int_0^1 g_z^{-1}(y_s(t), G(y_s(t), s\delta_2(t))ds \cdot \delta_2(t)\ .
\end{aligned}$$

By a compactness and a continuity argument, one can find a neighbourhood of the exact solution path on which both g_z^{-1} and $g_z^{-1}g_y$ are bounded. It follows that

$$\|z(t) - z_e(t)\| \leq \ell_1\|y(t) - y_e(t)\| + \ell_2\|\delta_2(t)\| \tag{3.5}$$

where we can choose, e.g., $\ell_1 = \sup\|g_z^{-1}g_y\|$, $\ell_2 = \sup\|g_z^{-1}\|$ and the suprema is over that neighbourhood.

Using a smoothness and a compactness argument for f and an integral formula for the Taylor remainder, we obtain first that

$$\begin{aligned}
&f(y,z) - f(y_e, z_e) = \\
&-\int_0^1 f_y(y_e(t) + s(y(t) - y_e(t)), z_e(t) + s(z(t) - z_e(t)))ds \cdot (y(t) - y_e(t)) \\
&+\int_0^1 f_z(y_e(t) + s(y(t) - y_e(t)), z_e(t) + s(z(t) - z_e(t)))ds \cdot (z(t) - z_e(t))
\end{aligned}$$

and then derive that there exist some positive constants L_1 and L_2 so that

$$\|f(y,z) - f(y_e, z_e)\| \leq L_1\|y - y_e\| + L_2\|z - z_e\|\ . \tag{3.6}$$

We can choose, e.g., $L_1 = \sup\|f_y\|$ and $L_2 = \sup\|f_z\|$ over an appropriate neighbourhood (the domain D is assumed convex). We denote the forward error in the differential variable by

$$x(t) = \|y(t) - y_e(t)\|\ .$$

At the first step, we subtract the differential equations from the systems (3.3) and (3.1) and we integrate the result on the interval $[0, t]$. At the next step, we apply the norm to the result and finally we use the inequalities (3.5)-(3.6). We derive successively

$$\begin{aligned} x(t) &\leq x(0) + \int_a^t \|f(y(s),z(s)) - f(y_e(s),z_e(s))\|ds + \|\int_a^t \delta_1(s)ds\| \\ &\leq x(0) + L_2\ell_2 \int_a^t \|\delta_2(s)\|ds + \|\int_a^t \delta_1(s)ds\| + (L_1 + L_2\ell_1)\int_a^t x(s)ds \\ &\leq \omega(t) + k_1\int_a^t x(s)ds \ , \end{aligned}$$

where $k_1 = L_1 + L_2\ell_1,\ k_2 = L_2\ell_2$ and

$$\omega(t) = x(0) + k_2\int_a^t \|\delta_2(s)\|ds + \int_a^t \|\delta_1(s)\|ds \ .$$

Note that ω is an increasing function. By applying Gronwall's lemma (see Appendix) on the compact interval $[a,b]$, we obtain that

$$x(t) \leq \omega(t) + k_1\int_a^t \omega(s)e^{k_1(t-s)}ds \leq e^{k_1(t-a)}\omega(t)$$

for all $t \in [a,b]$. Finally, if we take

$$K = e^{k_1(b-a)}\max\left(1, b-a, k_2(b-a)\right)$$

we find that

$$\|y(t) - y_e(t)\| \leq K\left(\|y(a) - y_0\| + \max_{a\leq s\leq t}\|\delta_2(s)\| + \max_{a\leq s\leq t}\|\delta_1(s)\|\right) \ . \tag{3.7}$$

It remains now to analyse the case when the functions f and g are piecewise analytic. This case has not been treated before, as far as we know. We assume that, in a neighbourhood of the solution, f is Lipschitz continuous in its arguments and $g_y,\ g_z^{-1}$ are bounded in norm on each region where the functions have one analytic formula. We also assume that the boundaries of these regions are continuous and piecewise analytic. Under these assumptions analogous results to (3.5) and (3.7) hold.

It is sufficient to analyse the behaviour of the forward errors when crossing one surface of separation between regions where f and g have one analytic formula. For more such crossings, induction applies.

Assume that the exact and the approximate solutions cross a surface $\mathcal{G}(t, y, z) = 0$ at times T_e and T, respectively (or equivalently, we say that the events occur at these times):

$$\begin{aligned} \mathcal{G}(t, y_e(T_e), z_e(T_e)) &= 0 \\ \mathcal{G}(t, y(T), z(T)) &= 0 \end{aligned}$$

The first step is to estimate the change in the time at which the event occurs, $|T_e - T|$. The type of arguments we use are specific to event location problems [73]. Without loss of generality, we may assume that $T < T_e$ (for the other case the analysis is similar).

We assume that the solutions obliquely cross the surface of separation (i.e. they do not slide along it),

$$\frac{d}{dt}\mathcal{G}(T_e, y_e(T_e), z_e(T_e)) = \mathcal{K} \neq 0 \ .$$

We estimate now the variation in the time when the event occurs. Since $\mathcal{G}_y$ and $\mathcal{G}_z$ are assumed to be bounded on a neighbourhood of the exact solution

$$\begin{aligned} 0 &= \mathcal{G}(T, y(T), z(T)) = \mathcal{G}(T, y_e(T), z_e(T)) + \mathcal{O}(y(T) - y_e(T)) + \mathcal{O}(z(T) - z_e(T)) \\ &= \mathcal{G}(T_e, y_e(T_e), z_e(T_e)) + \mathcal{G}'(t, y_e(T_e), z_e(T_e))(T - T_e) + \mathcal{O}((T - T_e)^2) \\ &\quad + \mathcal{O}(y(T) - y_e(T)) + \mathcal{O}(z(T) - z_e(T)) \end{aligned}$$

and $\mathcal{K} \neq 0$, then

$$|T - T_e| \leq k_3 \left(\|y(T) - y_e(T)\| + \|z(T) - z_e(T)\| \right)$$

where k_3 is a positive constant.

At the next step, we remark that, under our assumptions, the formulae (3.5) and (3.7) hold for $t \leq T$ and in particular for $t = T$. We obtain

$$|T - T_e| \leq k_4(\|y(a) - y_0\| + \max_{a\leq s\leq T} \|\delta_2(s)\| + \max_{a\leq s\leq T} \|\delta_1(s)\|) \tag{3.8}$$

On the other hand, using the triangle inequality, Taylor's formula and the equations (3.1)-(3.2) and (3.3)-(3.4), we derive

$$\begin{aligned}
\|y(T_e) - y_e(T_e)\| &\leq \|y(T_e) - y(T)\| + \|y_e(T_e) - y_e(T)\| + \|y(T) - y_e(T)\| \\
&\leq \left(\max_{t\in[T,T_e]} \|y'(t)\| + \max_{t\in[T,T_e]} \|y_e'(t)\|\right) |T_e - T| + \|y(T) - y_e(T)\| \\
&\leq \left(\max_{t\in[T,T_e]} \|f(y(t), z(t))\| + \max_{t\in[T,T_e]} \|f(y_e(t), z_e(t))\| \right. \\
&\quad \left. + \max_{t\in[T,T_e]} \|\delta_1(t)\|\right) |T_e - T| + \|y(T) - y_e(T)\|
\end{aligned}$$

which, by (3.7) and (3.8), becomes

$$\|y(T_e) - y_e(T_e)\| \leq k_5(\|y(a) - y_0\| + \max_{a\leq s\leq T} \|\delta_2(s)\| + \max_{a\leq s\leq T} \|\delta_1(s)\|) \tag{3.9}$$

where k_5 is some constant.

In the new region of analyticity (following the surface of separation), we can analogously prove the results from the case of globally analytic functions, but with the initial point $t = T_e$. The new initial condition satisfies (3.9).

We conclude by noticing that z, z_e satisfy (3.5) with some new constants, $\bar{\ell}_1$ $\bar{\ell}_2$, while, due to (3.9) and our assumptions, there exist positive constants $\bar{K}$ and k_6:

$$\begin{aligned}
\|y(t) - y_e(t)\| &\leq \bar{K}(\|y(T_e) - y_e(T_e)\| + \max_{T_e\leq s\leq t} \|\delta_2(s)\| + \max_{T_e\leq s\leq t} \|\delta_1(s)\|) \\
&\leq k_6(\|y(a) - y_0\| + \max_{s\in[a,T]\cup[T_e,t]} \|\delta_2(s)\| + \max_{s\in[a,T]\cup[T_e,t]} \|\delta_1(s)\|)
\end{aligned}$$

for $t > T_e$.

3.2 Numerical solution

In this chapter, we consider the Taylor series method developed by Pryce in [66] and [65], and we analyse it for the fixed order case.

More precisely, we shall analyse the cost of the following algorithm: assume we have obtained at time t_n the values (y_n, z_n) which satisfy the algebraic constraints more accurately than the desired tolerance. After generating the Taylor coefficients for the unknowns, we predict the values $(y_{n+1}, \hat{z}_{n+1})$ by computing the Taylor series with a chosen stepsize h_n. Then we correct $\hat{z}_{n+1}$ by applying one Newton iteration for the algebraic variables which ensures asymptotically that the algebraic constraints are satisfied more accurately. If y_{n+1} and the new value for z, namely z_{n+1}, satisfy the differential equation with the residual below the tolerance ε then the step is accepted.

3.2.1 Solution by Taylor series

The method proposed by Pryce in [66] and [65] for solving a large class of DAEs consists of generating the Taylor coefficients for the differential and algebraic equations and equating them to zero to solve for the Taylor coefficients of the variables y and z.

Pryce's method starts with a pre-processing stage which reveals in some cases a certain structure of the problem. In the cases when it is successful, the structure may be used for analysing the DAE: the structural index and the degrees of freedom for the solution manifold can be computed based on it. Also, once the structure is known, the system of the Taylor coefficients can be solved automatically.

The first step of the analysis consists in determining a set of integers, called *offsets* of the problem, that indicate which equations to solve for which unknowns. The next step is to generate the system Jacobian. If at each integration step, the system Jacobian is non-singular, the method succeeds, and the Taylor coefficients can be determined up to the desired order.

A critical observation is that at each integration step the current Jacobian J, once computed, generates after a few initial stages the Taylor coefficients for the unknowns iteratively, as solutions of some linear systems which have the same matrix J.

For a detailed presentation of Pryce's method we refer the reader to Chapter 4. For the particular case of semi-explicit index-1 DAE the results are given below. The offsets of the problem and the system Jacobian can be easily obtained to be the following vectors with $k+m$ components: $d = (1, \cdots 1, 0, \cdots, 0)$ (with the first k components 1) and $c = (0, \cdots, 0)$.

The system Jacobian is given by:

$$J = \begin{bmatrix} I & -f_z \\ 0 & g_z \end{bmatrix}$$

and it is non-singular in a neighbourhood of the solution where g_z is invertible (a necessary condition for the DAE (3.1)-(3.2) to be of index 1). This implies that the Taylor series method is successful for this problem.

Assume that the Taylor series for the equations and for the unknowns calculated at step n of integration and at time $t = t_n + h$ are, respectively,

$$\begin{aligned} y(t) &= \sum_{j\geq 0} y_{n,j}(t-t_n)^j/j! \\ z(t) &= \sum_{j\geq 0} z_{n,j}(t-t_n)^j/j! \\ F(t) &= y'(t) - f(y(t), z(t)) = \sum_{j\geq 0} F_{n,j}(t-t_n)^j/j! \\ G(t) &= g(y(t), z(t)) = \sum_{j\geq 0} G_{n,j}(t-t_n)^j/j! \ . \end{aligned}$$

According to Pryce's method we have to solve, for each stage $j = 0,\ 1,\ \cdots,\ p-1$, the systems

$$(F_{n,j} = 0, G_{n,j} = 0) \text{ in the unknowns } (y_{n,j+1}, z_{n,j}) \ .$$

The value $y_{n,0}$ is given either from the initial condition for the first integration step or from the previous step for the other steps.

The stage $j = 0$ is atypical. The equations ($F_{n,0} = 0,\ G_{n,0} = 0$) may be nonlinear in the corresponding unknown $z_{n,0}$. The value $z_{n,0}$ should be also given from the initial condition at the beginning or from the value at the previous step. Since one Newton correction is applied at the end of the previous step, we shall see that the condition $g(y_{n,0}, z_{n,0}) \approx 0$ will be satisfied accurately enough so that its residual does not essentially perturb the algebraic residual for the current integration step. The stage is completed by taking

$$y_{n,1} = f(y_{n,0}, z_{n,0}) \ . \tag{3.10}$$

The stages $j \geq 1$ are all linear in the corresponding unknowns and involve the same Jacobian $J(y_{n,0}, z_{n,0})$.

The approximate solution, given by

$$\begin{aligned} \hat{y}(t) &= \sum_{j=0}^{p} y_{n,j}(t-t_n)^j/j! \\ \hat{z}(t) &= \sum_{j=0}^{p-1} z_{n,j}(t-t_n)^j/j! \ , \end{aligned} \tag{3.11}$$

satisfies the problem

$$\begin{aligned} \hat{y}'(t) &= f(\hat{y}(t), \hat{z}(t)) + \hat{\delta}_1(t) \\ 0 &= g(\hat{y}(t), \hat{z}(t)) + \hat{\delta}_2(t) \ . \end{aligned} \tag{3.12}$$

on each interval (t_n, t_{n+1}).

The predicted values at time $t_{n+1} = t_n + h_n$ are

$$\begin{aligned} y_{n+1} &= \hat{y}(t_{n+1}^-) \ , \\ \hat{z}_{n+1} &= \hat{z}(t_{n+1}^-) \ . \end{aligned}$$

The algebraic residual after the Newton correction at the step $(n-1)$ equals the algebraic residual for the stage $j = 0$ at the nth integration step. In the next section, we will prove that this residual is $\mathcal{O}(h_{n-1}^{2p})$. This error is assumed negligible with respect to h_n^p, which is a reasonable approximation for small enough tolerance. Consequently, the residuals are

$$\begin{aligned}\hat{\delta}_1 &= \phi_{1,n} h_n^p \ , \\ \hat{\delta}_2 &= \phi_{2,n} h_n^p \ .\end{aligned}$$

An asymptotic evaluation for small h_n gives

$$\phi_{1,n} = \frac{1}{p!}\frac{d^p}{dt^p} f(y,z)(t_n) + \mathcal{O}(h_n) \ .$$

Thus the local error coefficients can be represented asymptotically as the value at (y_n, z_n) of some fixed function. That function involves multiplication, addition or subtraction of f, g, their derivatives with respect to y and z up to order p and g_z^{-1} and depends only on p. A similar expression can be obtained for $\phi_{2,n}$.

3.2.2 Newton projection

We wish to satisfy the algebraic equations more accurately than the differential equations, but we do not want to increase the cost unnecessarily. It has been observed in practice [60] that after one Newton iteration the algebraic constraints become sufficiently accurate and this agreement improves as the tolerance $\varepsilon \to 0$.

Since g is an analytic function in its variables and g_z is non-singular for the exact solution then g_z is non-singular in a neighbourhood of the solution, and we can consider a simplified Newton iteration. Note that all the neighbourhoods will be considered with respect to the exact solution (y_e, z_e, y'_e) in $\mathbb{R}^{2k+m}$.

If the predictor step is given by $(y_n, \hat{z}_n)$, we keep y_n constant and we correct only the algebraic variable, by considering a new value, z_n, which satisfies

$$z_n = \hat{z}_n - g_z^{-1}(y_n, \hat{z}_n) g(y_n, \hat{z}_n) \ . \tag{3.13}$$

From (3.13) we obtain that

$$\|z_n - \hat{z}_n\| \leq L\|g(y_n, \hat{z}_n)\| \tag{3.14}$$

where $L = \sup\limits_U \|g_z^{-1}\|$ and U is an open neighbourhood of the exact solution on which g_z is invertible and g_z^{-1} and g_{zz} are bounded.

Consider

$$H(s) = g(y_n, \hat{z}_n + s(z_n - \hat{z}_n)) \ .$$

By integrating the function $H''(s)(1-s)$ with respect to s on the interval $[0,1]$, we derive the following Taylor expansion:

$$g(y_n, z_n) = g(y_n, \hat{z}_n) + g_z(y_n, \hat{z}_n)(z_n - \hat{z}_n) + \frac{1}{2}(z_n - \hat{z}_n)^T G(y_n, z_n, \hat{z}_n)(z_n - \hat{z}_n)$$

where

$$G(y_n, z_n, \hat{z}_n) = \int_0^1 2(1-s) g_{zz}(y_n, \hat{z}_n + s(z_n - \hat{z}_n)) ds \ .$$

The first two terms in the Taylor expansion cancel each other due to (3.13), and by using (3.14) we obtain

$$\|g(y_n, z_n)\| \leq \frac{1}{2} M L^2 \|g(y_n, \hat{z}_n)\|^2 \ , \tag{3.15}$$

where $M = \sup\limits_U \|g_{zz}\|$.

We conclude the analysis of the Newton projection by observing that if the tolerance ε is sufficiently small then the residual is small enough to guarantee that $(y_n, \hat{z}_n)$ is in the basin of attraction of the Newton method.

3.2.3 Dense output

We construct below a continuous interpolant of the solution. Let P be the interpolant of the differential variable, y, and Q the interpolant corresponding to the algebraic variable, z.

We note that, in order to have continuous residuals, P should be at least differentiable at the mesh points, (y_n, z_n), while Q should be at least continuous. Useful interpolants for our problem are the Hermite interpolants, which satisfy on each interval $[t_n, t_{n+1}]$ the following conditions

$$\begin{aligned} P^{(l)}(t_n) &= y_{n,l} \text{ for all } 0 \le l \le p \\ P^{(l)}(t_{n+1}) &= y_{n+1,l} \text{ for all } 0 \le l \le 1 \\ Q^{(l)}(t_n) &= z_{n,l} \text{ for all } 0 \le l \le p-1 \\ Q(t_{n+1}) &= z_{n+1} \ . \end{aligned} \tag{3.16}$$

We denote the jumps in values of the variables and some of their first order derivatives (at time t_{n+1}, but we shall omit this in our notation) by

$$\begin{aligned} \delta y_0 &= y_{n+1}(t_{n+1}^+) - y_n(t_{n+1}^-) \\ \delta y_1 &= \frac{d}{dt} y_{n+1}(t_{n+1}^+) - \frac{d}{dt} y_n(t_{n+1}^-) \\ \delta z_0 &= z_{n+1}(t_{n+1}^+) - z_n(t_{n+1}^-) \ . \end{aligned}$$

It is convenient to consider the following basis for the Hermite interpolants P and Q, respectively

$$\begin{aligned} &1,\ t-t_n,\ ,(t-t_n)^2, \ldots,\ (t-t_n)^p,\ (t-t_n)^{p+1},\ (t-t_n)^{p+1}(t-t_{n+1}) \\ &1,\ t-t_n,\ ,(t-t_n)^2, \ldots,\ (t-t_n)^p \end{aligned}$$

By applying the conditions (3.16), it follows that the Hermite interpolants on each interval $[t_n, t_{n+1}]$ are given by

$$\begin{aligned} P(t) &= \hat{y}(t) + \delta y_0 \cdot \left(\frac{t-t_n}{h_n}\right)^{p+1} + \delta y_1 \cdot \left(\frac{t-t_n}{h_n}\right)^{p+1} (t - t_{n+1}) \\ Q(t) &= \hat{z}(t) + \delta z_0 \cdot \left(\frac{t-t_n}{h_n}\right)^{p} \end{aligned} \tag{3.17}$$

The algorithm ensures continuity in the differential variable at the end of the step, thus

$$\delta y_0 = 0 \ . \tag{3.18}$$

By using the formula for the Newton projection, (3.13), and the second equation in (3.12), we derive the jump in the algebraic variable at t_{n+1}

$$\delta z_0 = g_z^{-1}(y_{n+1}, \hat{z}_{n+1})\hat{\delta}_2(t_{n+1}^-) = g_z^{-1}(y_{n+1}, \hat{z}_{n+1})\phi_{1,n}h_n^p \ . \tag{3.19}$$

Finally, from formulae (3.10), (3.12) and (3.19), we compute the jump in the derivative of the differential variable at the end of the step

$$\begin{aligned} \delta y_1 &= f(y_{n+1}, z_{n+1}) - f(y_{n+1}, \hat{z}_{n+1}) - \hat{\delta}_1(t_{n+1}^-) \\ &= f_z(y_{n+1}, \hat{z}_{n+1}) \cdot \delta z_0 - \hat{\delta}_1(t_{n+1}^-) + \mathcal{O}((\hat{\delta} z_0)^2) \\ &= f_z(y_{n+1}, \hat{z}_{n+1}) \cdot g_z^{-1}(y_{n+1}, \hat{z}_{n+1})\hat{\delta}_2(t_{n+1}^-) - \hat{\delta}_1(t_{n+1}^-) + \mathcal{O}((\hat{\delta} z_0)^2) \\ &= \left[f_z(y_{n+1}, \hat{z}_{n+1}) \cdot g_z^{-1}(y_{n+1}, \hat{z}_{n+1})\phi_{2,n} - \phi_{1,n}\right] h_n^p + \mathcal{O}(h_n^{2p}) \end{aligned} \tag{3.20}$$

We make the following notation in terms of the error coefficients

$$\|\delta y_1\| = \psi_n h_n^p \tag{3.21}$$

and we remark that

$$\psi_n = f_z(y_{n+1}, \hat{z}_{n+1}) \cdot g_z^{-1}(y_{n+1}, \hat{z}_{n+1})\phi_{2,n} - \phi_{1,n} + \mathcal{O}(h_n^p) \ .$$

Our aim is to show that this error dominates the residuals in both the differential and the algebraic equations, as we shall see in the next section.

3.2.4 Error Analysis

We give below estimates for the residual errors obtained for the chosen interpolant. These estimates will help us establish which errors dominate the residuals and should therefore be controlled.

First, we estimate the residual in the algebraic equation at time t_{n+1}, after the corrector step, given by $\|\delta_2(t_{n+1})\| = \|g(y_{n+1}, z_{n+1})\|$. Using (3.12) and (3.15), we obtain

$$\|\delta_2(t_{n+1})\| \leq \frac{1}{2} M L^2 \|\phi_{1,n}\|^2 h_n^{2p} \ .$$

As was mentioned before, this result guarantees that the integration can be safely carried to the next step, as the algebraic residual at the beginning of the next step is small enough and consequently produces a small perturbation.

The differential and the algebraic residual corresponding to the interpolants (3.17) are, respectively,

$$\begin{aligned} \Delta_1(t) &= P'(t) - f(P(t), Q(t)) \\ \Delta_2(t) &= -g(P(t), Q(t)) \ . \end{aligned} \tag{3.22}$$

Let $\tau = (t - t_n)/h_n$, where $\tau \in [0, 1]$.

In order to estimate the residuals corresponding to our choice of interpolants, we apply the estimates (3.18), (3.19) and (3.20) to the equations (3.17), and substitute the latter into (3.22). We also use the formulae for the Taylor residuals (3.12) and successively find for the differential residual

$$\begin{aligned} \Delta_1(t) &= \hat{y}'(t) - f(\hat{y}(t) + \mathcal{O}(h_n^{p+1}), \hat{z}(t) + \delta z_0 \left(\frac{t - t_n}{h_n}\right)^p) \\ &\quad + \delta y_1 \left[\left(\frac{t - t_n}{h_n}\right)^{p+1} + (p+1) \left(\frac{t - t_n}{h_n}\right)^p \left(\frac{t - t_{p+1}}{h_n}\right) \right] \\ &= \hat{\delta}_1(t) + f(\hat{y}(t), \hat{z}(t)) - f(\hat{y}(t), \hat{z}(t) + \delta z_0 \left(\frac{t - t_n}{h_n}\right)^p) + \delta y_1 \sigma_1(\tau) + \mathcal{O}(h_n^{p+1}) \\ &= \left[\hat{\delta}_1(t) - f_z(y_{n+1}, \hat{z}_{n+1}) \cdot \delta z_0 \left(\frac{t - t_n}{h_n}\right)^p \right] \\ &\quad + \left[(f_z(y_{n+1}, \hat{z}_{n+1}) - f_z(\hat{y}(t), \hat{z}(t)) \cdot \delta z_0 \left(\frac{t - t_n}{h_n}\right)^p \right] + \delta y_1 \sigma_1(\tau) + \mathcal{O}(h_n^{p+1}) \end{aligned}$$

where

$$\sigma_1(\tau) = \tau^{p+1} + (p+1)\tau^p(\tau - 1) \ .$$

Here $\sigma_1(\tau)$ is a polynomial independent of the integration step and of the problem itself. In fact, it only depends only on p.

We note that, with a smoothness argument,

$$f_z(y_{n+1}, \hat{z}_{n+1}) - f_z(\hat{y}(t), \hat{z}(t) = \mathcal{O}(h_n) \ .$$

Substituting this and $f_z(y_{n+1}, \hat{z}_{n+1}) \cdot \delta z_0 = \delta y_1 + \hat{\delta}_1(t_{n+1}^-) + \mathcal{O}(h_n^{2p})$ (from (3.20)), we can further obtain for the differential residual

$$\Delta_1(t) = \delta y_1 \sigma_2(\tau) + \hat{\delta}_1(t) - \hat{\delta}_1(t_{n+1}^-)\left(\frac{t-t_n}{h_n}\right)^p + \mathcal{O}(h_n^{p+1})$$

with

$$\sigma_2(\tau) = \sigma_1(\tau) + \tau^p \ .$$

At each integration step there exists a constant c_1 such that the Taylor residual satisfies $\hat{\delta}_1(t) = c_1(t-t_n)^p + \mathcal{O}((t-t_n)^{p+1})$, due to the choice of the Taylor approximate solution (3.11). It easily follows that

$$\hat{\delta}_1(t) - \hat{\delta}_1(t_{n+1}^-)\left(\frac{t-t_n}{h_n}\right)^p = \mathcal{O}(h_n^{p+1}) \ .$$

Consequently, the differential residual is dominated by

$$\|\Delta_1(t)\| \leq \gamma \cdot \psi_n h_n^p + \mathcal{O}(h_n^{p+1}) \tag{3.23}$$

where

$$\gamma = \max_{\tau \in [0,1]} |\sigma_2(\tau)|$$

is independent of the integration step.

Similarly, we obtain for the algebraic residual

$$\begin{aligned}\Delta_2(t) &= -g(\hat{y}(t) + \mathcal{O}(h_n^{p+1}), \hat{z}(t) + \delta z_0 \left(\frac{t-t_n}{h_n}\right)^p) \\ &= -g(\hat{y}(t), \hat{z}(t)) - g_z(\hat{y}(t), \hat{z}(t)) \cdot \delta z_0 \left(\frac{t-t_n}{h_n}\right)^p + \mathcal{O}(h_n^{p+1}) \ .\end{aligned}$$

By substituting (3.19) and (3.12) into the expression above we get

$$\begin{aligned}\Delta_2(t) &= \hat{\delta}_2(t) - g_z(\hat{y}(t), \hat{z}(t)) g_z^{-1}(\hat{y}(t_{n+1}^-), \hat{z}(t_{n+1}^-))\hat{\delta}_2(t_{n+1}^-)\left(\frac{t-t_n}{h_n}\right)^p + \mathcal{O}(h_n^{p+1}) \\ &= \hat{\delta}_2(t) - (I + \mathcal{O}(h_n))\hat{\delta}_2(t_{n+1}^-)\left(\frac{t-t_n}{h_n}\right)^p + \mathcal{O}(h_n^{p+1}) \\ &= \hat{\delta}_2(t) - \hat{\delta}_2(t_{n+1}^-)\left(\frac{t-t_n}{h_n}\right)^p + \mathcal{O}(h_n^{p+1}) \ .\end{aligned}$$

An argument similar to that for the Taylor differential residual applies also for the Taylor algebraic residual, which is of the same order as the Taylor differential residual. We obtain

$$\hat{\delta}_2(t) - \hat{\delta}_2(t_{n+1}^-)\left(\frac{t-t_n}{h_n}\right)^p = \mathcal{O}(h_n^{p+1}) \ .$$

Therefore, we easily see that

$$\Delta_2(t) = \mathcal{O}(h_n^{p+1}) \ . \tag{3.24}$$

Remark. The behaviour (3.24) for the algebraic residual is due to our choice of the projection, (3.13). If instead of the projection on only the algebraic constraint, we choose, for example, a projection in (y_1, z_0) on both the algebraic and the differential constraints, then both algebraic and differential residuals are $\mathcal{O}(h_n^p)$ and should be controlled. This will change the complexity result by at most a (constant) factor depending on the new error coefficients.

We conclude by saying that, for small enough tolerance, it is enough to control just the differential residual, which is $\mathcal{O}(h_n^p)$. We require that

$$\psi_n h_n^p \leq \varepsilon \text{ for all } n \leq N \ .$$

We note that for problems 'near' to a higher index problem, e.g. if g_z is ill-conditioned, then 'small enough tolerance' may be impractically small. This is an indicator of stiffness, which we hope to examine in future works.

3.3 The problem is tractable

3.3.1 The minimum cost of the algorithm

We claim that the minimum cost to find an approximate solution that satisfies the desired accuracy with the algorithm described above is achieved if the residual error is equidistributed. In order to see this, we need a result from Corless [18], which we reproduce below.

For a vector $\Psi = [\psi_1, \cdots, \psi_N]$ we denote by $\|\Psi\|_s$ the *s-norm* defined by formula (2.6) and by $\mathcal{M}_s(\Psi)$ the *Hölder s-mean* given by (2.7).

Note that the s-norm is not, in fact, a true norm for $s < 1$, because then the triangle inequality is not always satisfied. We shall not need this property.

We assume that the vector of the error coefficients $\Psi_N = [\psi_1, \cdots, \psi_N]$ satisfies some regularity conditions. The first assumption is that there exists a positive integer N_0 such that

$$\mathcal{M}_s(\Psi_{N_2}) \leq \mathcal{M}_s(\Psi_{N_1}) \tag{3.25}$$

for any $N_2 > N_1 \geq N_0$. This means that if the mesh is sufficiently fine, the Hölder mean does not increase with further refinement. The second assumption is that the same property holds for the maximum norm of the local error coefficients,

$$\|\Phi_{N_2}\|_\infty \leq \|\Phi_{N_1}\|_\infty \tag{3.26}$$

for $N_2 > N_1 \geq N_0$. What we are really assuming is that the implementation can do this in practice.

As in the standard theory of computational complexity [84], we ignore memory hierarchy, overheads and interpolation costs. Though, while the standard theory considers the number of function evaluations as the main contributor to the cost, we measure the cost of the algorithm by the number of arithmetic operations.

> "One usual measure of efficiency, the counting of function evaluations, is not appropriate for series methods " - Corliss, Chang, [25]

the authors proposing instead the CPU execution time as adequate means of comparison for Taylor series methods. We adopt the latter view, this being yet another difference between our model and the standard theory model.

The cost per step for this method with a fixed order is the same for all steps and consists of the cost to obtain the Taylor series plus the cost of one Newton projection. The cost of computing a solution increases then with the number of steps to go from $t = a$ to $t = b$.

Definition 2 *A mesh is called optimal if it takes the minimum number of steps to go from a to b while satisfying the tolerance.*

By analogy with [18], we obtain the following consequences of the Minimax Theorem (see Chapter 2) and of the regularity conditions assumed for the vector of local error coefficients, (3.25) and (3.26):

1. The solution of the Minimax theorem gives also the optimal mesh for the algorithm considered in this chapter

$$\psi_i h_i^p = \bar{h}^p \mathcal{M}_{-1/p}(\Psi) = \varepsilon \text{ for all } 1 \leq i \leq N$$

and the corresponding total number of steps

$$N = \frac{b-a}{\bar{h}} = (b-a) \cdot (\mathcal{M}_{-1/p}(\Psi_N))^{1/p} \varepsilon^{-1/p} \ . \tag{3.27}$$

2. Computing the solution of (3.1)-(3.2) on a fixed step size mesh with an error tolerance ε is more expensive than computing the solution with the same tolerance on an equidistributed mesh for the algorithm analysed. The cost for a fixed step size mesh is greater than the cost on an equidistributed mesh by a factor of

$$\|\Psi_{fix}\|_\infty / (\mathcal{M}_{-1/p}(\Psi_{ad}))^{1/p} \ .$$

This factor is asymptotic to 1 as $p \to \infty$, but for realistic values of p may be quite substantial.

Remark. Equidistribution ensures that the mean stepsize controls the integration. The minimum stepsize is related to the maximum ψ_n and is assumed bounded here, which ensures that the integration does not grind to a halt.

Theorem 6 *For sufficiently small ε, the minimal cost of obtaining the solution with error ε of the IVP for (3.1)-(3.2) using Pryce's method is polynomial in the number of digits of accuracy. It is reached on the equidistributing mesh and is bounded above by*

$$C(b-a)\frac{e^2}{4} \cdot (\mathcal{M}_{-1/p}(\Psi_N))^{1/p} B^4. \tag{3.28}$$

If the Hölder p-mean of the local error coefficients depends weakly on the order p of the method, then the minimum is given by (3.28) and is reached for $p = \lceil B/2 \rceil$.

Proof. Computing the solution with Taylor series to order p and precision ε for one step takes $C_1 p^2 B^2$ operations. Indeed, the cost of generating and computing the Taylor series with p terms using automatic differentiation is $\mathcal{O}(p^2)$ (see Corliss and Chang [25]). Also, doing arithmetic with precision ε, that is with B digits of accuracy, costs $\mathcal{O}(B^2)$ if the naive algorithm for multiplication is used. The constant C_1 depends on the dimension of the problem (k and m).

The cost of the Newton projection is independent of p and is $\mathcal{O}(B^2)$, with the constant depending on the dimension of the problem.

Thus, overall, the cost of one step is Cp^2B^2. By using (3.27), we obtain the total cost (the number of steps times the cost for one step) on the equidistributed mesh

$$C(b-a) \cdot (\mathcal{M}_{-1/p}(\Psi_N))^{1/p} B^2 p^2 \varepsilon^{-1/p} \ . \tag{3.29}$$

Under the hypothesis on the local error coefficients, we find that the total cost

(3.29) reaches a minimum for the value $p = B/2$, given by

$$C(b-a)\frac{e^2}{4} \cdot (\mathcal{M}_{-1/p}(\Psi_N))^{1/p} B^4$$

Since the Taylor series method allows a variable order, we can choose

$$p = \lceil B/2 \rceil \ .$$

If the order is $p = \lceil B/2 \rceil$ and if the dependence of the Hölder mean of the local error coefficients on p is not weak, then (3.28) gives an upper bound for the minimal cost.

Remarks. The cost of computing the solution on a fixed step size mesh is also polynomial in number of digits of accuracy. It is obtained by replacing the Hölder mean norm by the infinity norm in (3.28).

By using asymptotically faster multiplication (e.g. FFT methods, [9]) we may reduce the bound on the minimum cost to $\mathcal{O}\left(B^3 \ln B\right)$, ignoring factors of $(\ln B)^2$ from event location.

3.4 Summary

We have obtained that the cost of approximating the solution of the semi-explicit index-1 DAE (3.1)-(3.2) with the Taylor series method developed by Pryce is polynomial in the number of digits of accuracy. We have shown that nonadaption is more expensive than adaption depending on the ratio of the maximum norm of the vector of local error coefficients to their Hölder mean.

Our results closely reflect the experience of practitioners of scientific computation. Practical implementation of the method has given excellent results in applications requiring very high precision, [4] and [60, 61]. This high precision can not be obtained using low order methods, as predicted by our results.

The key assumption which led to polynomial cost was that the problem is assumed smooth between the break points (which are assumed to be located automatically).

We believe that the theoretical study of the complexity of DAE solving sheds light on finite precision and finite order methods for such problems.

Chapter 4

High-index DAEs

In two previous chapters, we studied the theoretical (asymptotic) complexity, as the accuracy of the computed residual gets better, of the numerical solution of the initial value problem for ordinary differential equations following Corless [18] and for index - 1 differential algebraic equations, following [47]. The main result presented there is that the asymptotic cost of the numerical solution of such initial value problems is polynomial in the number of bits of accuracy. Of course, this assumes smoothness of the problem and solution. A second result is that adaptive step-size, under modest realistic assumptions, is never worse than fixed stepsizes and can be an unbounded factor better.

In this chapter, we extend the results to initial value problems for high-index differential algebraic equations. We shall analyse an algorithm based on Pryce's method [65, 66] which allows solving numerically a general class of differential algebraic equation can be high-index, fully implicit and which may contain derivatives of higher order. The method does not involve index reduction and is based on expansion of the solution in terms of Taylor series. The analysis of this Taylor series method for DAE (as discussed by Pryce [65, 66] and Nedialkov & Pryce [62]) is complex. Another difficulty is an appropriate (smooth enough) choice of interpolant to facilitate the analysis.

In this chapter we shall assume that the index of the problem does not change throughout the integration, and that singularities and discontinuities are isolated and may be located at negligible cost and that solutions are otherwise smooth. We also assume that the DAE is suitable for automatic differentiation (see Rall [69], Griewank [38]). Under these assumptions, we show that the numerical solution of the initial value problem for high-index differential algebraic equations to which Pryce's method successfully applies is again of cost polynomial in the number of digits of accuracy (that is, the problem under consideration is tractable).

It is hoped that this theoretical analysis may be of use in the design and analysis of practical codes for highly accurate solutions of high-index differential algebraic equations.

4.1 Pryce's structural analysis method for DAEs

We consider an autonomous initial value problem for a differential algebraic equation in m dependent variables $x_j = x_j(t)$, with t, the scalar independent variable, in an interval $[a, b]$. The DAE has the general form

$$f_i(x_j,\ \text{derivatives of } x_j) = 0 \quad \text{for } i = 1 \ldots m\ . \tag{4.1}$$

The nonautonomous problems can be put in an autonomous form by increasing the dimension by one. The functions f_i are assumed piecewise analytic in their variables. We allow higher order derivatives of the dependent variables and nonlinear expressions in them.

We assume that if we add to the DAE extra ODEs that describe standard functions used in the functions f_i, then the problem is converted to a larger DAE whose new functions involve only the four basic arithmetic operations. The functions f_i belong thus to the class of functions to which automatic differentiation applies.

We assume that the initial condition satisfies a set of constraints which is sufficient for it to be a consistent point for (4.1).

Definition 3 *A consistent point of the DAE (4.1) is defined as a scalar t^* together with a set of scalars $\xi_{j\ell}$, where (j, l) are all the elements of a finite set S_J, so that there exists a solution to the DAE (4.1) in a neighbourhood of $t = t^*$ with $x_j^{(\ell)}(t^*) = \xi_{j\ell}$ for all $(j, \ell) \in S_J$ and the solution is unique.*

The set S_J need not be minimal.

In [66, 65], Pryce develops an algorithm for solving such a system by expanding the solution in Taylor series. The method is based on a pre-analysis stage which consists in solving an assignment problem. The assignment problem gives the *offsets*, that is a set of nonnegative integers $c_i \geq 0,\ d_j \geq 0$, with c_i being the number of times the equation f_i has to be differentiated to reduce the DAE to an ODE and d_j is the highest order to which variable x_j appears in the corresponding ODE. The offsets also indicate which equations to solve for which unknowns, and give a systematic way of determining consistent initial conditions.

The next step of the method consists in generating the system Jacobian. Provided the Jacobian is non-singular at each integration step, the method succeeds and the Taylor series coefficients can be computed to the desired order. This is based on the fact that, at each step of integration, after some initial stages, the Taylor coefficients can be obtained as solutions of some linear systems involving the system Jacobian.

We review below the main steps of Pryce's structural analysis and the corresponding method following [66, 65].

4.1.1 Structural analysis

1. The first step is to form an $n \times n$ *signature matrix* $\Sigma = (\sigma_{ij})$ with

$$\sigma_{ij} = \begin{cases} \text{order of derivative to which the variable } x_j \\ \qquad \text{appears in equation } f_i \quad , \\ \text{or } -\infty \text{ if the variable does not occur .} \end{cases}$$

2. Solve an assignment problem to determine a HVT (*highest value transversal*), which is a subset of indexes (i, j) with just one element on each row and each column, such that $\sum \sigma_{ij}$ is maximized and is finite. This is a standard problem in combinatorial optimization and algorithms to solve it may be found in, e.g., [7, 50]. We make the assumption that such a HVT exists.

3. Determine the *offsets* of the problem, which are the vectors **c**, **d** with d_i, $c_j \geq 0$ the smallest such that

$$d_j - c_i \geq \sigma_{ij} \qquad \text{for all } i \leq m,\ j \leq m$$

with equality on the HVT. Algorithms to solve this (dual) linear programming problem can be found in [7].

The *structural index* is then defined as

$$\nu = \max_i c_i + \begin{cases} 0 & \text{if all } d_j > 0 \\ 1 & \text{if some } d_j = 0. \end{cases} \tag{4.2}$$

4. Form the *system Jacobian* J_f, given by

$$J_{f\,ij} = \begin{cases} \dfrac{\partial f_i}{\partial x_j^{(d_j - c_i)}} & \text{if } x_j^{(d_j - c_i)} \text{ appears in } f_i \ , \\ 0 & \text{otherwise.} \end{cases}$$

5. Choose a consistent point. If J_f is non-singular at that point, then the solution can be computed with Taylor series in a neighbourhood of that point.

   We assume that
$$x_j(t) = \sum_{\ell \geq 0} \frac{1}{\ell!} x_{j,\ell}(t-t^*)^\ell,$$
   substitute it in (4.1) and expand the equations in Taylor series
$$f_i(x_j(t), \text{ derivatives of } x_j(t)) = \sum_{q \geq 0} \frac{1}{q!} f_{i,q}(t-t^*)^q \ ,$$
   and solve $f_{i,q} = 0$. Each $f_{i,q} = f_i^{(q)}(t^*)$ depends of a finite number of $x_{j,\ell} = x_j^{(\ell)}(t^*)$.

4.1.2 Notation

We define
$$\begin{aligned} k_d &= -\max_j d_j \\ k_c &= -\max_i c_i \end{aligned}$$
$(k_d \leq k_c)$. In this chapter, we consider
$$k_c \leq -1 \ ,$$
when the DAE (4.1) is at least of index 1. The case of index-1 DAEs with $k_c = 0$ is considered in the previous chapter (see also [47]).

We follow the notations from Pryce [66] and define for $k \geq k_d$:
$$\begin{aligned} J_k &= \{(j,\ell) : \ell = k + d_j \geq 0, 1 \leq j \leq m\} \\ I_k &= \{(i,q) : q = k + c_i \geq 0, 1 \leq i \leq m\}. \end{aligned}$$
We consider the following notation
$$f_I = \{f_i^{(q)} : \text{the } q\text{-th derivative of } f_i, (i,q) \in I\}$$

for some set I and

$$x_J = \{x_j^{(\ell)} : \text{the } \ell\text{-th derivative of } x_j, (j, \ell) \in J\}$$

for some set J. We denote by $I_{\leq k}$ ($I_{<k}$) the union of all I_r with $r \leq k$ ($r < k$ respectively) and by $J_{\leq k}$ ($J_{<k}$) the corresponding unions.

We shall also use the notations f_{I_k} (and x_{J_k}) for the values of the functions at a specific time and we shall omit to write the dependency of the time, if this is clear from the context.

4.1.3 Theoretical results

The choice of the offsets induces a specific block triangular structure on the system of equations to be solved and this structure is exploited by the method. At each stage k, f_{I_k} is a function only of the variables $x_{J \leq k}$. The method consists in solving, for each stage $k = k_d, k_d + 1, k_d + 2, \ldots$ the equations

$$f_{I_k} = f_{I_k}(x_{J<k}, x_{J_k}) = 0 \quad \text{ for variables } x_{J_k}. \tag{4.3}$$

This is equivalent to solving, for each k, the system $f_{i,k+c_i} = 0$ for those i with $k + c_i \geq 0$ in the unknowns $x_{j,k+d_j}$ for those j with $d_j + k \geq 0$.

If we consider

$$m_k = |I_k|, \qquad n_k = |J_k|,$$

where $|I|$ represents the cardinality of set I, then

$$m_k \leq n_k$$

for all stages k. This implies that at each stage we solve at most as many equations as unknowns (see [66]).

The following result (see [25, 66]) plays an important role in deriving the Taylor series method for DAE due to Pryce:

Lemma 1 (Griewank) *Let f be a function of $x_i(t)$, $i = 1, \ldots, m$ and their derivatives. If f does not depend on derivatives of x_j of order higher than ℓ, then*

$$\frac{\partial f^{(p)}}{\partial x_j^{\ell+p}} = \frac{\partial f}{\partial x_j^{\ell}}$$

for all $p \geq 1$.

By permuting the variables $x_j(t)$ so that the d_j are in decreasing order and by permuting the equations f_i so that the c_i are in decreasing order, it can be shown ([66], Proposition 4.1) that the Jacobian

$$J_{f\,k} = \frac{\partial f_{I_k}}{\partial x_{J_k}} \tag{4.4}$$

is the leading $m_k \times n_k$ submatrix of J_f. If J_f is non-singular, then $J_{f\,k}$ is full-row rank and thus there exists $J_{f\,k}^{\dagger}$, the Moore-Penrose inverse of $J_{f\,k}$ (see e.g. Ben-Israel and Greville [5]).

If a point $(t^*, x^*_{J\leq 0}) \in \mathbb{R} \times \mathbb{R}^{J\leq 0}$ satisfies (4.3) for all stages $k \leq 0$ then it is a consistent point of the DAE (4.1).

The following theorem (Pryce [65], Theorem 1.1) holds:

Theorem 7 *Let the functions f_i be analytic in a neighbourhood of a consistent point. Then the Taylor series method applied at this point succeeds iff the system Jacobian J_f is non-singular at this point.*

If this holds, then the DAE is locally solvable, the solution manifold has $\sum_j d_j - \sum_i c_i$ degrees of freedom and the DAE has structural index ν given by (4.2).

The method, if started at any initial guess sufficiently close to the given point, converges to a consistent point, $(\tilde{t}, \tilde{Y})$, and generates a series around $\tilde{t}$ which converges in some neighbourhood of $\tilde{t}$ to the solution of the DAE through that point.

Alternatively, the DAE can be reduced to an ODE using this method (Pryce [66], Theorem 4.2).

We assume the initial conditions

$$\frac{d^\ell}{dt^\ell} x_j(a) = x^0_{j,\ell} \text{ for all } \ell \leq d_j$$

satisfy the stages $k \leq 0$ of Pryce's algorithm at $t = a$,

$$f_{I_k}(x^0_{J_k}) = 0$$

for all $k \leq 0$ and thus is a consistent point for the DAE (4.1). We also assume that the system Jacobian is non-singular in a neighbourhood of the exact solution on the interval $[a, b]$.

We are interested in obtaining an approximate solution, $\chi_j(t)$, with a tolerance ε in residual errors $\|\delta_i(t)\|$, where

$$f_i(t, \chi_j(t), \text{ derivatives of } \chi_j(t)) = \delta_i(t) \ . \tag{4.5}$$

Definition 4 *The approximate solution $\chi_j(t)$ satisfies the DAE (4.1) with a tolerance ε in residual error if the residuals (4.5) satisfy, for all $i \leq m$,*

$$\left\| \frac{d^k}{dt^k} \delta_i(t) \right\| \leq \varepsilon \qquad \textit{for all } 0 \leq k \leq c_i \ . \tag{4.6}$$

4.2 Numerical solution

We are interested in finding a bound for the cost of the following algorithm for obtaining a numerical solution of the DAE (4.1) using the structural analysis of [66]: assume we have obtained at time t_n some values $x^n_{J_{\leq 0}}$ which satisfy the constraints $f_{I_{k \leq 0}} = 0$ more accurately than the desired tolerance. We generate the Taylor coefficients at

$t = t_n$ for the unknown functions up to the desired order and predict the values $\hat{x}^{n+1}_{J\leq 0}$ at the next integration step by computing the Taylor series with a chosen stepsize h_n.

We correct these values by projecting the approximate solution back on the constraint manifold. We do this by applying only one Newton projection at each stage $k \leq 0$. As suggested by numerical experiments, one Newton iteration is sufficient for satisfying sufficiently accurately the constraint corresponding to that stage for sufficiently small tolerance, see, e.g., Barrio [4] and Nedialkov [60]. The reason is that the truncated Taylor solution at the end of the step is a good approximation of the solution (due to the fact that we use a high order method). Therefore the convergence of Newton iteration is assured at tight enough accuracies. The advantage is that the cost of the algorithm is not increased unnecessarily.

The step is accepted if a given measure of the residual errors satisfy the tolerance (thus guaranteeing, as shown later, that some continuous extension of the discrete solution satisfies (4.6)).

4.2.1 Predictor step

We consider the truncated Taylor solution at time $t = t_n + h$

$$\hat{x}_j(t) = \sum_{\ell=0}^{p+d_j-1} \frac{1}{\ell!} x^n_{j,\ell}(t-t_n)^\ell \quad \text{for } j \leq m \ . \tag{4.7}$$

substitute it in the equations and expand in Taylor series:

$$f_i(\hat{x}_j,\ \text{derivatives of } \hat{x}_j) = \sum_{q\geq 0} \frac{1}{q!} f_{i,q}(x^n_{j,\ell})(t-t_n)^q \quad \text{for } i \leq m \ .$$

We solve the systems

$$f_{i,k+c_i}(x^n_{J\leq k}) = 0 \text{ for } k + c_i \geq 0$$

in the unknowns

$$x^n_{j,k+d_j} \text{ for } k + d_j \geq 0$$

for each $k = k_d, k_d + 1, \ldots, p - 1$.

It is important to note that for $k \leq 0$, these equations are already satisfied with a much better accuracy than the tolerance. Indeed, at the first integration step, these equations are satisfied by the initial consistent point, and at the other integration steps, they are satisfied since the values are the ones projected on the constraint locus at the previous step (as we shall see, asymptotically, the Gauss-Newton projections double the number of digits of accuracy in satisfying each constraint).

The stages $k \geq 1$ are always linear in the corresponding unknowns and involve the Jacobian $J_f(x^n_{J\leq 0})$, which is non-singular if the chosen tolerance is small enough.

On each interval $[t_n, t_n + h_n)$, the approximate Taylor solution (4.7) satisfies the problem

$$f_i(\hat{x}_j(t), \text{ derivatives of } \hat{x}_j(t)) = \hat{\delta}_i(t) \ , \qquad i = 1 \ldots m \ . \tag{4.8}$$

At time $t_{n+1} = t_n + h_n$ the predicted values are

$$\hat{x}^{n+1}_{j,\ell} = \hat{x}^{(\ell)}_j(t^-_{n+1})$$

for $\ell \leq d_j$, $j \leq m$.

The predicted point, $\hat{x}^{n+1}_{J\leq 0}$, is close to a consistent point of the DAE, and its distance to the constraint manifold is given by the truncation error. So for a small ε it will be in a neighbourhood of the exact solution where the system Jacobian, J_f, is non-singular. In the same neighbourhood, each J_{f_k} given by (4.4) is full row-rank and therefore $J^\dagger_{f_k}$, the Moore-Penrose inverse of J_{f_k} exists and is continuous in that neighbourhood (see Corless and Jeffrey [22]).

4.2.2 Corrector step

For each $k \leq 0$ we have at most as many equations as unknowns. We wish to satisfy these equations with a better accuracy than the one given by the truncation error. For each stage $k \leq 0$ we compute a set of corrected values, by applying only one Newton iteration per stage.

For $k < k_c$, there are no constraints to satisfy and we preserve the values of the variables corresponding to that stage :

$$x_{J_k} = \hat{x}_{J_k}$$

for all $k_d \leq k < k_c$ (if $k_d < k_c$).

We begin with the the Gauss-Newton projection

$$x^{n+1}_{J_{k_c}} - \hat{x}^{n+1}_{J_{k_c}} = -{J_f}^{\dagger}_{k_c}(\hat{x}^{n+1}_{J_{k_c}}) \cdot f_{I_{k_c}}(\hat{x}^{n+1}_{J_{k_c}}) \tag{4.9}$$

and continue, at each stage $k = k_c + 1, \ldots, 0$, with the following projections

$$x^{n+1}_{J_k} - \hat{x}^{n+1}_{J_k} = -{J_f}^{\dagger}_{k}(x^{n+1}_{J<k}, \hat{x}^{n+1}_{J_k}) \cdot f_{I_k}(x^{n+1}_{J<k}, \hat{x}^{n+1}_{J_k}) \ . \tag{4.10}$$

The values $x^{n+1}_{J<k}$ are already computed from the previous stages.

4.3 Error estimates

We shall analyse the errors of the Newton projections and we shall compute the residual errors over one integration step.

Since all f_i are analytic functions in their variables, the interval of integration is compact, and the system Jacobian is non-singular for the exact solution, then there exists a neighbourhood, in the space $\mathbb{R}^{J\leq 0}$, of the exact solution $x^e_{J\leq 0}([a,b])$ such that the Jacobian $J_f(z)$ is non-singular and all the Moore-Penrose inverses ${J_f}^{\dagger}_{k}(z)$, with

$k \leq 0$ are uniformly bounded on that neighbourhood, that is there exists $L > 0$ so that

$$\|J_{f_k}^{\dagger}(z)\| \leq L \quad \text{for all } k \leq 0. \tag{4.11}$$

For $k_c + 1 \leq k$, $f_{I_k}(y_{J \leq k})$ is *Lipschitz continuous* with respect to the variables $\{y_{J_l}\}_{l=k_c,\ldots,k-1}$ in some neighbourhood in $\mathbb{R}^{J \leq 0}$ of the exact solution, if there exists $M_k > 0$ so that

$$\begin{aligned} &\|f_{I_k}(y_{J<k_c}, y^1_{J_{k_c}}, \ldots, y^1_{J_{k-1}}, y_{J_k}) - f_{I_k}(y_{J<k_c}, y^2_{J_{k_c}}, \ldots, y^2_{J_{k-1}}, y_{J_k})\| \\ &\leq M_k \sum_{l=k_c}^{k-1} \|y^1_{J_l} - y^2_{J_l}\| \end{aligned} \tag{4.12}$$

for all $y^q_{J \leq k} = (y_{J<k_c}, y^q_{J_{k_c}}, \ldots, y^q_{J_{k-1}}, y_{J_k})$, with $q = 1, 2$, in that neighbourhood.

Let $t = t_n + h$ and $h \in [0, h_n]$.

Proposition 1 *If for all $k_c + 1 \leq k \leq 0$, $f_{I_k}(y_{J \leq k})$ is Lipschitz continuous with respect to the variables $\{y_{J_l}\}_{l=k_c,\ldots,k-1}$ in some neighbourhood in $\mathbb{R}^{J \leq 0}$ of the exact solution, then for ε small enough the following holds:*

$$x^{n+1}_{J_k} - \hat{x}^{n+1}_{J_k} = \mathcal{O}(h_n^{p-k}) \ , \quad k_c \leq k \leq 0 \tag{4.13}$$

and

$$f_{I_k}(\hat{x}_{J \leq k}(t)) = \Phi^n_{I_k} h^{(p-k)} + \mathcal{O}(h^{p-k+1}) \ , \quad k_c \leq k \leq 0 \tag{4.14}$$

$$f_{I_k}(x^{n+1}_{J<k}, \hat{x}^{n+1}_{J_k}) = \Phi^n_{I_k} h_n^{(p-k)} + \mathcal{O}(h_n^{p-k+1}) \ , \quad k_c + 1 \leq k \leq 0 \tag{4.15}$$

where the vectors $\Phi^n_{I_k}$ do not depend on h_n.

Proof. Consider

$$\phi^n_i = \frac{1}{(p + c_i)!} \frac{d^{p+c_i}}{dt^{p+c_i}} f_i(x_{J \leq 0})(t_n) \ .$$

These coefficients are independent of h and h_n, depending only on the value of some fixed function at $(x^n_{J \leq 0})$ (the beginning of the step). Then

$$f_i(\hat{x}_j(t), \text{ derivatives of } \hat{x}_j(t)) = \phi^n_i (t - t_n)^{p+c_i} + \mathcal{O}((t - t_n)^{p+c_i+1})$$

and thus

$$f_i^{(k+c_i)}(\hat{x}_j(t),\ \text{derivatives of } \hat{x}_j(t)) = \Phi^n_{i,k+c_i}(t-t_n)^{p-k} + \mathcal{O}((t-t_n)^{p-k+1})$$

where

$$\Phi^n_{i,k+c_i} = ((p+c_i)!/(p-k)!)\phi^n_i \ ,$$

provided $k \leq 0$ and $k + c_i \geq 0$. Thus we have obtained (4.14).

We also note that, by using (4.12) for $k_c + 1 \leq k$, we derive

$$f_{I_k}(x^{n+1}_{J<k}, \hat{x}^{n+1}_{J_k}) = f_{I_k}(\hat{x}^{n+1}_{J\leq k}) + \sum_{l=k_c}^{k-1} \mathcal{O}(\|x^{n+1}_{J_l} - \hat{x}^{n+1}_{J_l}\|) \tag{4.16}$$

We prove now (4.13) for k and (4.15) for $k+1$ by induction over $k \geq k_c$.

- Let $k = k_c$. By using (4.9) and the uniform-boundedness condition (4.11), we obtain

  $$\|x^{n+1}_{J_{k_c}} - \hat{x}^{n+1}_{J_{k_c}}\| \leq L\|f_{I_{k_c}}(\hat{x}^{n+1}_{J_{k_c}})\| \ .$$

  If we take $k = k_c$ in (4.14), we derive

  $$f_{I_{k_c}}(\hat{x}^{n+1}_{J_{k_c}}) = \mathcal{O}(h_n^{p-k_c})$$

  and thus we obtained (4.13) for $k = k_c$.

  If we consider $k = k_c + 1$ in (4.16) and we apply (4.13) for $k = k_c$ (obtained above), then we find :

  $$f_{I_{k_c+1}}(x^{n+1}_{J_{k_c}}, \hat{x}^{n+1}_{J_{k_c+1}}) = f_{I_{k_c+1}}(\hat{x}^{n+1}_{J\leq k_c+1}) + \mathcal{O}(h_n^{p-k_c}) \ .$$

- We now assume the hypothesis is true for all $\ell \leq k-1$ and show that it is true for $\ell = k$. By using (4.10) and (4.11), we obtain

  $$\|x^{n+1}_{J_k} - \hat{x}^{n+1}_{J_k}\| \leq L\|f(x^{n+1}_{J<k}, \hat{x}^{n+1}_{J_k})\| \ .$$

But from the induction step at $k-1$ we know that

$$\|f(x_{J<k}^{n+1}, \hat{x}_{J_k}^{n+1})\| = \mathcal{O}(h_n^{p-k})$$

and thus (4.13) is also true for k.

The induction ends by showing (4.15) at $k+1$. By applying the induction hypothesis (4.13) for $\ell \leq k$ in (4.16) for $k+1$, we obtain

$$f_{I_{k+1}}(x_{J<k+1}^{n+1}, \hat{x}_{J_{k+1}}^{n+1}) = f_{I_{k+1}}(\hat{x}_{J\leq k+1}^{n+1}) + \mathcal{O}(h_n^{p-k}) \ .$$

Note. For small enough ε, the Newton projections approximately double the number of digits of accuracy in the corresponding residual errors,

$$f_{I_k}(x_{J\leq k}^{n+1}) = \mathcal{O}(h_n^{2(p-k)}),$$

for $k \leq 0$. Indeed, by a Taylor expansion we obtain

$$\begin{aligned} f_{I_k}(x_{J\leq k}^{n+1}) &= f_{I_k}(x_{J<k}^{n+1}, \hat{x}_{J_k}^{n+1}) + J_{f\,k}(x_{J<k}^{n+1}, \hat{x}_{J_k}^{n+1})(x_{J_k}^{n+1} - \hat{x}_{J_k}^{n+1}) \\ &+ \frac{1}{2}(x_{J_k}^{n+1} - \hat{x}_{J_k}^{n+1})^T G(x_{J\leq k}^{n+1}, \hat{x}_{J_k}^{n+1})(x_{J_k}^{n+1} - \hat{x}_{J_k}^{n+1}) \ . \end{aligned}$$

where

$$G(x_{J\leq k}^{n+1}, \hat{x}_{J_k}^{n+1}) = \int_0^1 2(1-s)\frac{\partial^2}{\partial x_{J_k}\partial x_{J_k}} f_{I_k}(x_{J<k}^{n+1}, \hat{x}_{J_k}^{n+1} + s(x_{J_k}^{n+1} - \hat{x}_{J_k}^{n+1}))ds \ .$$

The first two terms in the expansion cancel each other due to (4.9)-(4.10), and then by using (4.11) we derive

$$\|f_{I_k}(x_{J\leq k}^{n+1})\| \leq \frac{1}{2} N_k L^2 \|f_{I_k}(x_{J<k}^{n+1}, \hat{x}_{J_k}^{n+1})\|^2$$

where $N_k = \sup \|\dfrac{\partial^2 f_{I_k}}{\partial x_{J_k}\partial x_{J_k}}\|$ on some neighbourhood. We end the proof by applying Proposition 1.

We see below how this result works on a specific problem.

4.3.1 Example

We consider the example of nonlinear simple pendulum, which is described by the following equations

$$\begin{cases} 0 = f_1 = x'' + x\lambda \\ 0 = f_2 = y'' + y\lambda - g \\ 0 = f_3 = x^2 + y^2 - L^2 \end{cases} \tag{4.17}$$

where x, y are the ordinary orthogonal coordinates and $L\lambda$ is the tension in the string. The length of the pendulum, L, and the gravity, g, are positive constants. Of course, for the particular case of simple pendulum, by changing to polar coordinates, the DAE is transformed to an ODE. Since, in general, such a transformation of coordinates is very difficult to generate, we shall not consider this approach.

The structural analysis for this problem (see also [66]) is given below. The offsets of the variables (x, y, λ) are $(2, 2, 0)$, and those of the equations (f_1, f_2, f_3) are $(0, 0, 2)$ (see Table 4.1).

	x	y	λ	c_i
f_1	**2**	$-\infty$	0	0
f_2	$-\infty$	2	**0**	0
f_3	0	**0**	$-\infty$	2
d_j	2	2	0	

Table 4.1: The signature matrix and the offsets for the pendulum problem.

The system Jacobian is given by

$$J = \begin{bmatrix} 1 & 0 & x \\ 0 & 1 & y \\ 2x & 2y & 0 \end{bmatrix}$$

and is non-singular away from the origin since $\det(J) = -2(x^2 + y^2) = -2L^2$. Consequently, the system is solvable by Pryce's method. This is an index-3 DAE.

According to the method, the equations to be solved are grouped as follows:

$f_{I_{-2}} = (f_3)$ in unknowns $X_{J_{-2}} = (x, y)$ at stage $k = -2$,

$f_{I_{-1}} = (f_3')$ in unknowns $X_{J_{-1}} = (x', y')$ at stage $k = -1$,

$f_{I_0} = (f_1, f_2, f_3'')$ in unknowns $X_{J_0} = (x'', y'', \lambda)$ at stage $k = 0$.

We analyse the validity of the Lipschitz continuity conditions from Proposition 1 for the particular case of the pendulum problem.

Note that the exact solution, $x^e_{J\leq 0}$, lies in $\mathbb{R}^7$. Consider a neighbourhood U of $x^e_{J\leq 0}$ and a generic point $(x, y, x', y', x'', y'', \lambda) \in U$ (we think of x', y' , ... as variables, not derivatives).

The problem reduces to checking that for an appropriate choice of the neighbourhood U, the following hold

(I). $F_{-1} = F_{-1}(x, y, x', y')$ is Lipschitz continuous in the arguments x and y, where

$$F_{-1} = (f_3'(x, y)) = (2xx' + 2yy')\ ,$$

(II). $F_0 = F_0(x, y, x', y', x'', y'', \lambda)$ is Lipschitz continuous in the arguments x, y,x' and y' , where

$$F_0 = \begin{pmatrix} f_1 \\ f_2 \\ f_3'' \end{pmatrix} = \begin{pmatrix} x'' + \lambda x \\ y'' + \lambda y - g \\ 2xx'' + 2yy'' + 2x'^2 + 2y'^2 \end{pmatrix} .$$

In order to prove Lipschitz continuity of each function with respect to some variable, it suffices to show that the partial derivative of that function with respect to the chosen variable is bounded on an appropriate neighbourhood.

Therefore the following need to be bounded:

(I).

$$\begin{bmatrix} \frac{\partial f_3'}{\partial x'} & \frac{\partial f_3'}{\partial y'} \end{bmatrix} = \begin{bmatrix} 2x' & 2y' \end{bmatrix}$$

(II).

$$\begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial x'} & \frac{\partial f_1}{\partial y'} \\ \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial x'} & \frac{\partial f_2}{\partial y'} \\ \\ \frac{\partial f_3''}{\partial x} & \frac{\partial f_3''}{\partial y} & \frac{\partial f_3''}{\partial x'} & \frac{\partial f_3''}{\partial y'} \end{bmatrix} = \begin{bmatrix} \lambda & 0 & 0 & 0 \\ \\ 0 & \lambda & 0 & 0 \\ \\ 2x'' & 2y'' & 4x' & 4y' \end{bmatrix}$$

Indeed, with a compactness and smoothness argument (the solution is analytic on some compact interval), all of these variables are bounded on some neighbourhood of the exact solution.

Hence Proposition 1 applies and we can derive the order of the residual errors and of the local errors in the variables at each stage. If for a given p, the approximate Taylor solution is computed according to formula (4.7), then the order of the local errors corresponding to the initial stages are those listed in Tabel 4.2.

4.4 Dense Output

In the present chapter we are not concerned with finding the optimal interpolant (the one that minimizes some norm of the residual). We merely want to measure the size of the residual associated with a sufficiently accurate continuous approximation.

Stage	equations	order of residual	unknowns	order of error in projection
$k=-2$	$f_3 = 0$	h^{p+2}	x, y	h^{p+2}
$k=-1$	$f_3' = 0$	h^{p+1}	x', y'	h^{p+1}
$k=0$	$f_1 = 0$ $f_2 = 0$ $f_3'' = 0$	h^p	x'', y'', λ	h^p

Table 4.2: The order of the residual errors and of the error in projection, for the initial stages in the simple pendulum problem.

Let $[t_n, t_{n+1}]$ be the current interval of integration and t arbitrary in this interval. Consider χ_j to be a piecewise-polynomial approximation of the variable x_j. On each interval $[t_n, t_{n+1}]$, define $\chi_j(t)$ to be the Hermite interpolant which satisfies the following conditions:

$$\begin{aligned} \chi_j^{(k)}(t_n) &= x_{j,k}^n \quad \text{for all } k = 0, \ldots, p + d_j - 1 \\ \chi_j^{(k)}(t_{n+1}) &= x_{j,k}^{n+1} \quad \text{for all } k = 0, \ldots, d_j \ . \end{aligned} \tag{4.18}$$

This interpolant has continuous derivatives up to order d_j on $[a, b]$.

We denote the difference between the projected values of the variables and their corresponding Taylor approximation at t_{n+1} by

$$\delta x_{j,\ell}^{n+1} = x_{j,\ell}^{n+1} - \hat{x}_{j,\ell}^{n+1}$$

for all $1 \le j \le m$, $0 \le \ell \le d_j$. If for each χ_j we consider the basis

$$\begin{array}{lll} (t - t_n)^k & \text{for} & 0 \le k \le p + d_j - 1 \\ (t - t_n)^{p+d_j}(t - t_{n+1})^k & \text{for} & 0 \le k \le d_j \end{array}$$

then we find the following representation of the interpolant:

$$\chi_j(t) = \hat{x}_j(t) + \sum_{\ell=0}^{d_j} a_{p+d_j+\ell}^j (t - t_{n+1})^\ell (t - t_n)^{p+d_j} \tag{4.19}$$

where

$$a^j_{p+d_j+k} = h_n^{-(p+d_j+k)} \sum_{\ell=0}^{k} c^j_{k,\ell} h_n^\ell \delta x^{n+1}_{j,\ell}$$

for all $1 \leq j \leq m$ and $0 \leq k \leq d_j$. The coefficients $c_{k,l}$ are some constants depending only on p and the offsets.

Let the difference between the interpolant and the corresponding Taylor (truncated) solution be

$$R_j(t) = \chi_j(t) - \hat{x}_j(t)$$

for all $j \leq m$, then we find that

$$R_j(t) = \sum_{\ell=0}^{d_j} p_{j,\ell}(\tau) h_n^\ell \delta x^{n+1}_{j,\ell} \tag{4.20}$$

where $\tau = \frac{1}{h_n}(t - t_n)$, $\tau \in [0,1]$ and the polynomials $p_{j,l}$ have coefficients depending only on p and the offsets. By convention, the polynomials corresponding to those variables which are not projected are identically zero. In fact, we can show that

$$\|R_{J_0}(t)\| \leq \alpha \sum_{k=0}^{k_c} h_n^k \|\delta x^{n+1}_{J_k}\| \tag{4.21}$$

where we can choose, for example,

$$\alpha = \max |p^{(d_j)}_{j,l}(\tau)|$$

where the maximum is taken over all $\tau \in [0,1]$, all $j \leq m,\ l \leq d_j$ and depends only on p and the offsets.

By using equations (4.9)-(4.10) and Proposition 1, we derive that

$$\|R_{J_0}(t)\| \leq \alpha \sum_{\ell=0}^{k_c} h_n^\ell \|J_{f\ell}^\dagger(\xi^{n+1}_\ell) \cdot \hat{\delta}_{I_\ell}(t^-_{n+1})\|) + \mathcal{O}(h_n^{p+1}) \tag{4.22}$$

and is of order $\mathcal{O}(h_n^p)$.

4.4.1 Residual control

We are interested in measuring the magnitude of the residuals associated with the continuous approximation (4.19). Using Proposition 1, we obtain that at each stage $k \leq 0$ on the interval $[t_n, t_{n+1}]$, the corresponding residual is of order $\mathcal{O}(h_n^{p-k})$, thus the dominating residual is obtained at stage $k = 0$ and is of order $\mathcal{O}(h_n^p)$. This residual is given by

$$\delta_{I_0}(t) = f_{I_0}(\chi_{J\leq 0}(t)) \ . \tag{4.23}$$

By using (4.20) and a Taylor expansion of (4.23) around the predicted solution (4.7), we derive

$$\delta_{I_0}(t) = \hat{\delta}_{I_0}(t) + J_{f_0}(\hat{x}_{J\leq 0}(t)) \cdot R_{J_0}(t) + \mathcal{O}(h_n^{p+1}) \ . \tag{4.24}$$

Therefore, by using (4.21), (4.24) and Proposition 1, we find that the principal error term is bounded by $(\alpha + 1)\psi_n h_n^p$ with

$$\psi_n h_n^p = \|J_{f_0}(\hat{x}_{J\leq 0}^{n+1})\| \sum_{\ell=0}^{k_c} h_n^\ell \|J_{f\ell}^\dagger(\xi_\ell^{n+1})\hat{\delta}_{I_\ell}(t_{n+1}^-)\| \ . \tag{4.25}$$

It is sufficient to require that

$$\psi_n h_n^p \leq \frac{1}{\alpha + 1}\varepsilon$$

to guarantee that the residual is below the tolerance (higher order terms are neglected).

Remark. The maximum $\alpha = \alpha(p)$ depends on p and if $\mathcal{K}$ is the highest among all degrees of the polynomials $p_{j,\ell}^{(d_j)}$ with $\ell = 0, \ldots, d_j$, then $[\alpha(p) + 1]$ is $\mathcal{O}(p^{\mathcal{K}})$.

4.4.2 Example

We return now to the nonlinear simple pendulum problem governed by the equations (4.17). We want to construct the corresponding interpolants, as described above, and estimate the residual errors.

The Hermite interpolants corresponding to (x, y, z) and the step n of integration are respectively:

$$\begin{cases} \chi_1(t) &= \hat{x}(t) + p_{1,0}(\tau)\delta x_0^{n+1} + h_n p_{1,1}(\tau)\delta x_1^{n+1} + h_n^2 p_{1,2}(\tau)\delta x_2^{n+1} \\ \chi_2(t) &= \hat{y}(t) + p_{2,0}(\tau)\delta y_0^{n+1} + h_n p_{2,1}(\tau)\delta y_1^{n+1} + h_n^2 p_{2,2}(\tau)\delta x_2^{n+1} \\ \chi_3(t) &= \hat{\lambda}(t) + p_{3,0}(\tau)\delta\lambda_0^{n+1} \end{cases}$$

where

$$\begin{cases} p_{1,0}(\tau) = p_{2,0}(\tau) &= \tau^{p+2}[1-(p+2)(\tau-1)+\frac{1}{2}(p+2)(p+3)(\tau-1)^2] \\ p_{1,1}(\tau) = p_{2,1}(\tau) &= \tau^{p+2}(\tau-1)[1-(p+2)(\tau-1)] \\ p_{1,2}(\tau) = p_{2,2}(\tau) &= \frac{1}{2}\tau^{p+2}(\tau-1)^2 \\ p_{3,0}(\tau) &= \tau^p\ . \end{cases}$$

Using (4.20), we derive the errors in the interpolation at stage $k = 0$:

$$R_{J_0}(t) = \begin{pmatrix} \frac{1}{h_n^2}p_{1,0}^{(2)}(\tau)\delta x_0^{n+1} + \frac{1}{h_n}p_{1,1}^{(2)}(\tau)\delta x_1^{n+1} + p_{1,2}^{(2)}(\tau)\delta x_2^{n+1} \\ \frac{1}{h_n^2}p_{2,0}^{(2)}(\tau)\delta y_0^{n+1} + \frac{1}{h_n}p_{2,1}^{(2)}(\tau)\delta y_1^{n+1} + p_{2,2}^{(2)}(\tau)\delta y_2^{n+1} \\ p_{3,0}(\tau)\delta\lambda_0^{n+1} \end{pmatrix}$$

where

$$\begin{cases} p_{1,0}^{(2)}(\tau) &= p_{2,0}^{(2)}(\tau) = \frac{1}{2}(p+2)(p+3)(p+4)\tau^p(\tau-1)[(p+3)(\tau-1)+2] \\ p_{1,1}^{(2)}(\tau) &= p_{2,1}^{(2)}(\tau) = -(p+2)(p+3)(\tau-1)[(p+4)(\tau-1)+3] \\ p_{1,2}^{(2)}(\tau) &= p_{2,2}^{(2)}(\tau) = \frac{1}{2}(p+3)(p+4)(\tau-1)^2 + 2(p+3)(\tau-1)+1\ . \end{cases}$$

Therefore $\alpha(p) = \mathcal{O}(p^4)$.

4.5 Taylor series solution by automatic differentiation

Before stating the main result of this chapter we shall analyse the cost of generating Taylor series by automatic differentiation, for ordinary differential equations, following

Corliss and Chang [25], and then for differential algebraic equations, using Pryce's method.

We consider that the functions which define the differential equations are generated by a finite combination of the basic arithmetic operations (addition +, subtraction −, multiplication ×, division /, exponentiation ∗∗), and a set of library functions. The library functions are square root, the sine, cosine, tangent, their inverses, logarithms, exponential functions, hyperbolic functions and inverses, and other functions which are solutions of differential equations of a similar type, such as Bessel functions.

Examples of recurrence relations for automatic differentiation include

$$\begin{aligned} f &= \exp(u), & f' &= fu' \\ f &= \log(u), & uf' &= u' \\ f &= \sin(u), & f' &= gu' \\ g &= \cos(u), & g' &= -fu' \ . \end{aligned}$$

Each time a product of functions is differentiated, Leibnitz's rule is used.

4.5.1 Leibnitz's rule.

Let us denote, for any function g,

$$G(k+1) = \frac{g^{(k)}(t_0)h^k}{k!}, h = t - t_0.$$

Consider the product

$$h(t) = f(t)g(t).$$

Using Leibnitz's rule for derivation we obtain the recurrence

$$H(k) = \sum_{j=1}^{k} F(j)G(k-j+1).$$

Clearly, the number of arithmetic operations needed to compute the kth coefficient grows at most linearly with k. Therefore, generating p coefficients takes $\mathcal{O}(p^2)$ arithmetic operations.

We note that there is an $\mathcal{O}(p \log p)$ algorithm due to Brent and Kung [10]. We do not use this here because our main result, polynomial cost of the solution of high-index DAE, is true even with the more expensive naive algorithm.

Consider, for example, the function

$$f(t) = u(t)^s.$$

In this case, we differentiate it to obtain

$$f' = su^{s-1}u' = \frac{sfu'}{u}$$

which is rewritten as

$$f'u = su'f$$

and then apply Leibnitz's rule to both sides.

We first review the cost of computing Taylor series solutions for ODE, following [25].

4.5.2 Automatic differentiation for ODE

Assume next that we have m differential equations in our system, q products of functions, and p terms in the power series. Also, denote by r the number of temporary series. As each product or function requires the generation and storage of one or two temporary series, we have $r = \Theta(q)$ (that is there exist some constants c_1 and c_2 such that $c_1 q \leq r \leq c_2 q$ for all q). Denote also d_i the order of the highest derivative in equation i, $1 \leq i \leq m$.

The total number of storage locations is then (see Corliss and Chang[25])

$$\sum_{i=1}^{m}(p+d_i)+rp,$$

where any overhead, such as memory allocation costs, is ignored. The first sum appears because for each equation in the system we need to store p terms and d_i initial conditions. For the temporary series we need only the p terms for each, which gives the last term.

The number of arithmetic operations required at each integration step to generate all the coefficients in the series is

$$\mathcal{O}(p^2(r+m))$$

as there are $r+m$ series and we need (see above) $\mathcal{O}(p^2)$ operations for each.

4.5.3 Automatic differentiation for DAE.

Let us assume m, p, and q have the same meaning as for ODEs.

For the algorithm based on Pryce's method, we compute the coefficients involved in the stages $k \leq 0$ separately, with Newton projections (at the previous step). We are only interested at this point in the cost of computing the Taylor coefficients for the stages $k = 1, \ldots, p$, for one integration step.

The differential algebraic system is extended such that it includes also the ordinary differential equations describing the intermediate functions involved.

Let X_k denote the vector containing the highest order Taylor coefficients of the unknown function and of the intermediate functions, that we want to compute at the stage k. By $X_{\leq k-1}$ we denote the vector of the Taylor coefficients of the unknowns and of the products, already computed at stages $\ell \leq k-1$.

The extended recurrence relation is of the form

$$X_k = J^{-1}_{m+q,m+q} \cdot H_k(X_{\leq k-1}) \ .$$

The time needed to factor the extended Jacobian $J_{m+q,m+q}$ (of course we do not form the inverse explicitly) is $\mathcal{O}((m+q)^3)$ and this is done only once per integration step.

Then, for each stage k, $0 \leq k \leq p-1$, we perform first all operations needed to reduce $H_k(X_{\leq k-1})$ to an array of real numbers. For this we need $\mathcal{O}(k(m+q))$ arithmetic operations. Then, to multiply with the extended Jacobian, we need $\mathcal{O}((m+q)^2)$ operations.

For all k, $0 \leq k \leq p-1$, we need $\mathcal{O}(p^2(m+q) + p(m+q)^2)$.

Therefore, the total number of arithmetic operations to generate all Taylor coefficients up to stage $k = p$ per integration step is

$$\mathcal{O}((m+q)^3 + (m+q)p^2). \tag{4.26}$$

4.6 Polynomial cost of Pryce's algorithm

We show below that the cost of computing an approximate solution of the DAE (4.1) which satisfies the tolerance with the algorithm under investigation is polynomial in the number of digits of accuracy requested. Moreover, provided the local error coefficients given by (4.25) satisfy some regularity conditions described below, the cost of computing the solution with the same accuracy is minimized on the equidistributed mesh. In order to obtain this, we make use of a result on equidistribution from Corless [18], which is given below.

For a vector $\Psi = [\psi_1, \cdots, \psi_N]$ we define the *s-norm* as

$$\|\Psi\|_s = \left(\sum_{i=1}^{N} \psi_i^s \right)^{1/s}$$

and the *Hölder s-mean* as

$$\mathcal{M}_s(\Psi) = \left(\frac{1}{N} \sum_{i=1}^{N} \psi_i^s \right)^{1/s} .$$

We assume that the vector of the error coefficients $\Psi_N = [\psi_1, \cdots, \psi_N]$ satisfies some regularity conditions. First, we assume that there exists a positive integer N_0 such that

$$\mathcal{M}_s(\Psi_{N_2}) \leq \mathcal{M}_s(\Psi_{N_1})$$

for $N_2 > N_1 \geq N_0$. This means that if the mesh is sufficiently fine, the Hölder mean does not increase with further refinement. Second, we assume that the same property holds for the maximum norm of the local error coefficients, $\|\Phi_N\|_\infty$.

Following the standard theory of computational complexity Werschulz [84], we ignore memory hierarchy, overheads and interpolation costs. The standard theory counts the number of function evaluations as the main contributor to the cost. In contrast, we count, in our model of the cost for algorithms using Taylor series methods, the number of arithmetic operations following Corliss and Chang [25]) (see Chapter 3 for details).

The cost per step for the method investigated in this chapter, for a fixed order, is the same for all steps and consists of the cost of obtaining the Taylor series plus the cost of $[1 - k_c]$ Newton projections. Thus, the cost of the method for fixed order is directly proportional to the number of steps taken.

4.6.1 Optimal mesh

A mesh is optimal if it takes the minimum number of steps to compute an approximate solution on the interval $[a, b]$, satisfying the tolerance.

A first consequence of the Minimax theorem 1 (Chapter 2) and the regularity condition on the Hölder mean of the local error coefficients is that the equidistributing mesh

$$\psi_i h_i^p = \bar{h}^p \mathcal{M}_{-1/p}(\Psi) = \frac{\varepsilon}{1+\alpha} \text{ for all } 1 \leq i \leq N \tag{4.27}$$

is an optimal mesh for the algorithm.

Indeed, similarly to [18], we note that, due to the regularity condition, the sequence

$$\mathcal{M}_{-1/p}(\Psi_N)(\frac{b-a}{N})^p, \ N \geq 1$$

will become nonincreasing for $N > N_0$ and therefore the first time after N_0 when a component has a value less or equal with the tolerance, then that N represents the minimal number of steps for which the tolerance is satisfied, and the corresponding mesh is optimal.

Since the number of steps corresponding to the equidistributing mesh is $N = (b-a)/\bar{h}$ then, using (4.27), we derive

$$N = (b-a) \cdot (\mathcal{M}_{-1/p}(\Psi_N))^{1/p}(\frac{\varepsilon}{\alpha+1})^{-1/p}. \tag{4.28}$$

A second consequence of the Minimax theorem (and regularity conditions on the Hölder mean and infinity norm) is that the cost of computing an approximate solution satisfying a given tolerance is never less on a fixed step mesh than on an equidistributed mesh for the above algorithm.

Also by analogy to [18], we take the ratio of the number of steps for the non-adaptive mesh,

$$N_{fix} = (b-a)\|\Psi_{N_{fix}}\|_\infty^{1/p}(\frac{\varepsilon}{\alpha+1})^{-1/p} ,$$

and the number of steps of the equidistributing mesh, (4.28). We derive that

$$\frac{N_{fix}}{N} = \left(\frac{\|\Psi_{N_{fix}}\|_\infty}{\mathcal{M}_{-1/p}(\Psi_N)}\right)^{1/p}$$

which, due to the regularity conditions, can only happen when $N_{fix} \geq N$.

Thus, also the ratio of the two costs (the ratio of the number of steps corresponding to each mesh) is given by

$$\|\Psi_{fix}\|_\infty / (\mathcal{M}_{-1/p}(\Psi_{ad}))^{1/p} .$$

4.6.2 Main result

Theorem 8 *The minimal cost of computing the solution of (4.1) with Pryce's method is polynomial in the number of digits of accuracy requested and is bounded above by*

$$\mathcal{C}(b-a)e^{2+(\mathcal{K}/e)}(\mathcal{M}_{-1/p}(\Psi_N))^{1/p}B^4$$

where $\mathcal{C}, \mathcal{K}$ are some constants. The minimum cost is reached on the equidistributing mesh.

Proof. Order-p accurate solutions computed with Taylor series using automatic differentiation can be obtained in $\mathcal{O}(p^2)$ operations, according to (4.26).

If naive multiplication is used, the cost of doing arithmetics with on B bits of accuracy is $\mathcal{O}(B^2)$. Thus the cost of obtaining the Taylor series at one step costs cp^2B^2. The constant c depends on the size of the problem as well as of the complexity of the differential equation system.

The cost of $[1-k_c]$ Newton projection is independent of p and costs $\mathcal{O}(B^2)$. Therefore the cost of one step is $C_1 p^2 B^2$.

Using (4.28), we obtain the total cost corresponding to the equidistributed mesh:

$$C_1(b-a)(\mathcal{M}_{-1/p}(\Psi_N))^{1/p} B^2 p^2 (\frac{\varepsilon}{\alpha+1})^{-1/p} \ . \tag{4.29}$$

Since there exist positive constants C_2 and $\mathcal{K}$ so that

$$\alpha + 1 \leq C_2 p^{\mathcal{K}} \ ,$$

therefore the following holds

$$(\alpha+1)^{1/p} \leq C_2 e^{\mathcal{K}/e} \ .$$

Choosing

$$p = \lceil B/2 \rceil$$

in (4.29) and $\mathcal{C} = C_1 C_2 / 4$, we obtain an upper bound of the minimum cost given by

$$\mathcal{C}(b-a)e^{2+(\mathcal{K}/e)}(\mathcal{M}_{-1/p}(\Psi_N))^{1/p} B^4 \ .$$

Remarks. We note that the cost of computing the solution on a fixed-step mesh is also polynomial in the number of digits of accuracy.

Furthermore, we remark that the dependence on the dimension of the problem is hidden in the constant factor in the expression of the cost.

Our model proposes also an almost optimal order of the method depending on the prescribed precision. For the practical application, the round-off error should be correlated with the need for a higher accuracy.

Brent and Kung [10] series generation algorithm reduces the cost to $\mathcal{O}(B^3 \ln B)$.

We also mention that efficient methods for accurate location of the singularities exist. Automatic methods of detection for certain types of singularities for ODE, based on Darboux type tests, are investigated in Chang and Corliss [15].

In addition, techniques for avoiding passing through a singularity have been developed for singularities which are poles (Corliss [23, 24]). These techniques involve analytic continuation in the complex plane (also called “pole vaulting”) in regions of analyticity. One advantage of such methods of integration around the pole is that they allow large steps while maintaining the requested accuracy and are therefore cheaper.

It is important to note that *very high accuracies* (of order $\mathbf{10^{-150}}$) have been very recently reported for solving DAEs with Pryce’s method [67]. Currently, differential algebraic systems with up to 100 equations can be solved with existing software using Pryce’s method [67]. Such a solver (HIDAES - High-Index DAE Solver) was developed by Ned Nedialkov [60, 67]. These results support our theory which predicts polynomial cost in the number of bits of accuracy.

4.7 Summary

In this chapter, we have investigated the cost of solving initial value problems for high-index differential algebraic equations depending on the number of digits of accuracy requested.

We analysed an algorithm based on a Taylor series method developed by Pryce for a general class of differential algebraic equations. We showed that the cost of computing a solution with this algorithm is polynomial in the number of digits of accuracy. We also showed that adaption is better than nonadaption.

According to our model, we expect the Taylor series method for DAEs to behave very well for moderately stiff problems, when high precision is required for problems of not too high a dimension (as many of the interesting practical applications are). The theoretical results obtained with our model agree well with the results obtained using the existing implementations of the method.

Chapter 5

Numerical solution of polynomial systems

The problem of finding all isolated roots of polynomial systems is of major interest. The aim here is to compute the cost of finding *all* isolated roots of a polynomial system. By using the method presented below, each isolated root can be computed independently, and therefore sequential computation of all isolated roots is possible (and the cost of computing them is the sum of each individual cost of computing a root). This can be achieved provided that all the isolated roots have multiplicity 1.

The case of higher multiplicity for an isolated root is not treated here. We mention that there exist tools to detect multiplicity and methods to remove it (called "deflation of multiplicity"), e.g., differentiation of the system.

In this chapter, we focus on finding a low (polynomial) cost algorithm of approximating *one* isolated regular root of a polynomial system depending on one parameter, namely the number of digits of accuracy requested for the residual.

The study of the complexity of approximating all the isolated roots of a generic system of polynomials with respect to other parameters can be found in, e.g., [74].

We use the homotopy continuation method for finding all isolated roots of a polynomial system. The method consists in finding the numerical solution by continuously deforming the target system from a starting system with known solutions. A significant accomplishment of homotopy theory has been to give specific forms of the (new) start systems which guarantee that all isolated roots of the target system can be recovered by tracking the paths starting from the known solutions (see, e.g., [2, 77, 82]).

We note that the path tracking problem may be obtained by applying Pryce's method for DAE solving [65, 66] directly to the case of polynomial systems or indirectly, after transforming the problem to an index-1 DAE, to a problem to which the method applies. In this chapter, we shall choose the second approach and show that the results of this thesis apply.

The particular choice of the homotopy guarantees with probability 1 that the system Jacobian is non-singular along the path. Therefore, we may assume that the path stays away from the locus corresponding to singular Jacobian and thus Pryce's method applies.

Consequently, by assuming that such a curve exists and the Jacobian does not become singular along the path, we use the intrinsic smoothness of the problem and we find an algorithm of cost polynomial in the number of bits of accuracy for numerically approximating one isolated root of the given polynomial system.

5.1 Homotopy continuation

Consider a system of polynomials (the 'target system') which we want to solve

$$p = (p_1, \dots, p_m)^T = 0, \tag{5.1}$$

where $p : \mathbb{C}^m \to \mathbb{C}^m$ and $x = (x_1, \dots, x_m) \in \mathbb{C}^m$.

Assume there exists a polynomial system (a 'starting system') whose solutions are all isolated and known:

$$q(x) = 0 \tag{5.2}$$

We construct the following homotopy :

$$H(t, x) = (1 - t)q(x) + tp(x) \ . \tag{5.3}$$

We want to solve the problem

$$H(t, x(t)) = 0, t \in [0, 1]. \tag{5.4}$$

Obviously, $H(0, x) = q(x) = 0$ has known solutions and $H(1, x) = p(x) = 0$ is the system we want to solve. The starting point of the problem (5.4) is chosen to be a known solution $x_0 \in \mathbb{C}^m$ of (5.2), thus

$$H(0, x_0) = 0 \ .$$

Once the polynomial p is given, we wish to construct a polynomial q, such that the homotopy (5.3) is a "good homotopy" (see also Verschelde [82], Definition 1.2.4).

Definition 5 *A* good homotopy $H(t, x)$ *should satisfy the following properties:*

1. *(triviality) The solutions at* $t = 0$ *are known.*

2. *(smoothness) The solution set of* $H(t, x(t)) = 0$ *for all* $t \in [0, 1]$ *consists of a finite number of smooth paths, each parametrized by* t*.*

3. *(accessibility) Every isolated solution of* $H(1, x) = 0$ *is reached by some path originating at a solution of* $H(0, x) = 0$*. If* r *is the multiplicity of an isolated solution* x^**, then there are exactly* r *path converging to* x^**.*

For arbitrary target systems $p(x)$, methods for constructing good start systems are known (see, e.g., [77]).

The basic ideas behind showing that a homotopy is a good homotopy go back to the breakthrough paper [6], which shows that the number of roots is bounded by the mixed volume (which may be much lower than the Bèzout number). The proof of [6] relies on constructing a start system and showing that the analytic paths originating in the start system reach all the isolated roots of the target system. Since then, there has been much progress in finding efficient choices for the start systems ([77] and references therein) based on mixed volumes. Such choices not only find all isolated solutions, but also minimize and, in many cases, eliminate diverging paths.

The issue of optimal starting polynomial selections depending on the structure of the given polynomial p (such as density, sparsity, etc.) for constructing good homotopies is not discussed here.

For an example of a start system q for which the linear homotopy (5.3) is a good homotopy we refer the reader to Theorem 4.1.13 in [82]. Consequently, for such a homotopy (with probability 1) each isolated root will lie at the end of an analytic homotopy path starting from the roots of the starting system q. In particular, one such path, stays away from the locus corresponding to its singular Jacobian (with probability 1).

For the remainder of this chapter, we assume that q is such that (5.3) is a good homotopy.

5.2 Numerical solution

We assume the solution of the target polynomial system is of multiplicity 1.

By numerically solving (5.4), we shall obtain an approximate solution of the curve of exact solutions, $x_j^e(t), j = 1, \ldots, m$, where

$$H(t, x^e(t)) = 0, \text{ for all } t \in [0, 1] \ .$$

The numerical solution is obtained as values at the points of the mesh, which can be interpolated so that a continuous extension is obtained, $x_j(t)$, with $j = 1, \ldots, m$. Consequently, we can define the residual in the computed solution as

$$\delta_j(t) = H_j(t, x(t)), \text{ for all } j = 1, \ldots, m \text{ and all } t \in [0, 1]. \tag{5.5}$$

Given a small tolerance ε, we require that, along the path, the residual satisfies the tolerance

$$\|\delta(t)\| \leq \varepsilon \ ,$$

for all $t \in [0, 1]$.

Residual control vs. forward control Since the Jacobian $H_x(t, x^e(t))$ is nonsingular for all $t \in [0, 1]$, then, with a smoothness and a compactness argument, there exist $M > 0$ and a neighbourhood in $\mathbb{R} \times \mathbb{C}^m$ of the exact solution path such that

$$\|H_x^{-1}(t, y)\| < M$$

for all (t, y) in that neighbourhood.

Therefore, for small enough tolerance ε,

$$\|x(t) - x^e(t)\| \leq M \|H(t, x(t))\| = M \|\delta(t)\|$$

for all $t \in [0, 1]$.

For solving numerically (5.4), we use Pryce's structural analysis (as described in the previous chapter). We note that Pryce's method, which is based on Taylor series, applies also for the unknown variables and unknown function with complex values (that is $x(t) \in \mathbb{C}^m$ and $H(t, x(t)) \in \mathbb{C}^m$ for all $t \in [0,1]$).

The system Jacobian corresponding to Pryce's structural analysis is precisely

$$\frac{\partial H}{\partial x}(t, x(t)) \tag{5.6}$$

and is non-singular along the solution path since (5.3) is chosen to be a good homotopy. Consequently, Pryce's method applies.

We remark that the offsets of the problem are given by $\mathbf{d} = (0, \ldots, 0), \mathbf{c} = (0, \ldots, 0)$ (each consisting of m elements) and therefore the structural index is 1.

The problem is not a semi-explicit index-1 DAE, though, as required in Chapter 3. However, by considering the time as unknown variable, the problem can be written as a semi-explicit autonomous index-1 DAE

$$\begin{cases} t' &= 1 \\ 0 &= H(t, x) \; . \end{cases} \tag{5.7}$$

The $(m+1) \times (m+1)$ system Jacobian of the problem (5.7) is obviously non-singular and has the particular form

$$\mathcal{J} = \begin{bmatrix} 1 & 0 \\ 0 & H_x \end{bmatrix} .$$

Moreover, the initial condition for the DAE (5.7), $(0, x_0)$, is a consistent point due to the fact that $q(x_0) \; = \; 0$.

Thus Theorem 6 (which holds also for complex–valued functions and unknown variables) applies. According to it, the cost of numerically solving the problem is polynomial in the number of bits of accuracy. Furthermore, due to the linearity with respect to t of the homotopy (5.3), the transformation to an autonomous DAE does not increase much the cost of solving the problem.

Chapter 6

Conclusions and future research

The aim of this thesis was to analyse the computational complexity of numerically solving initial value problems for differential algebraic equations and to show that there exist algorithms of polynomial cost with respect to the number of digits of accuracy required. The model of the standard theory predicted exponential cost in the number of digits of accuracy for solving IVP for ODE. The success of our model is based on more realistic assumptions (piecewise analytic functions rather than fixed order of continuity).

The algorithm of choice was based on the Taylor series method of solving DAE developed by Pryce. The advantage of the Taylor series methods over fixed order methods is that they are highly competitive for tight tolerances, due to their variable order. We also took advantage of the fact that Taylor series can be very efficiently computed numerically with automatic differentiation.

Besides obtaining a polynomial cost of the algorithm for solving IVP for index-1 DAE, we also give a detailed analysis or errors (residual error is preferred in all our models, due to its low cost and simple interpretation).

We also obtained polynomial cost with respect to the number of bits of accuracy for solving IVP for a more general class of DAE, arbitrary high index, fully implicit equations, containing derivatives of any order. Due to its generality, the analysis of the errors is more difficult than for the index-1 case. One general form of smooth enough interpolants are also given.

In both cases, we show that adaptive meshes outperform non-adaptive ones, as observed in practice.

Our motivation for a theoretical study of the complexity of solving DAE came from our belief that these models could help improve practical codes.

There is a growing need of good software for efficient solution of DAE, due to their many important practical applications. But there is a wider class of applications than just the direct practical applications which are naturally modelled as DAE. An interesting example of such an application is also discussed. By applying homotopy continuation methods to a system of polynomials whose isolated roots we want to determine, we transform the problem to a set of IVP for index-1 DAE, whose solutions we want to compute. Our good (polynomial) cost with respect to the number of bits of accuracy of solving DAE gives also a polynomial cost of solving such problems.

A good asymptotic model of the cost sheds light on finite precision and finite order methods. The results obtained with our models match closely the experience of practitioners.

Future work

- We would like to find a better way of handling the constant factor for the complexity of solving IVP for DAE. Specifically, we are looking at geometrical methods: we would like to be able to give a geometric characterization of this constant.

- modelling the complexity of solving infinite dimensional systems such as DDE and PDE

- modelling the complexity of solving hybrid systems, such as mixtures of differential equations and difference equations. In this context, we need to study also the complexity of accurately detecting the boundaries of transit from one type of equations to the other. Of course, a better understanding of singularities is required.

Appendix

Theorem 9 (Gronwall Lemma) *Let x, ω, ξ be real continuous functions defined on an interval $[a,b]$ such that*

$$\begin{aligned} \xi(t) &\geq 0 \, , \\ x(t) &\leq \omega(t) + \int_a^t \xi(s)x(s)ds \end{aligned}$$

for all $t \in [a,b]$. Then the following inequality holds for all $t \in [a,b]$:

$$x(t) \leq \omega(t) + \int_a^t \xi(s)\omega(s)\exp(\int_s^t \xi(\tau)d\tau)ds \; .$$

The notations below are those of Chapter 4. The following theorem gives the reduction of DAE to ODE using structural analysis and can be found in [66].

Theorem 10 (Pryce) *Assume f_i, $i = 1, \ldots, m$ are smooth and $(t^*, x^*) \in \mathbb{R} \times \mathbb{R}^{J\leq 0}$ is a solution of the extended system $f_{I\leq 0}(t^*, x^*) = 0$. If the system Jacobian $J(t^*, x^*)$ is non-singular then there exist a function $\Phi(t, x_{J<0}) \in \mathbb{R}^m$ and some neighbourhood U of $(t^*, x^*_{J<0})$ in $\mathbb{R} \times \mathbb{R}^{J<0}$ such that:*

I. If $x(t)$ is a solution of the ODE

$$x_{J_0} = \Phi(t, x_{J<0}) \tag{1}$$

with initial conditions in U satisfying the stages $k < 0$, then $x(t)$ is a solution of the original DAE, (4.1).

II. There exists a neighbourhood V of (t^*, x^*) in $\mathbb{R} \times \mathbb{R}^{J\leq 0}$ such that if $x(t)$ is the solution of the original DAE (4.1) and if there exist t such that $(t, x_{J\leq 0}(t)) \in V$ then this solution can be extended to a solution of the ODE (1) passing through t^* such that $f_{I\leq 0}(t^*, x_{J\leq 0}(t^*)) = 0$.

Bibliography

[1] E.L. Allgower and K. Georg, *Introduction to Numerical Continuation Methods*, SIAM Classics in Applied Mathematics, Philadelphia, 2003.

[2] E.L. Allgower and K. Georg, Numerical path following, in: P.G. Ciarlet and J.L. Lions, eds., *Techniques of Scientific Computing (Part 2)*, Vol. 5 of *Handbook of Numerical Analysis*, North-Holland, 1997, 3 – 203.

[3] U. Ascher, R.M. Mattheij, and R.D. Russell, *The Numerical Solution of Two-point Boundary Value Problems*, Prentice Hall, Englewood Cliffs, New York, 1988.

[4] R. Barrio, Performance of the Taylor series methos for ODEs/DAEs, *Applied Mathematics and Computation*, in press.

[5] A. Ben-Israel and T.N.E. Greville, *Generalized Inverses, Theory and Applications*, Wiley-Interscience, New York, 1974.

[6] D.N. Bernshtein, The number of roots of a system of equations, *Functional Anal. Appl.* **9**(3) (1975) 183 – 185.

[7] D.P. Bertsekas, *Linear network optimization*, MIT Press, London, 1991.

[8] K.E. Brenan, S.L. Campbell, and L.R. Petzold, *Numerical Solution of IVP in DAE*, North-Holland Publishing Co., New York, 1989.

[9] R.P. Brent, Fast multiple-precision evaluation of elementary functions, *J. Assoc. Comput. Mach* **23**(2) (1976) 242 – 251.

[10] R.P. Brent and H.T. Kung, Algorithms for composition and reversion of power series, in: J.F. Traub, ed., *Analytic Computational Theory*, Academic Press, New York, 1976, 217 – 225.

[11] P.S. Bullen, D.S. Mitrinovic, and P.M. Vasic, *Means and Their Inequalities*, Mathematics and its Applications, East European Series, D. Reidel Publishing Company, Dordrecht, 1988.

[12] J.C. Butcher, *The Numerical Analysis of Ordinary Differential Equations*, John Wiley & Sons Ltd., Chichester, 1987.

[13] S.L. Campbell and C.W. Gear, The index of general nonlinear DAEs, *Numerische Mathematik* **72** (1995) 77 – 94.

[14] S.L. Campbell and E. Griepentrog, Solvability of general differential algebraic equations, *SIAM J. Sci. Comput.* **16**(2) (1995) 257 – 270.

[15] Y.F. Chang and G. Corliss, Ratio-like and recurrence relation tests for convergence of series, *J. Inst. Maths Applics* **25** (1980) 349 – 359.

[16] R.M. Corless, Error backward, *Chaotic Numerics, AMS Contemporary Mathematics Series* **172** (1994) 31 – 62.

[17] R.M. Corless, An elementary solution of a minimax problem arising in algorithms for automatic mesh selection, *SIGSAM Bulletin: Commun. Comput. Algebra* **34**(4) (2001) 7 – 15.

[18] R.M. Corless, A new view of the computational complexity of IVP for ODE, *Numerical Algorithms* **31** (2002) 115 – 124.

[19] R.M. Corless and G. Corliss, Rationale for guaranteed ODE defect control. *Computer Arithmetic and Enclosure Methods (Oldenburg,1991)*, North - Holland, Amsterdam, 1992, 3 – 12.

[20] R.M. Corless, M.W. Giesbrecht, M. van Hoeij, I.S. Kotsireas, and S.M. Watt, Towards factoring bivariate approximate polynomials, *Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation*, ACM, New York, 2002, 37 – 45.

[21] R.M. Corless and S. Ilie, Polynomial cost for solving IVP for high-index DAE, submitted.

[22] R.M. Corless and D.J. Jeffrey, The Turing factorization of a rectangular matrix, *SIGSAM Bulletin* **31**(3) (1997), 20 – 28.

[23] G.F. Corliss, Integrating ODE's along paths on Riemann surfaces, *Proceedings of the Seventh Manitoba Conference on Numerical Mathematics and Computing, Congress. Numer. XX*, Winnipeg, Manitoba, 1977, 279 – 295.

[24] G.F. Corliss, Integrating ODE's in the complex plane-pole vaulting, *Mathematics of Computation* **35**(152) (1980) 1181 – 1189.

[25] G.F. Corliss and Y.F. Chang, Solving ordinary differential equations using Taylor series, *ACM Tansactions on Mathematical Software* **8**(2) (1982) 114 – 144.

[26] G.F. Corliss and W. Lodwick, Role of constraints in the validated solution of DAEs, Tech. Rep. 430, Marquette University, Milwaukee, USA, 1996.

[27] K. Deimling, *Nonlinear functional analysis*, Springer Verlag, Berlin, 1985.

[28] W.H. Enright, A new error control for initial value solvers, *Appl. Math. Comput.* **31** (1989) 288 – 301.

[29] W.H. Enright, Analysis of error control strategies for continuous Runge-Kutta methods, *SIAM J. Num. Anal.* **26**(3) (1989) 588 – 599.

[30] W.H. Enright, Continuous numerical methods for ODEs with defect control, *J. Comp. Appl. Math* **125**(1-2) (2000) 159 – 170.

[31] W.H. Enright, K.R. Jackson, S.P. Nørsett, and P.G. Thomsen, Interpolants for Runge-Kutta formulas, *ACM Transactions on Mathematical Software* **12**(3) (1986) 193 – 218.

[32] W.H. Enright, K.R. Jackson, S.P. Nørsett, and P.G. Thomsen, Effective solution of discontinuous IVPs using a Runge-Kutta formula pair with interpolants, *Appl. Math. Comput.* **27**(4) (1988) 313 – 335.

[33] C.W. Gear, The simultaneous numerical solution of differential algebraic equations, *IEEE Trans. Circuit Theory* **18** (1971) 89 – 95.

[34] C.W. Gear, Differential-algebraic equation index transformations, *SIAM J. Sci. Stat. Comp.* **9** (1988) 39 – 48.

[35] C.W. Gear, Differential algebraic equations, indices and integral algebraic equations, *SIAM J. Num. Anal.* **27** (1990) 1527 – 1534.

[36] C.W. Gear and L. Petzold, ODE methods for the solution of differential algebraic systems, *SIAM J. Numerical Analysis* **21** (1984) 716 – 728.

[37] I. Gladwell, L.F. Shampine, L.S. Baca, and R.W. Brankin, Practical aspects of interpolation in Runge-Kutta codes, *SIAM J. Sci. Stat. Comput* **8**(3) (1987) 322 – 341.

[38] A. Griewank, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, Frontiers in Applied Mathematics, SIAM, Philadelphia, PA, 2000.

[39] E. Hairer, C. Lubich, and M. Roche, *The numerical solution of differential algebraic systems by Runge-Kutta methods*, Lecture Notes in Mathematics **1409**, Springer-Verlag, Berlin, 1989.

[40] E. Hairer, S.P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I*, Springer series in computational mathematics, Vol. 8, Springer, Berlin, 1987.

[41] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II*, Springer series in computational mathematics, Vol. 14, Springer, Berlin, 1991.

[42] J. Hoefkens, Rigorous numerical analysis with high-order Taylor models, PhD dissertation, Michigan State University, 2001.

[43] J. Hoefkens, M. Berz, and K. Makino, Efficient high-order methods for ODEs and DAEs, in: G.F. Corliss et al., eds., *Automatic Differentiation: From Simulation to Optimization*, Springer-Verlag, New York, 2001, 343 – 350.

[44] J. van der Hoeven, Fast evaluation of holonomic functions, *Theoret. Comput. Sci.* **210** (1999) 199 – 215.

[45] J. van der Hoeven, Fast evaluation of holonomic functions near and in regular singularities, *J. Symbolic Comput.* **31**(6) (2001) 717 – 743.

[46] S. Ilie and R.M. Corless, The cost of locating singularities and rank changes, in preparation.

[47] S. Ilie, R.M. Corless, and G. Reid, Numerical solutions of index-1 differential algebraic equations can be computed in polynomial time, *Numerical Algorithms*, in press.

[48] A. Iserles and G. Söderlind, Global bounds on numerical error for ordinary differential equations, *Journal of Complexity* **9** (1993) 97 – 112.

[49] K.R. Jackson and N. Nedialkov, Some recent advances in validated methods for IVPs for ODEs, *Appl. Numer. Math.* **42**(1) (2002) 269 – 284.

[50] R. Jonker and A. Volgenant, A shortest augmenting path algorithm for dense and sparse linear assignment problems, *Computing* **38**(4) (1987) 325 – 340.

[51] B.Z. Kacewicz, On the optimal error of algorithms for solving a scalar autonomous ODE, *BIT* **22** (1982) 503 – 518.

[52] B.Z. Kacewicz, Optimality of Euler-integral information for solving a scalar autonomous ODE, *BIT* **23** (1983) 217 – 230.

[53] B.Z. Kacewicz, How to increase the order to get minimal-error algorithms for systems of ODE, *Numerische Mathematik* **45** (1984) 93 – 104.

[54] W. Kahan, Handheld calculator evaluates integrals, *Hewlett-Packard J.* **31**(8) (1980) 23 – 32.

[55] S.E. Mattsson and G. Söderlind, Index reduction in differential-algebraic equations using dummy derivatives, *SIAM J. Sci. Comput.* **14**(3) (1993) 677 – 692.

[56] F. Mazzia and F. Iavernaro, Test set for initial value problem solvers, Department of Mathematics, University of Bari, http://pitagora.dm.uniba.it/testset/problems.html, 2003.

[57] J.C.P. Miller, *Introduction to tables of Weber parabolic cylinder functions*, National Physical Laboratory, H.M.S.O., London, 1955.

[58] C. Moler, Are we there yet? Zero crossing and event handling for differential equations, *Matlab News & Notes*, 16-17, 1997.

[59] R.E. Moore, *Interval Analysis*, Prentice Hall, Englewood Cliffs, New York, 1966.

[60] N.S. Nedialkov, private communication.

[61] N.S. Nedialkov, software HIDAES (High-Index DAE Solver).

[62] N.S. Nedialkov and J.D. Pryce, Solving differential-algebraic equations by Taylor series (I): computing Taylor coefficients, submitted for publication.

[63] H. Nguyen, Interpolation and error control schemes for DAE using continuous implicit Runge-Kutta methods, Ph.D. thesis, Univ. of Toronto, 1995.

[64] C. Pantelides, The consistent initialization of differential-algebraic systems, *SIAM J. Sci. Stat. Comput.* **9**(2) (1988) 213 – 231.

[65] J.D. Pryce, Solving high-index DAEs by Taylor series, *Numerical Algorithms* **19** (1998) 195 – 211.

[66] J.D. Pryce, A simple structural analysis method for DAEs, *BIT* **41**(2) (2001) 364 – 394.

[67] J.D. Pryce, Design and implementation of a solver for high-index DAEs, talk, *International Conference on Scientific Computation and Differential Equations*, Nagoya, Japan, 2005.

[68] P.J. Rabier and W.C. Rheinboldt, A geometric treatement of implicit differential algebraic equations, *Journal of Differential Equations* **109** (1994) 110 – 146.

[69] L.B. Rall, *Automatic Differentiation: Techniques and Applications*, Springer-Verlag, Berlin, 1981.

[70] G.J. Reid, P. Lin, and A.D. Wittkopf, Differential-elimination completion algorithms for DAE and PDAE, *Studies in Applied Mathematics* **106**(1) (2001) 1 – 45.

[71] L.F. Shampine, *Numerical Solution of Ordinary Differential Equations*, Chapman & Hall, New York, 1994.

[72] L.F. Shampine, Solving ODEs and DDEs with residual control, *Appl. Numer. Math.* **52**(1) (2005) 113 – 127.

[73] L.F. Shampine and S. Thompson, Event location for ordinary differential equations, *Computers and Mathematics with Applications* **39** (2000) 43 – 54.

[74] K. Shub and S. Smale, Complexity of Bézout's theorem V: polynomial time, *Theoretical Computer Science* **133**(1) (1994) 141 – 164.

[75] G. Söderlind, Automatic control and adaptive time-stepping, *Numerical Algorithms* **31** (2002) 281 – 310.

[76] G. Söderlind, Digital filters in adaptive time-stepping, *ACM Trans. Math. Software* **29** (2003) 1 – 26.

[77] A.J. Sommese and C.W. Wampler, *The numerical solution of systems of polynomials arising in engineering and science*, World Scientific Press, Singapore, 2005.

[78] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, Springer Verlag, New York-Heidelberg, 1980.

[79] J.F. Traub, G.W. Wasilkowski, and H. Wozniakowski, *Information-Based Complexity*, Academic Press, New York, 1988.

[80] J.F. Traub and H. Wozniakowski, *A General Theory of Optimal Algorithms*, Academic Press, New York, 1980.

[81] J.F. Traub and H. Wozniakowski, Perspectives on information-based complexity, Bulletin of the American Mathematical Society **26**(1) (1992) 29 – 52.

[82] J. Verschelde, Homotopy continuation methods for solving polynomial systems, Ph.D. thesis, Katholieke Universiteit Leuven, 1996.

[83] G.W. Wasilkovski, Information of varying cardinality, *Journal of Complexity* **2** (1986) 204 – 228.

[84] A.G. Werschulz, *The Computational Complexity of Differential and Integral Equations*, Oxford Science, Oxford, 1991.