KRYLOV DEFERRED CORRECTION METHODS FOR DIFFERENTIAL EQUATIONS WITH ALGEBRAIC CONSTRAINTS

Jun Jia

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Mathematics.

Chapel Hill
2006

Approved by:

Professor Jingfang Huang
Professor M. Gregory Forest
Professor Cass T. Miller
Professor Michael L. Minion
Professor Matthew W. Farthing

ABSTRACT

JUN JIA: Krylov Deferred Correction Methods For Differential Equations With Algebraic Constraints
(Under the direction of Professor Jingfang Huang)

In this dissertation, we introduce a new class of spectral time stepping methods for efficient and accurate solutions of ordinary differential equations (ODEs), differential algebraic equations (DAEs), and partial differential equations (PDEs). The methods are based on applying spectral deferred correction techniques as preconditioners to Picard integral collocation formulations, least squares based orthogonal polynomial approximations are computed using Gaussian type quadratures, and spectral integration is used instead of numerically unstable differentiation. For ODE problems, the resulting Krylov deferred correction (KDC) methods solve the preconditioned nonlinear system using Newton-Krylov schemes such as Newton-GMRES method. For PDE systems, method of lines transpose (MoL^T) couples the KDC techniques with fast elliptic equation solvers based on integral equation formulations and fast algorithms. Preliminary numerical results show that the new methods are of arbitrary order of accuracy, extremely stable, and very competitive with existing techniques, particularly when high precision is desired.

To my grandma and my parents.

ACKNOWLEGEMENTS

I would like to express my greatest gratitude and appreciation to a bunch of people, without whom this thesis would not be possible.

To my mentor Jingfang Huang, for his patient guidance and generosity and for everything he has done for me. To my advising committee, Gregory Forest, Cass T. Miller, Michael L. Minion and Matthew W. Farthing, for all their intelligent advice and helps. To each one in the applied math program at UNC, I extend my deepest thanks.

I am also grateful to my family and my girlfriend Yuan, who have always been there for me.

And finally, I own many thanks to Ruhai Zhou, Anita Layton and all my friends at UNC, and everyone who has supported me during all these years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CFL	Courant-Friedrichs-Lewy
DAE	Differential Algebraic Equation
KDC	Krylov Deferred Correction
MoL	Method of Lines
MoL^T	Method of Lines Transpose
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
SDC	Spectral Deferred Correction
RE	Richards' Equation

Chapter 1

Introduction

In this dissertation, a new spectral time stepping strategy is developed for accurate and efficient solutions of ordinary and partial differential equations with algebraic constraints. It couples spectral deferred correction (SDC) ideas, Newton-Krylov methods, and integral equation formulation based fast elliptic equation solvers. Preliminary numerical experiments show that the resulting numerical methods, including the Krylov deferred correction methods for ordinary differential equations (ODEs) and differential algebraic equations (DAEs), and the method of lines transpose (MoL^T) for partial differential equations (PDEs), can achieve an arbitrary order of accuracy and are competitive with existing schemes.

This research is inspired by the following observations.

1.1 Stable and High Order Initial Value ODE and DAE Solvers

In many respects, the construction of efficient and stable methods for solving initial value problems governed by systems of ODEs and DAEs is considered a mature subject [3, 13, 14, 25, 35, 36, 37, 48, 62]. Existing methods for such problems include the backward differentiation formulas (BDF) based package DASSL developed by Petzold et al., which is applicable to ODEs as well as DAEs of index 0 and 1 [13, 62]; and the Runge-Kutta

based RADAU developed by Hairer et al. which can be applied to ODEs and DAE problems of index up to 3 [35, 37]. Detailed discussions of these available solvers as well as a test set can be found in [1] and the references therein.

Although great progress has been made in the last century, the disadvantages of these existing schemes are also becoming obvious. These include (a) the step-sizes of the solvers are often constrained by the stability properties, in particular, for ODE systems derived by discretizing PDEs using the method of lines (MoL) approach, unacceptably small time steps may have to be used due to the Courant-Friedrichs-Lewy (CFL) condition or stability issues; and (b) higher order (> 10) versions of existing ODE solvers either lack desired stability properties or may experience efficiency problems.

Recently, Dutt et al.[21] proposed higher order and stable numerical schemes for ODE initial value problems, which couple the Picard integral equation formulation, Gauss quadrature, and deferred correction ideas. Numerical results show that the resulting spectral deferred correction (SDC) methods can achieve very high order of accuracy (> 40) and are very competitive with existing schemes for ODE initial value problems. Instead of giving the details of the SDC strategy, which will be discussed in later chapters, we note that although successful, numerical experiments also reveal that when the SDC methods are applied to stiff ODE systems, order reduction is observed[51]; and when the SDC ideas are extended to DAE problems, the methods may no longer converge.

One purpose of this dissertation is to study the mechanism for order reduction and divergence, and search for possible remedies in order to develop stable, higher order and efficient ODE and DAE initial value problem solvers.

1.2 Spectral Methods and Initial Value Problems

Spectral methods have been shown to be advantageous for many problems compared with other techniques including the finite difference and finite element methods, especially

when higher order accuracy is required and the solution is "smooth." Also, by using the orthogonal "optimal" basis, much less computer storage is required [76, 72, 26, 56, 24, 12].

However, spectral methods are mainly developed for boundary value problems. In particular, for time dependent PDE problems, there exist a class of methods in which spectral techniques are only applied to the spatial direction, while the temporal direction is solved by low-order time stepping schemes. There are certain circumstances under which the solutions can be resolved by low-order methods in time but require higher order approximations in space. For most problems, however, it may be beneficial to apply spectral approximations in both directions. Unfortunately, as far as we know, no spectral or pseudo-spectral methods have been implemented for initial value problems.

Spectral formulations for initial value problems do exist. In [34], it was shown that when Gaussian nodes are used in the collocation formulation (which is also a Runge-Kutta method), the method is super convergent (when p nodes are used, the method is order $2p$), symplectic (area preserving), symmetric (structure preserving), A-stable and B-stable. Also, for fixed time step sizes, the numerical error decays exponentially fast as a function of the number of terms p in the approximation, and hence "pseudo-spectral." Previous research shows that direct solution of such collocation formulations is extremely inefficient for large p, and existing numerical implementations are limited to order less than 10 or so.

In this dissertation, we couple the SDC ideas with Newton Krylov methods to improve the efficiency when solving "pseudo-spectral" formulations. This strategy leads to a new class of spectral methods for initial value problems.

1.3 Fast Elliptic Solvers, Method of Lines and Rothe's Method

Accurate and efficient numerical schemes for boundary value elliptic equations are an active area of research. In the last twenty years, great progress has been made for ODE boundary value problems and constant coefficient elliptic PDEs using integral equation formulations and fast algorithms. In particular, robust, adaptive and extremely efficient solvers exist for two-point boundary value ODEs of the form

$$u''(x) + p(x)u'(x) + q(x)u(x) = f(x)$$

as presented in 1997 by Greengard and Lee [52]; and fast multipole methods have been developed for constant coefficient elliptic equations (see, e.g., [28, 18, 23, 29]). The details of these methods will be discussed in the following chapters.

As for time dependent PDEs, two different sequential approximation approaches coexist for a long time in the numerical analysis community. The first approach is the well-known method of lines (MoL) [68] which discretizes the spatial direction first using finite difference, finite element or spectral methods, and the resulting system of ODEs for the temporal direction is solved by well developed ODE initial value problem solvers. The second approach, on the other hand, first discretizes the temporal direction, leading to coupled elliptic equations at different time steps, which can be solved using standard finite element or finite volume techniques. The second strategy is less known and sometimes referred to as Rothe's method [65, 10, 49].

We want to mention that although the resulting algebraic systems may be identical for both discretization strategies, especially when fixed time step sizes and time independent spatial grids are used, yet their efficiency, accuracy, adaptive implementation (in both time and space), and ease of programming are different. The MoL approach has

traditionally been the preferred method due to available adaptive and optimized ODE initial value problem solvers. However, as pointed out in [10], adaptive mesh generation (or regridding) is in general difficult in MoL. Also, as the PDE structure in space is ignored by the ODE system, efficient elliptic equation solvers can not be easily adapted. Rothe's method, on the other hand, easily allows for adaptive grids when coupled with finite element or finite volume type solvers. For an extended discussion of the MoL and Rothe approach (for reaction-diffusion type equations) the interested reader is referred to [20]. However, as the resulting equations after temporal discretization are coupled at different time steps, the method might be very inefficient, especially in higher order versions.

In this thesis, we study how Rothe's method can be accelerated using KDC techniques. In our scheme, we first discretize the temporal direction, and then apply the KDC ideas to the coupled elliptic equation system so that each time step only requires the solution of a decoupled elliptic boundary value problem. Finally and most importantly, instead of the finite element/volume methods commonly used in Rothe's method, the elliptic equations are solved efficiently using integral equation methods accelerated by fast algorithms. This new approach will be fully adaptive, high-order, and efficient. We refer to this technique as method of lines transpose (MoL^T). As most problems of current interests have multiple scales in both temporal and spatial directions. We believe the proposed MoL^{T}can also yield efficient and accurate solutions to multiscale problems.

1.4 Outline of Dissertation

This dissertation is organized as follows. In Chapter 2, several mathematical and numerical preliminaries, including spectral deferred correction methods, Newton-Krylov schemes, and fast elliptic equation solvers, are presented. These are the fundamental building blocks for the construction of the new time stepping strategies. In Chapter 3,

we analyze the original spectral deferred correction methods using results from numerical linear algebra and matrix theory, and show that the original SDC strategy is equivalent to a Neumann series expansion for a special preconditioned system. We then briefly discuss how Krylov subspace methods can be introduced to accelerate SDC methods for ODE problems. Preliminary numerical results are presented to show the efficiency and accuracy of the new methods for ODE problems. In Chapter 4, the new techniques are extended to general DAE problems by introducing the details of the Krylov deferred correction (KDC) methods. We show analytically and numerically that with the Krylov subspace acceleration methods, the original SDC methods become convergent for DAE problems and order reduction can be removed. The KDC technique is further extended in Chapter 5 for PDE problems; the resulting MoL^{T}couples existing fast elliptic equation solvers with KDC technique. Numerical results show that MoL^{T}is of arbitrary order in both temporal and spatial directions. Finally, a brief summary of results and recommendations are discussed in Chapter 6.

Chapter 2

Preliminaries

In this chapter, necessary building blocks are introduced to construct the Krylov Deferred Correction (KDC) methods for ODE and DAE problems, and the method of lines transpose (MoL^T) for time dependent PDE systems.

2.1 Spectral Deferred Correction Methods

The first building block we consider is the spectral deferred correction (SDC) technique first introduced by Dutt et. al. in 2000 [21]. It is designed for accurate and efficient solutions of ordinary differential equation (ODE) initial value problems of the form

$$\varphi'(t) = F(t, \varphi(t)) \qquad t \in [0, T] \tag{2.1}$$

$$\varphi(0) = \varphi_0 \ , \tag{2.2}$$

where $\varphi_0, \varphi(t) \in \mathbb{C}^N$ and $F : \mathbb{R} \times \mathbb{C}^N \to \mathbb{C}^N$.

The construction of efficient and stable methods for solving ODE initial value problems has been considered a mature subject and great progress has been made in the last century. Existing methods include the backward differentiation formulas (BDF) based package DASSL and the Runge-Kutta based RADAU. Detailed discussions of these available solvers as well as a test set can be found in [1] and the references therein. However although successful, the disadvantages of existing schemes are also becoming obvious as

mentioned in Chapter 1. These include (a) the step-sizes of the solvers are often constrained by stability properties; and (b) higher order (> 10) versions of existing ODE solvers either lack desired stability properties or are extremely complicated to solve, in particular, as far as we know, no spectral or pseudo-spectral implementations are available for initial value problems.

In order to improve the accuracy and stability of existing time stepping schemes, in 2000 [21], by coupling (a) the classical defect and deferred correction methods [60, 80, 81]; (b) the Picard integral equation formulations; and (c) orthogonal polynomials and the corresponding Gaussian quadratures, Dutt et al. [21] introduced the spectral deferred correction (SDC) strategy for the construction of stable explicit and implicit methods with extremely high order of accuracy. As this technique is not widely known, in the following, we first summarize the basic ideas of the SDC technique, and then describe the technical details in subsequent sections.

As with classical deferred and defect correction methods, a single time step of an SDC method begins by first dividing the time step $[t_n, t_{n+1}]$ into a set of intermediate sub-steps defined by the points $\vec{t} = [t_0, t_1, \cdots, t_p]$ with $t_n = t_0 < t_1 < \cdots < t_p \leq t_{n+1}$. For simplicity, we assume $t_n = t_0 = 0$ in the following discussions. Next, a provisional approximation $\vec{\varphi}^{[0]} = [\varphi^{[0]}(t_0), \varphi^{[0]}(t_1), \cdots, \varphi^{[0]}(t_p)]$ is computed at the intermediate points using a standard numerical method, e.g. the explicit Euler method for non-stiff problems or the implicit Euler method for stiff problems [21]. Applying standard approximation or interpolation theory, the continuous counterpart of $\vec{\varphi}^{[0]}$ can be constructed and is represented as $\varphi^{[0]}(t)$. Using $\varphi^{[0]}(t)$, an equation for the error $\delta(t) = \varphi(t) - \varphi^{[0]}(t)$ is then constructed. This correction equation for $\delta(t)$ can be approximated using a similar low order method, and an improved numerical solution is constructed. This procedure can then be repeated resulting in a sequence of approximate solutions.

To construct the correction equation, the classical deferred and defect correction methods rely on differentiation of $\varphi^{[0]}(t)$ to form an ODE for $\delta(t)$, where $\varphi^{[0]}(t)$ is the

interpolating polynomial of $\vec{\varphi}^{[0]}$. On the other hand, SDC methods utilize the Picard integral equation formulation of the ODE

$$\varphi(t) = \varphi_0 + \int_0^t F(\tau, \varphi(\tau))d\tau \tag{2.3}$$

to construct a corresponding integral equation for $\delta(t)$. Specifically

$$\delta(t) = \int_0^t \left[F(\tau, \varphi^{[0]}(\tau) + \delta(\tau)) - F(\tau, \varphi^{[0]}(\tau))\right] d\tau + \epsilon(t) \tag{2.4}$$

where

$$\epsilon(t) = \varphi_0 + \int_0^t F(\tau, \varphi^{[0]}(\tau))d\tau - \varphi^{[0]}(t). \tag{2.5}$$

The discretizaton of these equations will be discussed in more detail in the following section, but for now note that the discretization of $\epsilon(t)$ is simply a numerical integration. It is for this reason that the points $\vec{t}$ which define the sub-steps in SDC methods are chosen to be Gaussian quadrature nodes so the numerically stable spectral integration technique can be applied [27]. The integral equation formulation for $\delta(t)$ in Eq. (2.4) coupled with spectral integration rules allows SDC methods to overcome the loss of stability of classical deferred/defect correction methods as the order of the method increases. For a detailed discussion of the different choices of quadrature nodes see [51].

Deferred correction methods based on the Picard integral formulation and spectral integration are of interest for several reasons, most notably because of the relative ease with which one can theoretically construct methods with arbitrarily high order of accuracy. Preliminary numerical tests presented in [21] suggest that SDC methods are competitive with the best existing ODE initial value problem solvers, especially for stiff problems or where high accuracy is required. Furthermore the stability regions of the implicit methods are close to optimal and do not degrade with increased orders of accuracy [51]. Semi-implicit and multi-implicit variations of SDC methods have also been

presented, which enable the construction of very high-order methods for equations with both stiff and non-stiff components [11, 58, 59]. Also noted in these papers, however, is the fact that when SDC methods are applied to very stiff equations, the effective order of accuracy of the method is reduced for values of the time step above a certain threshold. This type of order reduction, which is also present in many popular types of Runge-Kutta methods [15, 19, 67], means that, although the methods are stable for larger time steps, one must use a very small time step for the method to converge with full order. To understand the order reduction and accelerate the convergence, in the following, we present the technical details of the SDC methods.

2.1.1 Picard Integral Equation and Error Equation

Consider the Picard integral equation representation of the ODE initial value problem given in Eq. (2.3). Suppose an approximate solution $\varphi^{[0]}(t)$ to Eqs.(2.1-2.2) is given, and define the error $\delta(t)$ as

$$\delta(t) = \varphi(t) - \varphi^{[0]}(t). \tag{2.6}$$

Substituting (2.6) into (2.3) yields

$$\varphi^{[0]}(t) + \delta(t) = \varphi_0 + \int_0^t F(\tau, \varphi^{[0]}(\tau) + \delta(\tau))d\tau. \tag{2.7}$$

To reduce notational clutter here and in the following the time dependence of the second argument of F will be implicitly assumed, e.g. $F(t, \varphi^{[0]}(t))$ is written simply as $F(t, \varphi^{[0]})$. Now consider the residual function

$$\epsilon(t) = \varphi_0 + \int_0^t F(\tau, \varphi^{[0]})d\tau - \varphi^{[0]}(t), \tag{2.8}$$

which simply gives the error in the Picard equation (2.3). Rearranging Eq. (2.7) and using Eq. (2.8) gives a Picard-type integral equation for the error

$$\delta(t) = \int_0^t [F(\tau, \varphi^{[0]} + \delta) - F(\tau, \varphi^{[0]})]\, d\tau + \epsilon(t). \tag{2.9}$$

Note that unlike the classical deferred or defect correction methods in [60, 80, 81], the equation for $\delta(t)$ is not written here as an ODE.

2.1.2 Euler's Methods on Gaussian Quadrature Nodes

Deferred correction methods proceed by iteratively solving the error equation (2.9) using a low order method to improve the provisional solution $\vec{\varphi}^{[0]}$. To describe the time stepping procedure, suppose as before that the time step interval $[t_n, t_{n+1}]$ has been subdivided using the points $t_0, t_1, t_2, \cdots, t_p$ such that

$$t_n = t_0 < t_1 < t_2 \cdots < t_p \leq t_{n+1}. \tag{2.10}$$

Note that Eq. (2.9) gives the identity

$$\delta(t_{m+1}) = \delta(t_m) + \int_{t_m}^{t_{m+1}} [F(\tau, \varphi^{[0]} + \delta) - F(\tau, \varphi^{[0]})]\, d\tau + \epsilon(t_{m+1}) - \epsilon(t_m). \tag{2.11}$$

Letting δ_m denote the numerical approximation to $\delta(t_m)$ (and likewise for $\varphi_m^{[0]}$ and ϵ_m), a simple discretization of Eq. (2.11) similar to the explicit Euler (forward Euler) method for ODEs is

$$\delta_{m+1} = \delta_m + \Delta t_m \left(F(t_m, \varphi_m^{[0]} + \delta_m) - F(t_m, \varphi_m^{[0]})\right) + \epsilon_{m+1} - \epsilon_m, \tag{2.12}$$

where $\Delta t_m = t_{m+1} - t_m$. Similarly, an implicit scheme for the solution based on the backward Euler method is

$$\delta_{m+1} = \delta_m + \Delta t_m \left(F(t_{m+1}, \varphi_{m+1}^{[0]} + \delta_{m+1}) - F(t_{m+1}, \varphi_{m+1}^{[0]}) \right) + \epsilon_{m+1} - \epsilon_m. \tag{2.13}$$

Denoting the "low order" approximation of $\delta(t)$ by $\vec{\delta}^{[1]} = [\delta_1, \delta_2, \cdots, \delta_p]$, a refined solution is given by $\vec{\varphi}^{[1]} = \vec{\varphi}^{[0]} + \vec{\delta}^{[1]}$. In order to complete the discretization, we must specify how the terms ϵ_m are computed.

2.1.3 Spectral Integration Matrix

First notice that there are various ways to choose the points $t_0, t_1, t_2, \cdots, t_p$ to define the sub-steps in the SDC method. When Gaussian quadrature nodes are used, $\{t_1, \cdots, t_p\}$ are interior points in $[t_n, t_{n+1}]$ and the endpoints are not used. On the other hand the Radau Ia quadrature nodes $t_0, t_1, t_2, \cdots, t_p$ use the left end point while the Radau IIa nodes $t_1, t_2, \cdots, t_p$ have $t_p = t_{n+1}$. Finally, the Lobatto quadrature rule requires the use of both end points.

Using the Gaussian nodes as an example, suppose we are given the scalar function values $\vec{\varphi} = \{\varphi_1, \varphi_2, \cdots, \varphi_p\}$ at the nodes, then the Legendre polynomial expansion

$$L^p(\vec{\varphi}, t) = \sum_{k=0}^{p-1} c_k L_k(t)$$

can be constructed to approximate $\vec{\varphi}$ where the coefficients are computed using Gaussian quadrature rules. This gives a numerically stable and efficient way to find the equivalent interpolating polynomial of degree $p-1$. Integrating this interpolating polynomial analytically from t_0 to t_m, a linear mapping Q is derived, which maps the function values $\vec{\varphi}$

to the integral of the interpolating polynomial

$$[\vec{\varphi}]_m = \int_{t_0}^{t_m} L^p(\vec{\varphi}, \tau) d\tau.$$

This can be written in matrix form

$$Q\vec{\varphi} = \Delta t S \vec{\varphi}, \tag{2.14}$$

where S will be referred to as the integration matrix, and is independent of Δt. Note that in the more general case where $\varphi(t) \in \mathbb{C}^N$, Eq. (2.14) must be interpreted as being applied component-wise to $\vec{\varphi}$, i.e. $\vec{\varphi}$ is a vector of length Np and

$$Q\vec{\varphi} = \Delta t (I_p \otimes S)\vec{\varphi}, \tag{2.15}$$

where I_p is the identity matrix of size $p \times p$. In the following, we use script font to denote this tensor product, i.e. $\mathcal{S}$ denotes the $Np \times Np$ block diagonal matrix $I_p \otimes S$. The matrix S can be precomputed using Mathematica to any specified accuracy.

2.1.4 The Spectral Deferred Correction Algorithm

For traditional deferred/defect correction methods, there are two factors that prevent the use of extremely high order methods: The first problem relates to the instability of interpolation at equispaced nodes where the Runge phenomenon can be observed when the number of interpolation points p is large. The second problem is that numerical differentiation of the original ODE formulation (Eqs. 2.1-2.2) introduces instabilities [77]. Spectral deferred correction methods avoid both of these difficulties by introducing Gaussian-type nodes and using the Picard integral equation. The procedure is explained next.

Given an approximate solution $\vec{\varphi}^{[0]} = [\varphi_1^{[0]}, \cdots, \varphi_p^{[0]}]$, consider the error equation given

by (2.9). Discretizing the integral in (2.8) using the spectral integration matrix yields

$$\vec{\epsilon} = \vec{\varphi_0} + \Delta t \mathcal{S} \vec{F} - \vec{\varphi}^{[0]}, \tag{2.16}$$

where $\vec{\epsilon} = [\epsilon(t_1), \cdots, \epsilon(t_p)]$ is the residual at the intermediate points. Once the residual is calculated, an approximation $\vec{\delta}^{[1]}$ to the error equation (2.9) is computed using p steps of the Eq. (2.12) for non-stiff problems or Eq. (2.13) for stiff problems. The provisional solution is then updated with $\vec{\varphi}^{[1]} = \vec{\varphi}^{[0]} + \vec{\delta}^{[1]}$, and this procedure can be repeated. The algorithm for SDC is given by the following:

Pseudo-code: Spectral Deferred Correction Method

Comment [Compute initial approximation]

For non-stiff/stiff problems, use the forward/backward Euler method to compute an approximate solution $\varphi_m^{[0]} \approx \varphi(t_m)$ at the sub-steps $t_1, \ldots, t_p$ on the interval $[t_n, t_{n+1}]$.

Comment [Compute successive corrections.]

do j = 1,. . . ,J

1) Compute the approximate residual function $\vec{\epsilon}$ using $\vec{\varphi}^{[j-1]}$ and Eq. (2.16).

2a) For non-stiff problems, compute $\vec{\delta}^{[j]}$ using p steps of Eq. (2.12).

2b) For stiff problems, compute $\vec{\delta}^{[j]}$ using p steps of Eq. (2.13).

3) Update the approximate solution $\vec{\varphi}^{[j]} = \vec{\varphi}^{[j-1]} + \vec{\delta}^{[j]}$.

end do

It can be shown that each correction procedure in this algorithm can improve the order of the method by one, as long as such improvement has not gone beyond the degree of the underlying interpolating polynomial and the quadrature rules [32, 39, 38]. For linear ODE problems, a proof will be provided in Sec. 3.2 utilizing the Neumann series expansion.

2.2 Newton-Krylov Methods

In this section, we discuss the Newton-Krylov (inexact Newton) methods for the efficient solutions of linear and nonlinear algebraic equations. This is the second building block of the KDC and MoL^{T}methods.

2.2.1 Krylov Subspace Methods

Given a matrix A and a vector b, the Krylov subspace is defined as

$$\mathcal{K}_m(A, b) = span\{b, Ab, \cdots, A^m b\}.$$

Define $r_m = b - Ax_m$, with $x_0 = 0$, the generalized minimal residual (GMRES) algorithm works by searching for the "best" solution of the equation $Ax = b$ in the Krylov subspace that either makes $r_m \perp \mathcal{K}_m$ or minimizes r_m in L_2 norm. In general, the convergence rate of the algorithm depends on the eigenvalue distribution of the matrix A as well as the initial guess x_0. We cite the following pseudo-code from [66].

The Generalized Minimal Residual Method

Comment [Compute initial approximation]

Choose initial guess x_0 and compute $r_0 = b - Ax_0$ and $v_1 = r_0/\|r_0\|$.

Comment [Arnoldi's method: Iterative Orthogonalization]

do j = 1,. . . ,k

1) $h_{i,j} = (Av_j, v_i)$, i=1,2, . . . , j

2) $\hat{v}_{j+1} = Av_j - \sum_{i=1}^{j} h_{i,j} v_i$

3) $h_{j+1,j} = \|\hat{v}_{j+1}\|$

4) $v_{j+1} = \hat{v}_{j+1}/h_{j+1,j}$

enddo

comment [Form the approximate solution]

Choose the approximate solution x_k using Hessenberg matrix $[h_{i,j}]$ and the basis $\{v_i\}$.

Note that the memory required by the GMRES method increases linearly with the iteration number k, and the number of multiplications scales like $\frac{1}{2}k^2n$ where n is the number of unknowns and the size of the matrix A is $n \times n$. When k is chosen to be n, a full orthogonalization cycle is implemented and in theory $b - Ax_n$ should be close to machine precision. Although accurate, this procedure is expensive and requires excessive memory storage. For practical reasons, instead of a full orthogonalization procedure, GMRES can be restarted every k_0 steps where $k_0 < n$ is some fixed integer parameter. The restarted version is often denoted as GMRES(k_0). Interested readers are referred to the original paper [66] for further discussions.

Because of the excessive work and storage requirements of GMRES, alternative Krylov subspace methods have also been developed in the last century. Specifically, we want to mention the biconjugate gradients stabilized (BiCGStab) method and transpose-free quasi-minimal residual (TFQMR) algorithm for non-symmetric linear systems (See [7] for a summary of existing Newton-Krylov methods). The storage required in both methods is

independent of iteration number k, and the number of multiplications grows only linearly as a function of k.

2.2.2 Newton-Krylov Methods

Consider a general nonlinear algebraic system $M(x) = 0$ with N equations and unknowns. Suppose an approximate solution x_0 is known. Newton's method can be used to iteratively compute a sequence of quadratically convergent approximations (assuming the Jacobian matrix J_M is nonsingular at the solution)

$$x_{n+1} = x_n - \delta x,$$

where δx is the solution of the linear equation

$$J_M(x_n)\delta x = b \tag{2.17}$$

with $b = M(x_n)$ and $J_M(x_n)$ the Jacobian matrix of $M(x)$ at x_n. As the matrix J_M is in general a dense matrix, computing the solution of this linear equation with Gaussian elimination requires $O(N^3)$ operations. Instead, the Krylov subspace methods can be applied. Note that the iterations in Newton's method and the Krylov subspace methods can be intertwined, and the resulting methods are usually referred to as the Newton-Krylov methods or the inexact Newton methods. The readers are referred to [46, 45, 66] for detailed discussions.

Notice that for a general $M(x) = 0$, the number of Krylov subspace iterations may be large and hence direct application of the Newton-Krylov methods may be inefficient. However, it is noticed that for many special systems, the amount of work required by the Krylov subspace methods to find the solution of Eq. (2.17) can be greatly reduced.

Consider the case

$$J_M(x_n) = \pm I - C,$$

where most of the eigenvalues of C are clustered close to 0. Because of the rapid decay of most eigenmodes in $C^q b$, the Krylov subspace methods converge extremely efficiently.

In summary, an efficient numerical implementation of a Newton-Krylov method depends on two things:

(a) A formulation of the problem $M(x) = 0$ such that J_M is close to the identity matrix $\pm I$.

(b) An efficient procedure for computing the matrix vector product Cb (or equivalently $J_M b$).

For (a), the preconditioning strategy is generally applied and for (b) the forward difference approximation, which will be briefly discussed next.

2.2.3 Preconditioning and Forward Difference Approximation

One common technique to improve the convergence of the Krylov subspace methods is to apply a "preconditioner" to the original system so that the Jacobian matrix of the new system is closer to the identity matrix. Traditionally, such preconditioners are chosen as sparse matrices close to J_M^{-1} [17]. Dense integral operators have also been used as preconditioners (see e.g. [47]), which are efficiently applied to an arbitrary vector using fast convolution algorithms such as the fast multipole method (FMM) [28]. One of the main themes of this dissertation is that the SDC procedure in which lower-order methods are used to produce a higher-order solution, is equivalent to using a lower-order approximation process of a particular equation as a preconditioner for a higher-order method. This presents a different way to understand the deferred correction methods discussed in Sec. 2.1, and allows one to easily adapt existing Newton-Krylov methods to accelerate the convergence of the SDC method as will be discussed later.

In regards to the computation of the matrix vector product $J_M b$, as the Jacobian J_M is not always easy to derive, a general forward difference approximation technique can be adapted as in most Newton-Krylov solvers where for any vector v, $J_M(x)v$ is approximated by

$$D_h M(x : v) = \left(M(x + hv) - M(x)\right)/h$$

for some properly chosen parameter h (h may be complex). This difference approximation technique as well as the choice of h have been carefully studied previously and the readers are referred to [46] for details.

2.3 Fast Elliptic Equation Solvers

Accurate and efficient numerical schemes for boundary value elliptic equations are an active area of research. One motivation behind the thrust of this research stems from the recent success of integral equation methods (IEM) for ODE boundary value problems and constant coefficient elliptic PDEs, and related matrix compression based fast direct solvers. In the following, we discuss these fast solvers as our last building block. This fundamental building block may be considered as the driving force for developing the novel time stepping scheme for PDE problems where the fast integral equation methods are applied to the elliptic equation systems in space.

2.3.1 ODE Two-Point Boundary Value Problems

We first consider the two-point boundary value ODEs of the form

$$u''(x) + p(x)u'(x) + q(x)u(x) = f(x),$$

for which a robust, adaptive, and extremely efficient algorithm was presented in 1997 by Greengard and Lee [52]. In their algorithm, the solution is first represented as $u(x) =$

$u_h(x) + u_i(x)$ where u_i is a simple linear function and u_h satisfies the homogeneous boundary conditions. More specifically, u_h solves the equation

$$u_h''(x) + p(x)u_h'(x) + q(x)u_h(x) = \tilde{f}(x), \tag{2.18}$$

where

$$\tilde{f}(x) \equiv f(x) - (u_i''(x) + p(x)u_i'(x) + q(x)u_i(x)).$$

Notice that the Green's function for the operator on the left of Eq. (2.18) is not readily available, therefore, we consider the Green's function $G_0(x,t)$ for a simpler problem

$$\phi''(x) + q_0(x)\phi(x) = 0 \tag{2.19}$$

for some given q_0, and represent u_h as the convolution of $G_0(x,t)$ with an unknown density function $\sigma(x)$ as

$$u(x) = \int_a^c G_0(x,t) \cdot \sigma(t)dt. \tag{2.20}$$

A well-conditioned second kind integral equation is then derived for $\sigma(x)$

$$\sigma(x) + \tilde{p}(x)\int_a^c G_1(x,t)\sigma(t)dt + \tilde{q}(x)\int_a^c (12)G_0(x,t)\sigma(t)dt = \tilde{f}(x), \tag{2.21}$$

where $\tilde{p}(x) = p(x)$, $\tilde{q}(x) = q(x) - q_0(x)$, and

$$G_1(x,t) = \frac{d}{dx}G_0(x,t). \tag{2.22}$$

We want to mention that for general $q_0(x)$, the Green's function usually takes the form

$$G_0(x,t) = \begin{cases} g_l(x)g_r(t)/s, & \text{if } x \le t; \\ g_l(t)g_r(x)/s, & \text{if } x \ge t, \end{cases}$$

where s is given by $s = g_l(x)g_r'(x) - g_l'(x)g_r(x)$. In practice, we can easily construct the Green's functions for $q_0 = 0$ or $q_0 = -1$.

To facilitate the discussion, next we introduce the operator $P : L^2[a,c] \to L^2[a,c]$ as

$$\begin{aligned} P\eta(x) &= \eta(x) + \tilde{p}(x)\int_a^c G_1(x,t)\eta(t)dt + \tilde{q}(x)\int_a^c G_0(x,t)\eta(t)dt. && (2.23)\\ &= \eta(x) + \psi_l(x)\int_a^x g_l(t)\eta(t)dt + \psi_r(x)\int_x^c g_r(t)\eta(t)dt, && (2.24)\end{aligned}$$

where

$$\begin{aligned} \psi_l(x) &= (\tilde{p}(x)g_r'(x) + \tilde{q(x)}g_r(x))/s, && (2.25)\\ \psi_r(x) &= (\tilde{p}(x)g'(x) + \tilde{q(x)}g_l(x))/s. && (2.26)\end{aligned}$$

Thus, Eq. (2.18) becomes

$$P\sigma = \tilde{f}. \tag{2.27}$$

As the linear system from the discretization of P is dense, finding $\sigma(x) = P^{-1}\tilde{f}(x)$ directly is computationally unattractive. Instead, by studying the "restricted" operator of P in the subinterval $B = [b_l, b_r]$ of $[a,c]$, a divide and conquer strategy can be applied. Note that when $b_l = a$ (or $b_r = c$), b_r(or b_l) divides $[a,c]$ into 2 subintervals. We denote the restriction of η to B by η_B and the restriction of P to B by $P_B : L^2(B) \to L^2(B)$. More specifically, for $x \in B$,

$$\begin{aligned} P_B\eta_B(x) &= \eta_B(x) + \tilde{p}(x)\int_{b_l}^{b_r} G_1(x,t)\eta_B(t)dt + \tilde{q}(x)\int_{b_l}^{b_r} G_0(x,t)\eta_B(t)dt && (2.28)\\ &= \eta_B(x) + \psi_l(x)\int_{b_l}^{x} g_l(t)\eta_B(t)dt + \psi_r(x)\int_x^{b_r} g_r(t)\eta_B(t)dt. && (2.29)\end{aligned}$$

Note that for $x \in B$, we also have

$$P\sigma(x) = P_B\sigma_B(x) - \psi_l(x)\mu_l^B - \psi_r(x)\mu_r^B = \tilde{f}(x), \tag{2.30}$$

where μ_l^B and μ_r^B are given by

$$\mu_l^B = -\int_a^{b_l} g_l(t)\sigma(t)dt, \tag{2.31}$$

$$\mu_r^B = -\int_{b_r}^{c} g_r(t)\sigma(t)dt. \tag{2.32}$$

Rearrange the terms we have

$$\sigma_B(x) = P_B^{-1}\tilde{f}(x) + \mu_l^B P_B^{-1}\psi_l(x) + \mu_r^B P_B^{-1}\psi_r(x). \tag{2.33}$$

The computational cost of P_B^{-1} is much less expensive compared to solving Eq. (2.27) directly. We can therefore construct an efficient solver by dividing $[a, c]$ into a large number of subintervals $B_1, B_2, \ldots, B_M$ and applying $P_{B_M}^{-1}$ locally to each subinterval. We still need to figure out the coefficients $\mu_l^{B_i}$ and $\mu_r^{B_i}$ for each B_i, which can be done by utilizing the following theorem.

Theorem 2.1 *Suppose the subinterval B is subdivided into a left and a right subinterval, denoted by D and E. Then the coefficients $\mu_l^D, \mu_r^D, \mu^D, \mu_l^E, \mu_r^E$ and μ^E of*

$$P_B\eta_B = \mu_l^B\psi_l + \mu_r^B\psi_r + \mu^B\tilde{f}, \tag{2.34}$$

$$P_D\eta_D = \mu_l^D\psi_l + \mu_r^D\psi_r + \mu^D\tilde{f}, \tag{2.35}$$

$$P_E\eta_E = \mu_l^E\psi_l + \mu_r^E\psi_r + \mu^E\tilde{f}, \tag{2.36}$$

satisfy

$$\begin{aligned}
\mu^D &= \mu^E = \mu^B \\
\mu_l^D &= \mu_l^B \\
\mu_r^E &= U_r^B \\
\begin{pmatrix} \mu_r^D \\ \mu_r^E \end{pmatrix} &= \begin{pmatrix} 1 & \alpha_r^E \\ \beta_r^D & 1 \end{pmatrix}^{-1} \begin{pmatrix} \mu_r^B(1-\beta_r^E) - \mu^B\delta_r^E \\ \mu_l^B(1-\alpha_l^D) - \mu^B\delta_l^D \end{pmatrix},
\end{aligned} \tag{2.37}$$

and

$$\begin{aligned}
\alpha_l^B &= \frac{(1-\alpha_l^E)(\alpha_l^D - \beta_l^D\alpha_r^E)}{\Delta} + \alpha_l^E, \\
\alpha_r^B &= \frac{(1-\beta_r^D)(\alpha_r^E - \alpha_l^D\alpha_r^E)}{\Delta} + \alpha_r^D, \\
\beta_l^B &= \frac{(1-\beta_r^E)(\beta_l^D - \beta_l^D\alpha_l^E)}{\Delta} + \beta_l^E, \\
\beta_r^B &= \frac{(1-\beta_r^D)(\beta_r^E - \beta_l^D\alpha_r^E)}{\Delta} + \beta_r^D, \\
\delta_l^B &= \frac{1-\alpha_l^E}{\Delta}\delta_l^D + \delta_l^E + \frac{(\alpha_l^E-1)\alpha_l^D}{\Delta}\delta_r^E, \\
\delta_r^B &= \frac{1-\beta_r^D}{\Delta}\delta_r^E + \delta_r^D + \frac{(\beta_r^D-1)\alpha_r^E}{\Delta}\delta_l^D,
\end{aligned} \tag{2.38}$$

where $\Delta = 1 - \alpha_r^E\beta_l^D$ *and*

$$\begin{aligned}
\alpha_l^X &\equiv \int_X g_l(t)P_X^{-1}\psi_l(t)dt, \qquad \alpha_r^X \equiv \int_X g_r(t)P_X^{-1}\psi_l(t)dt, \\
\beta_l^X &\equiv \int_X g_l(t)P_X^{-1}\psi_r(t)dt, \qquad \beta_r^X \equiv \int_X g_r(t)P_X^{-1}\psi_r(t)dt, \\
\delta_l^X &\equiv \int_X g_l(t)P_X^{-1}\tilde{f}(t)dt, \qquad \delta_r^X \equiv \int_X g_r(t)P_X^{-1}\tilde{f}(t)dt,
\end{aligned} \tag{2.39}$$

and X *denote a subinterval of* $[a, c]$.

For completeness, the algorithm is briefly described as follows:

Pseudo-code: Direct Adaptive ODE Boundary Value Problem Solver

Comment [Step 1: Generate subinterval binary tree]

Starting from the root interval $[a, c]$, subdivide each interval into two subintervals recursively until each leaf node is small enough for the restricted integral equation to be computed directly and inexpensively. We refer to each subinterval on the finest level as a leaf node.

Comment [Step 2: Solve the restricted integral equations on leaf nodes]

For each leaf node B_i, compute $P_{B_i}^{-1}\psi_l, P_{B_i}^{-1}\psi_r$ and $P_{B_i}^{-1}\tilde{f}$, and generate α, β and δ by definition (2.39).

Comment [Step 3: Compute α, β and δ]

α, β and δ are needed to compute μ. On the leaf nodes, these numbers are available from step 2. For other levels, they are generated by (2.38).

Comment [Step 4: Compute μ]

Note that μ for the root interval is essentially $\mu_l^B = \mu_r^B = 0$ and $\mu^B = 1$. Once we have all the α, β and δ for each node available, we can generate μ for all intervals from formula (2.37).

Comment [Step 5: Construct global solution]

Now that μ's are available for all the leaf nodes, we can construct the global solution by equation (2.33) and results from step 2.

We leave the implementation details of this method to [52], and conclude this section by listing several remarkable features of the solver: (a) The method is direct and very robust. The adaptive strategy requires no *a priori* information of the solution, and the solution is resolved to a specified accuracy; (b) The method is extremely efficient. With N grid points in a given mesh structure, the number of operations is asymptotically optimal

$O(N)$ with a small prefactor; and more remarkably, (c) the adaptive code requires at most about twice as much work as a non-adaptive code that is simply given the final resolved mesh structure as input.

2.3.2 New Version Fast Multipole Methods

The algorithm in [52] has been extremely successful for ODE (1D) problems. However in higher dimensions, although the local solutions can be obtained easily using integral equation ideas, they can not be patched together as efficiently as in the ODE case. In the last twenty years, alternative approaches in d (> 1) dimensions using IEM and fast algorithms have been a hot research topic and great progress has been made. In particular, we want to mention the new version of FMM accelerated fast IEM solver for

$$\nabla^2 u(\mathbf{x}) - \beta^2 u(\mathbf{x}) = f(\mathbf{x})$$

developed by Cheng et al. in 2006 [18]. This equation is often referred to as the *Yukawa* equation, which comes from the fact that the free space Green's function for this equation is the Yukawa or screened Coloumb potential. β is often referred to as the screening parameter and its inverse the electron Debye length. This PDE is important in steady state physics and biochemistry.

For simplicity, assume that Dirichlet boundary conditions $g(x)$ are imposed on the boundary $\partial\Omega$ (Neumann or Robin boundary conditions are also possible). Similar to the ODE case in [52], the solution is first decomposed into $u = \tilde{u} + \psi$ where $\tilde{u}$ satisfies

$$\nabla^2 \tilde{u}(\mathbf{x}) - \beta^2 \tilde{u}(\mathbf{x}) = f(\mathbf{x}) \text{ for } \mathbf{x} \in \Omega, \qquad \tilde{u}(\mathbf{x}) = \tilde{g}(\mathbf{x}) \text{ for } \mathbf{x} \in \partial\Omega,$$

and $\psi(\mathbf{x})$ satisfies the homogeneous Yukawa equation

$$\nabla^2 \psi(\mathbf{x}) - \beta^2 \psi(\mathbf{x}) = 0 \text{ for } \mathbf{x} \in \Omega, \qquad \psi(\mathbf{x}) = g(\mathbf{x}) - \tilde{g}(\mathbf{x}) \text{ for } \mathbf{x} \in \partial\Omega.$$

The IEM strategy is to compute $\tilde{u}$ through the convolution of f with the **free space** Yukawa Green's function $G(\mathbf{x}, \mathbf{y})$

$$\tilde{u}(\mathbf{x}) = \int_{\Omega} G(\mathbf{x}, \mathbf{y}) f(\mathbf{y}) d\mathbf{y}.$$

Clearly $\tilde{u}$ satisfies the original Yukawa equation, however since the free space Green's function is used in the convolution, the boundary values of $\tilde{u}(x)$ will not be $g(x)$ but rather some function $\tilde{g}(x)$. Next the equation for ψ is solved through the classical potential theory approach. Namely

$$\psi(\mathbf{x}) = \int_{\partial\Omega} \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial \hat{n}_{\mathbf{y}}} \sigma(\mathbf{y}) dy.$$

The boundary potential $\sigma(y)$ is not known *a priori*, but satisfies the Fredholm integral equation of second kind

$$\frac{1}{2}\sigma(\mathbf{x}) + \int_{\partial\Omega} \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial \hat{n}} \sigma(\mathbf{y}) d\mathbf{y} = g(\mathbf{x}) - \tilde{g}(\mathbf{x}).$$

As this equation is well conditioned, it can be solved after a few iterations using Krylov subspace based iterative methods such as GMRES.

Notice that the above solution process requires the explicit evaluation of a volume convolution integral, and a solution of a well conditioned integral equation by iteration in which the explicit evaluation of a boundary convolution integral is required. These can be done efficiently using the new version of FMM, which has $O(N)$ complexity with an optimized prefactor as discussed in [18].

There are several attractive features of the fast algorithms accelerated IEM methods. First, integration based methods are almost trivially spatially adaptive, and using higher-order quadrature rules does not significantly effect the complexity or efficiency of the method unlike in finite difference methods where large stencils result. Second, the

condition number of the boundary integral equation does not depend significantly on the grid spacing, and the error in the numerical solution can be strictly controlled since it depends on the error in the integration of the right-hand side. Hence extremely accurate results are possible, even when the right-hand side is not smooth. Third, the complexity of the boundary has only a small effect on the efficiency or accuracy of the method. Fourth, since a large part of the computational work in the IEM is spatially local, the IEM is highly parallelizable. In fairness, there are some disadvantages to this approach as well, and the most significant is perhaps the complexity of the algorithms, which makes it extremely difficult to implement, especially in three dimensions. Nevertheless, as the following example shows, the efficiency and accuracy of this approach is impressive when compared with a FFT-based solver in two dimensions. The integral equation solver utilizes adaptive mesh refinement, as opposed to the uniform grid FFT solvers, so the total time as well as the time per grid point for each method are reported. In Table 2.1, the

N	T_{hwscrt}	E_2	$Rate$
$256^2 = 65,536$	0.17	$9.4\ 10^{-4}$	$3.8\ 10^5$
$512^2 = 262,144$	0.78	$2.4\ 10^{-4}$	$3.4\ 10^5$
$1024^2 = 1,048,576$	4.0	$5.9\ 10^{-5}$	$2.6\ 10^5$
$2048^2 = 4,194,304$	19.4	$1.5\ 10^{-5}$	$2.2\ 10^5$

Table 2.1: Timing results for HWSCRT.

averaged results for the FFT based second-order HWSCRT [74, 73] are listed. Here, N denotes the number of grid points, T_{hwscrt} denotes the required solution time in seconds, E_2 denotes the relative L_2 error of the computed solution, and $Rate$ denotes the number of grid points "processed" per second (N/T_{hwscrt}). The analytical solution is given by $\psi(\mathbf{x}) = \sum_{i=1}^{3} e^{-\alpha|\mathbf{x}-\mathbf{x}_i|^2}$ where $\mathbf{x}_1 = (.1, .1)$, $\mathbf{x}_2 = (0, 0)$, $\mathbf{x}_3 = (-.15, .1)$, and $\alpha = 250$. We choose $\beta = 0.1$, and the right hand side is found accordingly. The timing results for the 4th order Yukawa equation solver are given in Table 2.2. Here, ϵ_{FMM} denotes the requested precision from far-field interactions within the FMM, ϵ_{RHS} denotes the requested relative precision in discretizing the right-hand side. Compared with Table

N	T_{fmm}	ϵ_{FMM}	ϵ_{RHS}	E_2	$Rate$
11,488	0.14	10^{-3}	10^{-3}	$9.7\ 10^{-4}$	$7.7\ 10^{4}$
96,592	1.08	10^{-3}	10^{-6}	$1.0\ 10^{-3}$	$8.9\ 10^{4}$
96,592	1.82	10^{-6}	10^{-6}	$7.5\ 10^{-7}$	$5.3\ 10^{4}$
821,824	11.26	10^{-6}	10^{-9}	$7.0\ 10^{-7}$	$7.3\ 10^{4}$
821,824	16.77	10^{-9}	10^{-9}	$7.8\ 10^{-10}$	$4.9\ 10^{4}$

Table 2.2: Timing results for the 4th order fast Yukawa solver

2.1, the FMM solvers only require about $3 \sim 6$ times more CPU time per point than the FFT based methods. More significantly, due to the grid adaptivity, the increase in cost is more than made up for by an increase in accuracy, i.e. the FMM method provides much greater accuracy for a comparable computational cost.

2.3.3 Fast Iterative and Direct Solvers

In recent years, the new version of FMM accelerated fast IEM solvers have been developed for many linear constant coefficient elliptic equations, including the Laplace, Poisson, Yukawa, Helmholtz, Stokes, and biharmonic equations. For variable coefficient problems, however, results are still limited. In the following, we discuss recent progress along this direction using iterative techniques and fast direct methods.

For iterative techniques, the fundamental idea is to use an integral equation operator based on a given Green's function to precondition the variable coefficient problem. In 1994, Strain proposed such a scheme for general elliptic equations with periodic boundary conditions, in which the solution is represented as

$$u(\mathbf{x}) = \int_{\Omega} K(\mathbf{x}, \mathbf{y}) \mu(\mathbf{y}) d\mathbf{y}$$

for some known Green's function $K(\mathbf{x}, \mathbf{y})$ and unknown potential $\mu(\mathbf{y})$. Once μ is known, u can be efficiently computed, hence the original problem of computing $u(\mathbf{x})$ is changed to that of computing $\mu(\mathbf{x})$. In [71], K is chosen to be the Green's function for the

"averaged" constant coefficient elliptic equation, and the procedure is performed in the frequency domain. Analytical results show that the new equation system is Fredholm second kind (as the Green's function captures the analytical information of a "nearby" problem), which can be solved by the Krylov iterative methods. However, as noted in our numerical experiments, when the variable coefficients have different scales in space, the (bounded) number of iterations in the Krylov subspace methods may be extremely large and hence the resulting numerical approach is inefficient. Recently in [44], the integral equation idea is coupled with domain decomposition schemes, in which the domain is decomposed into regions on which the coefficients vary only marginally. In each region, the solution can then be computed efficiently by the integral equation strategy similar to [71]. To "glue" individual solutions, a strategy similar to the imposition of boundary conditions using potential theory can be applied. The iterative methods have shown great promise but are still being studied.

The second class of methods tries to directly solve the linear equations resulting from the discretization of the integral equation formulation for the variable coefficient problems, by utilizing the special structure of the matrices. The fundamental observation is that many blocks in the matrix are "low separation rank," hence the matrix can be compressed and inverted recursively. This idea has already been successfully applied to the boundary integral equations in 2005 by Martinsson and Rokhlin [55], and the resulting algorithm is asymptotically $O(N)$ where N is the number of nodes in the discretization. Because the method is direct, it can be applied to relatively ill-conditioned systems. Once the inversion is constructed, its application to multiple right hand sides, which often arise in optimization problems, is extremely efficient.

Both iterative and direct methods are promising methods for variable coefficient problems. As a preliminary comparison, numerical experiments in [55] show that in two-dimensional space, a single FMM matrix vector multiply is about 15–20 times faster than the direct matrix inversion. Thus, an iterative method is more efficient if it requires

less than 15–20 iterations. However in the fast direct method, once the inversion is constructed, it can be applied to an additional right hand side in about one tenth of the time required for a single FMM accelerated matrix vector multiply.

Chapter 3

Understanding Spectral Deferred Correction Methods

In this chapter[1], by introducing numerical linear algebra and matrix theory, we study the convergence mechanism of the SDC methods for ODE initial value problems, and try to explain the order reduction phenomenon which happens when the SDC methods are applied to stiff ODE initial value problems.

This chapter is organized as follows. In Sec. 3.1, we study the limit of the SDC methods. In Sec. 3.2, we rewrite the SDC methods in matrix form, and show that for linear problems, the original SDC is equivalent to the preconditioned Neumann series expansion. In Sec. 3.3, we describe how the convergence of the original SDC methods can be easily accelerated for both linear and non-linear problems by introducing Krylov subspace methods. In Sec. 3.4, we present the stability and accuracy analyses for the accelerated SDC methods, and finally in Sec. 3.5, we numerically demonstrate the improved accuracy and stability of the accelerated methods using several linear and nonlinear examples.

[1]Parts of this chapter are reprinted from Journal of Computational Physics, Volume 214, Issue 2, Jingfang Huang, Jun Jia and Michael Minion, "Accelerating the convergence of spectral deferred correction methods", Pages 633-656, Copyright 2006, with permission from Elsevier

3.1 Collocation Formulation: Limit of Iterations

The original spectral deferred correction (SDC) method described in previous chapter can be considered as an iterative scheme. In this section, we consider the limit of the SDC iterations.

For a fixed time step of size $\Delta t = t_{n+1} - t_n$, observe that if the correction iteration in the SDC method converges, then

$$\epsilon(t) = \varphi_0 + \int_0^t F(\tau, \varphi^{[0]}(\tau))d\tau - \varphi^{[0]}(t)$$

will approach zero at the Gaussian nodes $\vec{t} \in [t_n, t_{n+1}]$. Hence the resulting limit solution will satisfy the collocation (or pseudo-spectral) approximation of the Picard equation

$$\varphi(t) = \varphi_0 + \int_0^t F(\tau, \varphi(\tau))d\tau$$

given by

$$\vec{\varphi} = \vec{\varphi_0} + \Delta t \mathcal{S} \vec{F}, \tag{3.1}$$

where

$$\vec{F} = [F(t_0, \varphi(t_0)), F(t_1, \varphi(t_1)), \ldots F(t_p, \varphi(t_p))]^T,$$

$\vec{\varphi_0}$ is a vector of initial conditions

$$\vec{\varphi_0} = [\varphi(t_0), \varphi(t_0), \ldots, \varphi(t_0))]^T,$$

and $\mathcal{S}$ is the spectral integration matrix [26, 27] corresponding to the Gaussian nodes $\vec{t}$ discussed in section 2.1.3. Conditions specifying when the SDC method converges to this limit for linear systems will be presented in section 3.2.

Since Eq. (3.1) couples the solution values at each of the sub-steps defined by $\vec{t}$, for $\varphi(t) \in \mathbb{C}^N$ and assuming p interior points are used in each time step, the total

number of unknowns in the collocation formula is $M = pN$. Therefore a direct solution of this equation using Newton's method requires inverting a matrix of size $M \times M$ at each Newton iteration step. In contrast, each correction iteration of the SDC method requires solving p linear or nonlinear systems with N unknowns. When the number of iterative corrections is small, the SDC methods will be more efficient compared with the direct Newton's method approach, especially when the order p is high. Historically, direct Newton's method (or simplified Newton's method) for the collocation formulation in Eq. (3.1) has been limited to $p < 10$ or so in existing implementations.

3.2 Spectral Deferred Corrections in Matrix Form

To better understand the SDC methods, in this section, we consider the linear ODE system given by

$$\phi'(t) = F(t, \varphi(t)) = L\varphi(t) + f(t), \tag{3.2}$$

where L is a constant matrix. Given an approximate solution $\varphi^{[0]}(t)$, the discretized collocation formulation for the error equation in (2.9) becomes

$$\vec{\delta} - \Delta t \mathcal{S}\mathcal{L}\vec{\delta} = \vec{\varphi_0} + \Delta t \mathcal{S}\vec{F} - \vec{\varphi}^{[0]},$$

where $\mathcal{L} = I_p \otimes L$ (see section 2.1.3). Denoting the right hand side by $\vec{\epsilon}$, the SDC procedure iteratively approximates the solution of

$$(\mathcal{I} - \Delta t \mathcal{S}\mathcal{L})\vec{\delta} = \vec{\epsilon} \tag{3.3}$$

using combination of the low order approximations $\vec{\delta}^{[j]}$ for $j = 1, 2, \cdots$. The goal of this section is to rewrite SDC methods in a matrix form and show that the original SDC technique is equivalent to solving Eq. (3.3) using a preconditioned Neumann series

expansion, i.e., $\vec{\delta} = \sum_{j=1}^{\infty} \vec{\delta}^{[j]}$ where $\vec{\delta}^{[j+1]} = \mathcal{C}\vec{\delta}^{[j]}$ for an explicit matrix $\mathcal{C}$. This analysis proves the convergence of the SDC methods for linear problems, and also reveals how the order reduction can happen for stiff ODE systems.

3.2.1 Euler's Methods in Matrix Form

First, consider the forward Euler method in Eq. (2.12) which is appropriate for non-stiff problems. For the linear correction Eq. (3.3), a sub-step is given by

$$\delta_{m+1} = \delta_m + \Delta t_m L\delta_m + (\epsilon_{m+1} - \epsilon_m). \tag{3.4}$$

Summing successive values of δ and using the fact that both the error $\delta(t)$ and the residual $\epsilon(t)$ are zero at t_0, some manipulation gives

$$\delta_{m+1} = \sum_{i=1}^{m} \Delta t_i L\delta_i + \epsilon_{m+1}. \tag{3.5}$$

Notice that $\sum_{i=1}^{m} \Delta t_i L\delta_i$ is the composite rectangular rule approximation (where the left end point is used) of the integral

$$\int_0^{t_{i+1}} L\delta(s)ds.$$

Therefore, in matrix form, the forward Euler method is equivalent to solving

$$\left(\mathcal{I} - \Delta t\tilde{\mathcal{S}}\mathcal{L}\right)\vec{\delta}^{[1]} = \vec{\epsilon}, \tag{3.6}$$

where $\vec{\delta}^{[1]} = [\delta_1, \delta_2, \cdots \delta_p]^T$, $\vec{\epsilon} = [\epsilon_1, \epsilon_2, \cdots, \epsilon_p]^T$, and

$$\Delta t\tilde{S} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ \Delta t_1 & 0 & \cdots & 0 & 0 \\ \Delta t_1 & \Delta t_2 & \cdots & 0 & 0 \\ \Delta t_1 & \Delta t_2 & \cdots & 0 & 0 \\ \cdot & \cdot & \cdots & 0 & 0 \\ \Delta t_1 & \Delta t_2 & \cdots & \Delta t_{p-1} & 0 \end{bmatrix}. \tag{3.7}$$

Notice that $\tilde{S}$ is a strictly lower triangular approximation of the spectral integration matrix S. Similarly, for the implicit Euler method, the matrix $\tilde{S}$ takes the form

$$\Delta t\tilde{S} = \begin{bmatrix} \Delta t_0 & 0 & \cdots & 0 & 0 \\ \Delta t_0 & \Delta t_1 & \cdots & 0 & 0 \\ \Delta t_0 & \Delta t_1 & \cdots & 0 & 0 \\ \cdot & \cdot & \cdots & 0 & 0 \\ \Delta t_0 & \Delta t_1 & \cdots & \Delta t_{p-2} & 0 \\ \Delta t_0 & \Delta t_1 & \cdots & \Delta t_{p-2} & \Delta t_{p-1} \end{bmatrix}. \tag{3.8}$$

This lower triangular matrix is also an approximation of the spectral integration matrix, with non-zero diagonal entries.

To summarize, each correction in the SDC method may be considered as solving an approximation of the collocation formulation of the correction equation (3.3), where the spectral integration matrix is approximated by a lower triangular matrix. Clearly, the solution given by

$$\vec{\delta}^{[1]} = \left(\mathcal{I} - \Delta t\tilde{\mathcal{S}}\mathcal{L}\right)^{-1}\vec{\epsilon} \tag{3.9}$$

is a low order approximation of $\vec{\delta}$ in Eq. (3.3).

3.2.2 Neumann Series

Suppose after k corrections, we have a provisional approximation $\vec{\varphi}^{[k]}$, the new residual is then defined as

$$\vec{\epsilon} = \vec{\varphi}_0 + \Delta t \mathcal{S}\mathcal{L}\vec{\varphi}^{[k]} - \vec{\varphi}^{[k]}.$$

Applying Euler's method (which is equivalent to Eq. (3.9)) and denoting the solution by $\vec{\delta}^{[k+1]}$, the relationship between $\vec{\varphi}^{[k+1]}$ and $\vec{\varphi}^{[k]}$ is

$$\begin{aligned}
\vec{\varphi}^{[k+1]} &= \vec{\varphi}^{[k]} + \vec{\delta}^{[k+1]} \\
&= \vec{\varphi}^{[k]} + \left(\mathcal{I} - \Delta t \tilde{\mathcal{S}}\mathcal{L}\right)^{-1} \vec{\epsilon} \\
&= \vec{\varphi}^{[k]} + \left(\mathcal{I} - \Delta t \tilde{\mathcal{S}}\mathcal{L}\right)^{-1} \left(\vec{\varphi}_0 - (\mathcal{I} - \Delta t \tilde{\mathcal{S}}\mathcal{L})\vec{\varphi}^{[k]} + \Delta t(\mathcal{S} - \tilde{\mathcal{S}})\mathcal{L}\vec{\varphi}^{[k]}\right) \\
&= \left(\mathcal{I} - \Delta t \tilde{\mathcal{S}}\mathcal{L}\right)^{-1} \vec{\varphi}_0 + \mathcal{C}\vec{\varphi}^{[k]}, \qquad (3.10)
\end{aligned}$$

where we define

$$\mathcal{C} = \left(\mathcal{I} - \Delta t \tilde{\mathcal{S}}\mathcal{L}\right)^{-1} \Delta t(\mathcal{S} - \tilde{\mathcal{S}})\mathcal{L}. \qquad (3.11)$$

It is also straightforward to derive the recursive relationship between $\vec{\delta}^{[k+1]}$ and $\vec{\delta}^{[k]}$. First note that

$$\vec{\varphi}^{[k+1]} = \left(\mathcal{I} - \Delta t \tilde{\mathcal{S}}\mathcal{L}\right)^{-1} \vec{\varphi}_0 + \mathcal{C}\vec{\varphi}^{[k]},$$

and

$$\vec{\varphi}^{[k]} = \left(\mathcal{I} - \Delta t \tilde{\mathcal{S}}\mathcal{L}\right)^{-1} \vec{\varphi}_0 + \mathcal{C}\vec{\varphi}^{[k-1]}.$$

Subtracting the two identities yields

$$\vec{\delta}^{[k+1]} = \mathcal{C}\vec{\delta}^{[k]}. \qquad (3.12)$$

Assuming our initial provisional approximation is given by $\vec{\varphi}^{[0]}$, then from the recursive relation (3.12), the solution after k corrections is given by the Neumann series expansion:

$$\vec{\varphi}^{[k]} = \vec{\varphi}^{[0]} + \sum_{m=1}^{k} \mathcal{C}^{m-1}\vec{\delta}^{[1]}. \tag{3.13}$$

We can also derive the Neumann series expansion by solving the error equation (3.3). Multiplying both sides by $\left(\mathcal{I} - \Delta t\tilde{\mathcal{S}}\mathcal{L}\right)^{-1}$, we have the preconditioned linear system

$$\left(\mathcal{I} - \Delta t\tilde{\mathcal{S}}\mathcal{L}\right)^{-1}(\mathcal{I} - \Delta t\mathcal{S}\mathcal{L})\vec{\delta} = \left(\mathcal{I} - \Delta t\tilde{\mathcal{S}}\mathcal{L}\right)^{-1}\vec{\epsilon}. \tag{3.14}$$

Notice that the right hand side of (3.14) is $\vec{\delta}^{[1]}$ and the operator on the left is

$$\left(\mathcal{I} - \Delta t\tilde{\mathcal{S}}\mathcal{L}\right)^{-1}(\mathcal{I} - \Delta t\mathcal{S}\mathcal{L}) = (\mathcal{I} - \mathcal{C}), \tag{3.15}$$

where $\mathcal{C}$ is defined in Eq. (3.11). Hence, the preconditioned error equation is given by the linear system

$$(\mathcal{I} - \mathcal{C})\vec{\delta} = \vec{\delta}^{[1]}.$$

As $\tilde{\mathcal{S}}$ is an approximation of the matrix $\mathcal{S}$, when Δt is small, we expect the norm of $\mathcal{C}$ to be small. If so, the solution to the linear system is given by the Neumann series expansion

$$\vec{\delta} = \vec{\delta}^{[1]} + \mathcal{C}\vec{\delta}^{[1]} + \mathcal{C}^2\vec{\delta}^{[1]} + \cdots. \tag{3.16}$$

This is clearly equivalent to Eq. (3.13).

There are two immediate consequences of the Neumann series expansion:

Corollary 3.1 *For linear problems, given a sufficiently small fixed time-step Δt, the correction iteration in the SDC method using either of the first order correction procedures described by Eq.(2.12) or (2.13) is convergent.*

Corollary 3.2 *For linear problems, given a sufficiently small fixed time-step Δt, each*

iteration of the correction equation in the SDC method using either of the first order correction procedures described by Eq.(2.12) or (2.13) increases the formal order of the method by one order of Δt, provided the order is not greater than that of the underlying quadrature rule.

The proof of both corollaries follows directly from Eq. (3.16) and the fact that $\mathcal{C}$ in Eq. (3.11) is $O(\Delta t)$.

3.3 Accelerating SDC Methods

In previous section, we showed that for linear problems, an SDC method may be considered as an iteration scheme for solving the collocation formulation (3.3) using a preconditioned Neumann series expansion. In this section, we show how this fact can be used to accelerate the convergence of the original SDC method.

3.3.1 GMRES Acceleration for Linear Problems

For linear problems, consider the preconditioned linear system in Eq. (3.14). The original SDC approximates this equation using a Neumann series expansion in the matrix $\mathcal{C}$ defined in Eq. (3.11) . Since the matrix $\mathcal{C}$ contains a factor of Δt, if Δt is sufficiently small (and hence the expansion is convergent), each additional term in the expansion produces an additional order of accuracy in the approximation. Note however that when the norm of any eigenvalue of $\mathcal{C}$ is greater than 1, the series expansion is divergent. Also, if the norm is smaller but close to 1, the series expansion will still converge, but will do so slowly. The latter case is the cause of order reduction for stiff problems which will be further analyzed numerically in Sec. 3.5.1.

It is straightforward to apply Krylov subspace methods such as GMRES or GMRES(k_0) to the linear system in Eq. (3.14) and to hence find the optimal solution in the Krylov subspace. Using the GMRES as an example, note that each GMRES iteration requires a

matrix vector product be computed. In the present context, this requires the evaluation of

$$\left(\mathcal{I} - \Delta t\tilde{\mathcal{S}}\mathcal{L}\right)^{-1}(\mathcal{I} - \Delta t\mathcal{S}\mathcal{L})\vec{x}_0$$

for any given $\vec{x}_0$. However, applying this operator is equivalent to time marching with either the forward or backward Euler method for the correction equation. The full algorithm is as follows:

Pseudo-code: Matrix Vector Product Algorithm

Comment [Suppose input $\vec{x}_0$ is given.]

1) Calculate $\vec{\epsilon} = (\mathcal{I} - \Delta t\mathcal{S}\mathcal{L})\vec{x}_0$.

2a) Use the forward Euler method and solve $\left(\mathcal{I} - \Delta t\tilde{\mathcal{S}}\mathcal{L}\right)\vec{y} = \vec{\epsilon}$
where $\Delta t\tilde{\mathcal{S}}$ is defined in Eq. (3.7).

2b) Use the backward Euler method and solve $\left(\mathcal{I} - \Delta t\tilde{\mathcal{S}}\mathcal{L}\right)\vec{y} = \vec{\epsilon}$
where $\Delta t\tilde{\mathcal{S}}$ is defined in Eq. (3.8).

3) Output $\vec{y}$.

In this algorithm, the first step is equivalent to evaluating the residual function, and the second step is equivalent to time-stepping the correction equation. Therefore, the amount of work for each matrix vector product in the GMRES accelerated SDC methods is the same as one correction in the original SDC method. However, depending on k_0, GMRES accelerated SDC requires additional work to search the optimal solution in the Krylov subspace. Notice that **no additional function evaluations** are required in this searching process, and so we are expecting minimal efficiency loss due to the use of GMRES. Additional storage is necessary, however, and this could prove to be

prohibitive when applying the GMRES acceleration to PDE problems, and alternative Krylov subspace methods may have to be used as will be discussed in later chapters.

3.3.2 Nonlinear Problems

The GMRES accelerated SDC methods can be applied to nonlinear problems as well. This requires the coupling of Newton iterations with the GMRES accelerated SDC technique for linear problems. In our following discussion in this chapter, we use a "linearly implicit" formulation as described in [21]. In this formulation, notice that for small $\delta(t)$, the error equation (2.9) can be approximated by

$$\delta(t) = \int_a^t J_{\varphi^{[0]}}(s, \varphi^{[0]})\delta(s)\, ds + \epsilon(t) + O(||\delta||^2), \tag{3.17}$$

where $J_{\varphi^{[0]}}$ is the Jacobian matrix of the function $F(t, \varphi^{[0]})$ defined as

$$J_{\varphi^{[0]}}(t, \varphi^{[0]}) = \frac{\partial F(t, \varphi^{[0]})}{\partial \varphi}.$$

Discretizing Eq. (3.17) yields the linear system

$$(I - \Delta t \mathcal{S}\mathcal{J})\vec{\delta} = \vec{\epsilon}, \tag{3.18}$$

where $\mathcal{J}$ is the tensor form of $J_{\vec{\varphi}^{[0]}}$ which represents the Jacobian matrix at each Gaussian node. Since this equation is of the same type as Eq. (3.9), it can be solved using the GMRES accelerated SDC methods for linear problems discussed in the previous section. The Jacobian matrix $\mathcal{J}$ is updated after the linear problem is solved to a prescribed precision tol_G, as described by the following:

Pseudo-code: Nonlinear GMRES accelerated SDC Method

Comment [Compute initial approximation]

Use the Euler method to compute an approximate solution $\vec{\varphi}^{[0]}$.

Comment [Compute successive corrections.]

while residual $\|\vec{\epsilon}\| > tol$ **do**

1) Compute the Jacobian matrix $J_{\vec{\varphi}^{[0]}}$.

2) Use GMRES accelerated SDC for linear problems to solve Eq. (3.18) to tolerance tol_G.

3) Update the approximate solution $\vec{\varphi}^{[0]} = \vec{\varphi}^{[0]} + \vec{\delta}$.

end do

Note that a generalization to this linear implicit algorithm (which couples Newton iterations with GMRES accelerated SDC) is to implement the method under the "inexact Newton Methods" framework [46]. This will be discussed in the more general case of differential algebraic equations in next chapter.

3.4 Stability and Accuracy Analysis

Consider the model problem

$$\begin{aligned} \varphi'(t) &= \lambda \cdot \varphi(t) \qquad\qquad t \in [0,1] \\ \varphi(0) &= 1 \ , \end{aligned} \tag{3.19}$$

following the terminology in [21], the amplification factor, $Am(\lambda)$, for $\lambda \in \mathbb{C}$ is defined by the formula

$$Am(\lambda) = \widetilde{\varphi}(1) \tag{3.20}$$

where $\widetilde{\varphi}(1)$ is the numerical solution at $t = 1$ using $\Delta t = 1$. If, for a given value of λ,

$$| Am(\lambda) | \leq 1, \tag{3.21}$$

then the numerical method is said to be stable for that value of λ. When a numerical method is applied to the model problem, the *stability region* is defined to be the subset of the complex plane consisting of all λ such that the amplification factor defined in Eq. (3.20) satisfies $|Am(\lambda)| \leq 1$.

The most interesting stability diagrams are generated by the GMRES accelerated SDC schemes based on the forward (**explicit**) Euler method. In figure 3.1, we show the stability regions for the restarted GMRES(k_0) using 4 Radau IIa nodes. For $k_0 = 0$, this gives the original SDC, and when $k_0 = 4$, GMRES(k_0) is equivalent to the full GMRES which solves the collocation formulation. It can be seen that the stability region of the GMRES accelerated SDC method is much larger than that of the original SDC method. This is not surprising if one considers the preconditioned system (3.14): Even though the explicit Euler method is a bad preconditioner for λ with large negative real part, the GMRES procedure can still converge to the collocation solution as long as the preconditioning process does not produce numerical overflow. This suggests the possibility of using explicit GMRES accelerated SDC methods for mildly stiff problems. However, we want to mention that when more substeps are used, the explicit Euler based preconditioner is more likely to encounter overflow problems. Hence the stability region will be much smaller. This can be seen in figure 3.2 where 10 Radau IIa nodes are used.

For **implicit** GMRES accelerated SDC methods (using the backward Euler scheme) where the preconditioner is well conditioned, when the full GMRES is performed, the stability regions can be considered the same as those of the corresponding collocation method. A-stability of these methods can be proven in some cases (all collocation methods using the Gaussian points are A-stable), and appears to be true for many others

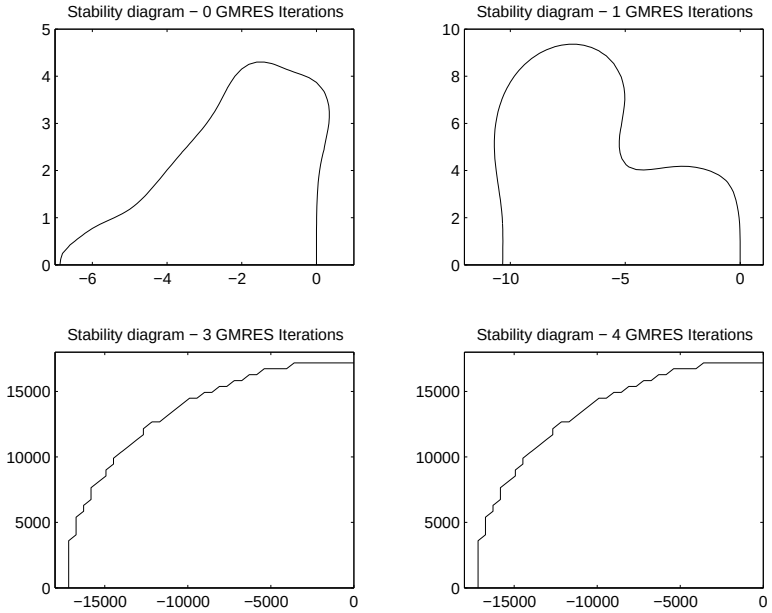

Figure 3.1: Stability region of GMRES(k_0), 4 Radau IIa nodes

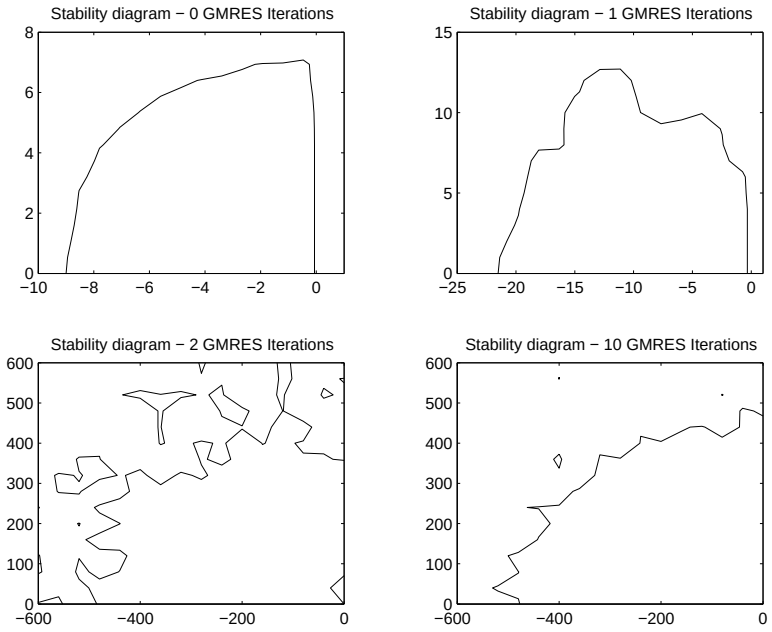

Figure 3.2: Stability region of GMRES(k_0), 10 Radau IIa nodes

based on numerical results [8]. Our numerical results also show that all the implicit GMRES accelerated SDC methods using Radau IIa nodes (up to machine precision) are A-stable. Further stability and convergence analysis for the GMRES accelerated SDC methods are still being pursued, including the B-stability and B-convergence.

For the original SDC methods, recently, Hagstrom and Zhou showed that when p Gauss nodes are used, after $2p$ corrections, the order of the method is $2p$ [32]. This result can be generalized to the GMRES accelerated SDC methods which solve the collocation formulation as shown by the following theorem. Notice that when GMRES is applied, at most p corrections are necessary for linear scalar problems, compared with $2p$ in [32].

Theorem 3.1 *Using p Gauss nodes, the collocation method which solves Eq. (3.1) has order $2p$.*

Proof: The proof follows closely that of Thm. 1.5 in [34]. Notice that the collocation solution to Eq. (3.1) at t_{n+1} is derived by spectral integration, which is equivalent to evaluating at t_{n+1} the degree p polynomial $P(t)$ obtained by integrating the degree $p-1$ interpolating polynomial $L^p(\vec{F},\tau)$ where $\vec{F} = [F(t_1,\varphi_1),\dots F(t_p,\varphi_p)]^T$, i.e.,

$$P(t) = \varphi_0 + \int_{t_0}^{t} L^p(\vec{F},\tau)d\tau.$$

For this polynomial $P(t)$ it is straightforward to show:

1. $P(t_0) = \varphi_0$.
2. $P'(t_i) = F(t_i,\varphi_i)$ at all Gauss nodes (by the definition of $P(t)$).
3. $P(t_i) = \varphi_i$ for $i = 1,\cdots,p$ (from the collocation formulation).

Therefore, for $t_n < t < t_{n+1}$, the polynomial $P(t)$ satisfies

$$P'(t) = F(t,P(t)) + \sigma(t),$$

where $\sigma(t) = P'(t) - F(t, P(t))$ and satisfies $\sigma(t_i) = 0$ at the Gauss nodes. The error $P(t) - \varphi(t)$ then satisfies

$$P'(t) - \varphi'(t) = F(t, P(t)) - F(t, \varphi(t)) + \sigma(t).$$

Constructing the Taylor expansion of $F(t, P(t))$ at $F(t, \varphi(t))$ yields

$$P'(t) - \varphi'(t) = \frac{\partial F}{\partial \varphi}(t, \varphi(t))\,(P(t) - \varphi(t)) + \sigma(t) + O(\|P(t) - \varphi(t)\|^2).$$

As $P(t_0) - \varphi(t_0) = 0$, the solution to this equation is given by the variation of constants formula (see [36])

$$P(t_{n+1}) - \varphi(t_{n+1}) = \int_{t_0}^{t_{n+1}} R(t_{n+1}, \tau)\left(\sigma(\tau) + O(\|P(\tau) - \varphi(\tau)\|^2)\right) d\tau,$$

where $R(t, \tau)$ is the Green's function of the corresponding homogeneous differential equation and is smooth for $\tau < t$. Applying the theorem that the local truncation error $P(t) - \varphi(t)$ is at least $O\left(\Delta t^{p+1}\right)$ (see e.g., page 29 in [34]), we can neglect the term $O(\|P(t) - \varphi(t)\|^2)$ which is at least $O\left(\Delta t^{2p+2}\right)$ and derive

$$P(t_{n+1}) - \varphi(t_{n+1}) = \int_{t_0}^{t_{n+1}} R(t_{n+1}, \tau)\sigma(\tau)d\tau + O\left(\Delta t^{2p+2}\right).$$

Since Gauss quadrature is applied to the integral and $\sigma(t_i) = 0$ at the Gauss nodes, the collocation solution $P(t)$ has the same order as the underlying quadrature formula. When Gauss-Legendre nodes are used, the order of the local truncation error is therefore $2p + 1$. The same proof can be applied to show that when Radau IIa nodes are used, the local truncation error is $O(\Delta t^{2p})$.

3.5 Numerical Experiments

In this section, we show some preliminary numerical results for both linear and nonlinear problems. Depending on the stiffness of the problem, we present results for both the explicit and implicit GMRES accelerated SDC methods.

3.5.1 The Cosine Problem

For the first numerical example, define $p(t) = \cos(t)$ and consider

$$\begin{aligned} \varphi'(t) &= p'(t) - \frac{1}{\varepsilon}(\varphi(t) - p(t)), \qquad t \in [0, t_{final}], \\ \varphi(0) &= p(0). \end{aligned}$$

The exact solution is clearly $\varphi(t) = p(t)$. Notice that when ε is small, this problem is stiff, however, the solution itself is smooth and independent of ε.

For the first example, we set $\varepsilon = 0.02$ and $\Delta t = 1$. For each time step, we use 12 Radau IIa nodes. For the time-stepping we use the explicit Euler method in Eq. (2.12). In Table 3.1, we show the numerical error after one step ($\Delta t = t_{final} = 1$) for different GMRES(k_0). For $k_0 = 0$, the method is the original SDC. Also, for each step, we fix the number of explicit Euler corrections to 12. The total number of function evaluations is therefore fixed to 12×12.

k_0	0	1	2	3	4	6	12
error	4.2e+57	4.6e-1	3.8e-3	2.1e-3	9.2e-4	1.7e-4	3.6e-13

Table 3.1: Errors versus k_0 for the Cosine Problem for the explicit GMRES accelerated SDC method.

These results are consistent with the stability analysis in Sec. 3.4. Clearly, the GMRES accelerated SDC methods give better numerical results even though the original SDC method is unstable. Also, for the restarted GMRES(k_0), keeping more data in

memory (larger k_0) reduces the error. The full orthogonalization process ($k_0 = 12$, the same as the number of unknowns) returns converged numerical results but loses a few digits in accuracy due to the fact that the forward Euler predictor is actually unstable here (see also Fig. 3.3).

Next, consider the case $\varepsilon = 10^{-6}$. As the problem is very stiff, the implicit GMRES accelerated SDC method is used. Note that for this example, the original SDC method is stable. As in the explicit examples, the results shown in Table 3.2 demonstrate that increasing k_0 reduces the error and residual (defined as $b - Ax$ when solving $Ax = b$) and that both go to machine precision with the full GMRES.

k_0	0	1	2	3	4	6	12
error	1.4e-4	3.6e-4	1.6e-4	5.5e-5	3.3e-5	1.2e-5	4.4e-16
residual	2.3e-4	2.9e-4	1.6e-4	8.1e-5	4.8e-5	1.6e-5	2.8e-16

Table 3.2: Errors and residuals versus k_0 for the Cosine Problem for the implicit GMRES accelerated SDC method.

Note that in both the explicit and implicit examples, the error first decays slowly as a function of k_0, and then suddenly decreases to close to machine precision once k_0 is the same as the number of nodes p. The convergence of the GMRES procedure depends in general on the distribution of the eigenvalues of the matrix being considered. In the present context, the eigenvalues of the matrix C defined in Eq. (3.11) are of interest, and these in turn depend on the matrix $S - \tilde{S}$. The eigenvalues of C for both the explicit and implicit cases above are shown in Fig. 3.3. The eigenvalues in the implicit case are smaller by about four orders of magnitude than in the explicit case, but in neither case are the eigenvalues clustered about a single point.

Order Reduction

In [51], when the original SDC method is applied to stiff problems and the number of corrections for each step is fixed, the effective order of accuracy is reduced for values

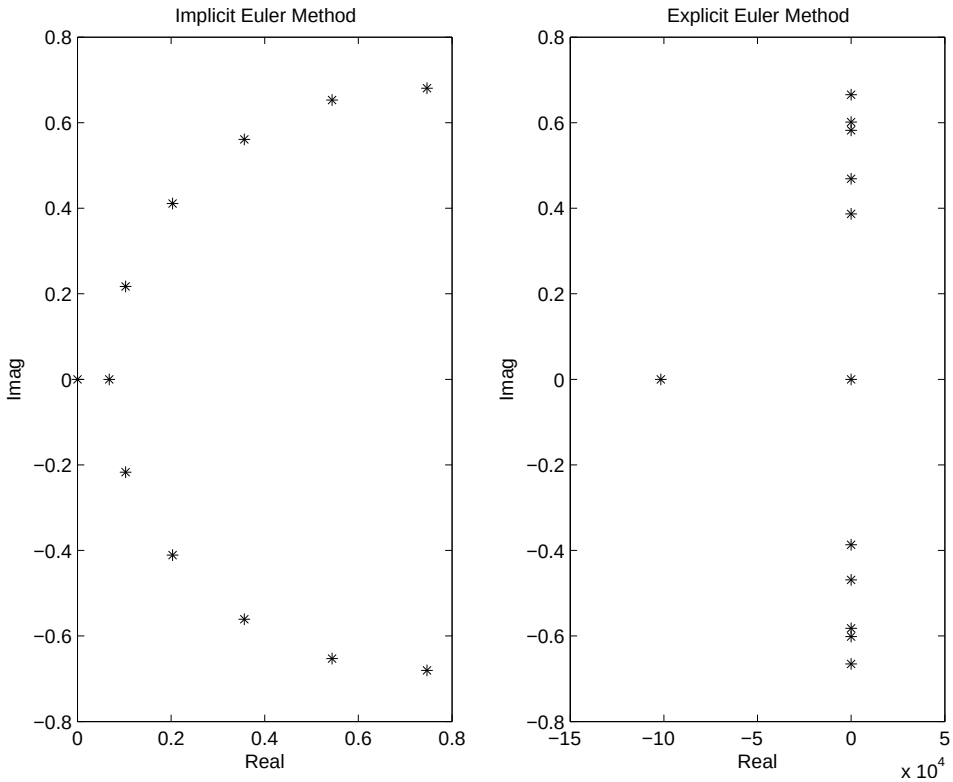

Figure 3.3: Eigenvalue distribution of C for both the implicit and explicit method. Note that the axis in the right panel are scaled by 10^4.

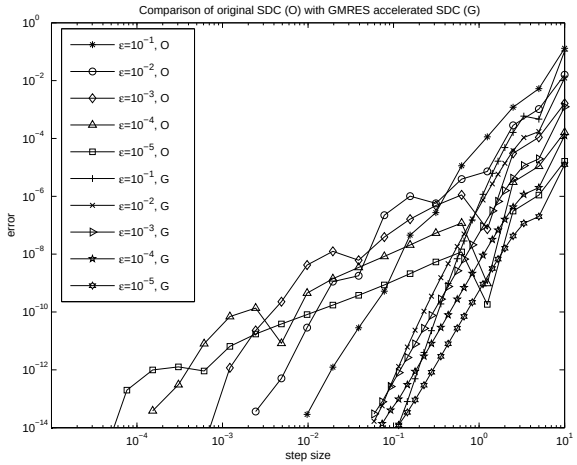

Figure 3.4: Order reduction: the original SDC and the GMRES accelerated SDC.

of the time step size in a certain range. This type of order reduction is also present in many popular types of Runge-Kutta methods [15, 19, 67]. The implication of order reduction is that, although the methods are stable for larger time steps, one must use a very small time step, or increase the number of SDC corrections for the method to converge with full order. However, with the GMRES accelerated SDC methods, when full orthogonalization is used, order reduction is no longer observed. In Fig. 3.4, convergence results are presented for both the original SDC and the new implicit GMRES accelerated SDC methods for different ε and step size selections. In the calculation, 10 Radau IIa nodes are used, and 10 iterations are performed. The order reduction phenomenon can be easily observed when ε is small (curves on the left). The plots also indicate the benefit in terms of computational cost the GMRES acceleration provides. For example, when $\varepsilon = 10^{-5}$, in order to have 13 digits of accuracy, the original SDC requires a step size of approximately 10^{-5}. For the GMRES accelerated SDC method with full orthogonalization, the necessary step size is approximately 0.1, or 4 orders of magnitude greater.

3.5.2 The Linear Multiple Mode Problem

As mentioned above, the convergence of a GMRES accelerated SDC method will depend on the eigenvalues of the matrix C in Eq. (3.11), which depend also on the eigenvalues of the linear operator L. Hence in our second set of tests, we study an ODE system similar to the cosine problem in which we can specify the distribution of the eigenvalues. When GMRES is applied to the original SDC, it is usually expensive to use the full orthogonalization process since it would require $k_0 = pN$ iterations for a system of N ODEs using p nodes. This increases both the memory required and the amount of work performed. Therefore a natural question is, given some information on the distribution of the eigenvalues, can we determine the "optimal" number k_0? The following numerical experiments are intended to provide some basic guidelines.

The problem studied in this example is

$$\begin{aligned} \bar{y}'(t) &= \bar{p}'(t) - B(\bar{y}(t) - \bar{p}(t)) \\ \bar{y}(0) &= \bar{p}(0) \end{aligned}$$

where $\bar{y}(t)$ and $\bar{p}(t)$ are vectors of dimension N. The exact solution is again $\bar{y}(t) = \bar{p}(t)$. The matrix B is constructed by

$$B = U^T \Lambda U,$$

where U is a randomly generated orthogonal matrix, and Λ is a diagonal matrix whose diagonal entries $\{\lambda_i\}_{i=1}^N$ are all positive. For $\bar{p}(t)$, we choose the ith component as $\cos(t + \alpha_i)$ with phase parameter $\alpha_i = 2\pi i/N$.

In our first experiment, we set the dimension of the system to 10, and use 10 Radau IIa nodes in the simulation. We use $\Delta t = 0.1$ and study one time step (i.e. $t_{final} = \Delta t$). In the left of Fig. 3.5, we set $\lambda_1 = 10^7$, and all other λ_i to 1. It can be seen that when $k_0 \approx 12$, the residual converges to machine precision in about 25 iterations. Notice that

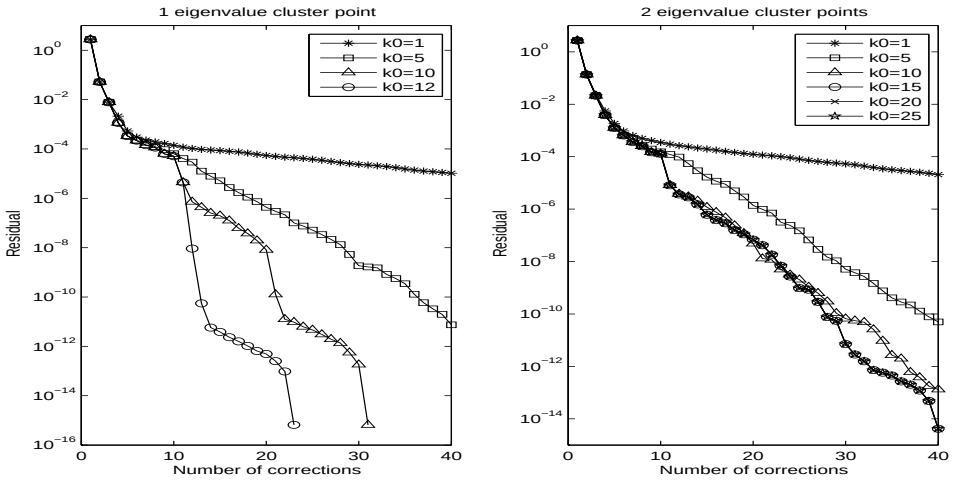

Figure 3.5: Comparison of errors for different k_0

10 Radau IIa nodes resolve the solution to 14 digits, therefore the residual is equivalent to the error (up to a constant factor). In the right panel, results are shown for the case when two eigenvalues are 10^7, two are 10^4, and the rest are 1. In this case, more iterations are required to reduce the residual to machine precision, and it requires a slightly higher k_0 of approximately 15 to yield the best convergence results (i.e. results for $k_0 = 15$, are almost identical to those using larger k_0). However, these values of k_0 are much smaller compared with the full GMRES which requires $k_0 = 100$. Since the original SDC method would require ten iterations of the correction equation, in this case, there is a factor of 4 increase in the number of iterations for GMRES accelerated SDC method, however, this results in a reduction in the error of approximately 10 orders of magnitude.

In our second experiment, we consider the case where $N = 100$ and the $\log_{10}$ of the eigenvalues are uniformly distributed on $[0, 7]$. For 10 Radau IIa nodes, numerical results for different k_0 are shown in Fig. 3.6. In this example, convergence profiles for $k_0 > 10$ are very similar. Notice that the full GMRES requires $k_0 = 1000$, hence only a small fraction of the full method is required for machine precision. At present, the optimal strategy for picking k_0 for a given problem is not completely understood, although these experiments suggest that a successful strategy must depend on the time step, the size

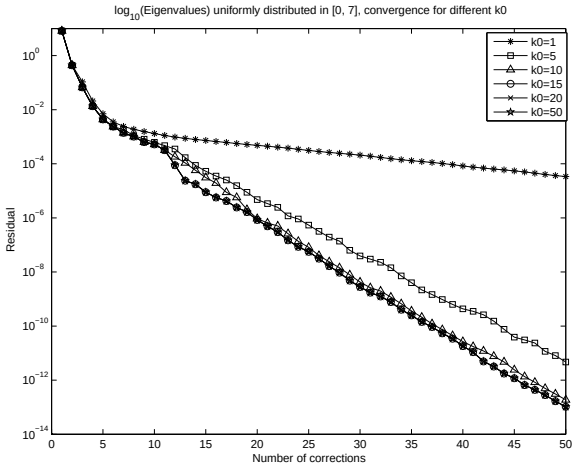

Figure 3.6: Comparison of errors for different k_0

of the system, the distribution of the eigenvalues, and of course any memory restrictions based on the problem size. We are currently investigating strategies for choosing k_0 in the broader context of step size selection.

3.5.3 The Van der Pol Oscillator

In our third example, we consider the nonlinear ODE initial value problem which describes the behavior of vacuum tube circuits. It was proposed by B. Van der Pol in the 1920's, and is often referred to as the Van der Pol oscillator. As a first order ODE system, the problem takes the form

$$\begin{cases} y_1'(t) &= y_2(t), \\ y_2'(t) &= \left(1 - y_1^2(t) y_2(t) - y_1(t)\right)/\varepsilon \end{cases} \tag{3.22}$$

where the initial values are given by $[y(0), y'(0)] = [2, -0.6666654321121172]$. This is a stiff system when ε is small. For this nonlinear problem, we use the "linear implicit" GMRES accelerated SDC methods discussed in Sec. 3.3, and choose the following strategy in the implementation: GMRES(k_0) is applied to the linearized system until the residual $b - Ax$ is reduced by a factor of tol_G; once this is done, we update the Jacobian matrix

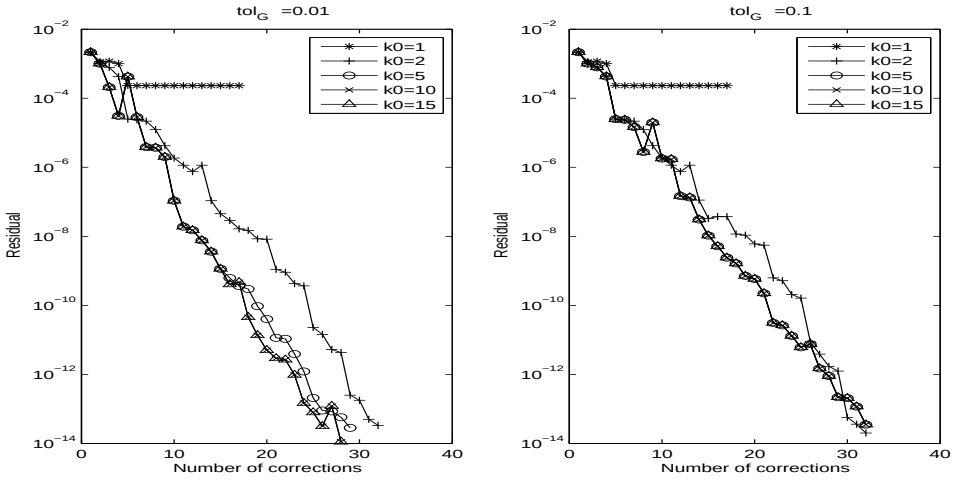

Figure 3.7: Comparison of different k_0 for the Van der Pol problem, explicit method

and restart GMRES(k_0).

As our first experiment, we set $\varepsilon = 10^{-3}$, and use $\Delta t = 0.001$. We apply the **explicit** GMRES accelerated SDC method, and in Fig. 3.7, we show how the residual decays (as the analytical solution is not readily available) for different k_0 when $tol_G = 0.01$ (left) and $tol_G = 0.1$ (right). Here, the residual is defined as the error $\|b - Ax\|$ for the linearized system in Eq. (3.18). It can be seen that the GMRES accelerated SDC method converges quickly to the solution of the collocation formulation. This is consistent with the stability analysis in Sec. 3.4 and the linear cosine test problems in Sec. 3.5.1. Notice that for this problem, the original SDC method is divergent (not shown on plot), and GMRES(1) converges very slowly.

Because of the nonlinearity of the problem, the convergence behavior of the GMRES accelerated SDC method also depends on tol_G. The two panels in Fig. 3.7 compare convergence for $tol_G = 0.01$ and $tol_G = 0.1$. In the left panel, it appears that using $k_0 = 10$ is sufficient for achieving the best convergence results since the convergence for $k_0 = 15$ is nearly identical. In the right panel, convergence for $k_0 = 5$ is the same as for $k_0 = 10$ and $k_0 = 15$, although the overall number of iterations required to achieve a specified error tolerance increases slightly compared to $tol_G = 0.01$. Determining the

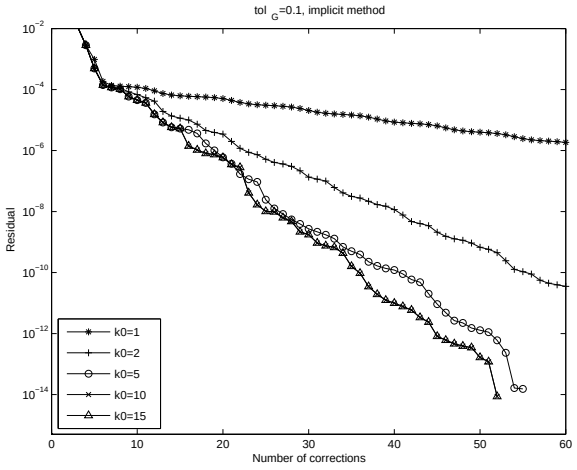

Figure 3.8: Comparison of different k_0 for the very stiff Van der Pol problem with implicit method.

"optimal" choice of tol_G or an adaptive strategy for choosing tol_G is an open issue.

Next we apply the implicit GMRES accelerated SDC method to the very stiff case with $\varepsilon = 10^{-8}$ and $\Delta t = 0.5$. The results are shown in Fig. 3.8 for different choices of k_0 and $tol_G = 0.1$. In all cases, the GMRES accelerated SDC methods converge more rapidly to the collocation solution than the original SDC method.

It is possible to apply a different numerical method for the time marching of the correction equation. In the above examples, either the explicit or implicit Euler method is used for the correction equation. It is reasonable to expect that the use of a higher-order numerical method for the time marching of the correction equation would result in a method which requires fewer iterations of the correction equation to converge to a specified tolerance. In the linear case, this is equivalent to choosing a different preconditioner for the Neumann series expansion. We investigate this idea by repeating the above numerical example using the trapezoid rule. The results are compared with those from the implicit Euler method in Fig. 3.9 for $k_0 = 1$, 2 and 10. From this figure, we can see that using larger k_0 again improves the numerical convergence in both cases. However, when $k_0 = 10$, the trapezoid rule results are not significantly better than those computed with the first-order method. Hence, at least for this limited experiment, using a higher-order

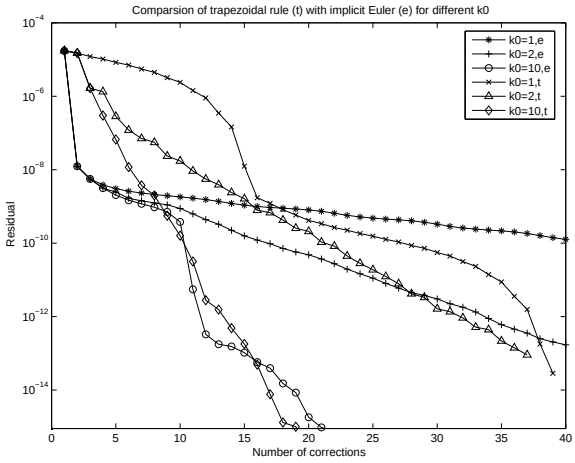

Figure 3.9: Comparison using trapezoidal rule versus the implicit Euler method.

marching method does not seem to have a significant effect on the convergence when the GMRES acceleration procedure is used. (See also [38].)

3.5.4 The Nonlinear Multi-mode Problem

In this example, we study a nonlinear generalization of the multi-mode example in Sec. 3.5.2. The problem is given by a system of N nonlinear equations

$$\begin{cases} y_i'(t) &= p_i'(t) - \lambda_i y_{i+1}(t)(y_i(t) - p_i(t)) \qquad 1 < i < N-1, \\ y_N'(t) &= p_N'(t) - \lambda_i(y_i(t) - p_i(t)) \qquad i = N. \end{cases}$$

The analytical solution is again $\bar{y}(t) = \bar{p}(t)$ where the ith component of $\bar{p}(t)$ is given by $p_i(t) = 2 + \cos(t + \alpha_i)$ with phase parameter $\alpha_i = 2\pi i/N$. In our first experiment, we set $N = 7$ and the eigenvalues are chosen as $[10^8, 10^8, 1, 1, 1, 1, 1]$. In the left of Fig. 3.10, as in the first linear multi-mode test, we show how the residual decays in one time step for different k_0 where $tol_G = 10^{-1}$. In the simulation, we use the implicit GMRES accelerated SDC method with $p = 10$ Radau IIa nodes and Δt is chosen to be 0.3. It can be seen that the linear implicit GMRES accelerated SDC greatly improves the convergence of the SDC procedure. Also, when $k_0 > p$, the convergence of the method is very satisfactory. In our

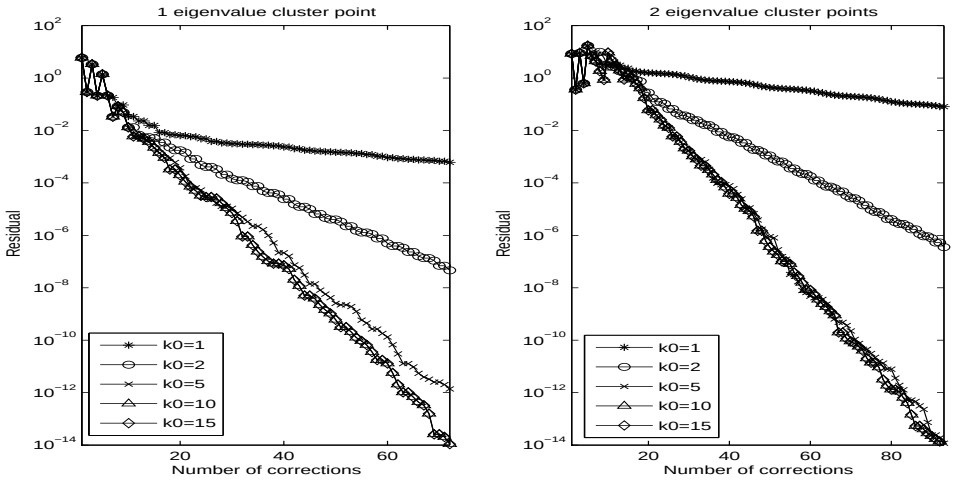

Figure 3.10: Nonlinear multi-mode convergence

second experiment, we choose the eigenvalues as $[10^8, 10^8, 10^5, 10^5, 1, 1, 1]$ so that there are two eigenvalue cluster points away from 1 as in the second linear multi-mode example. The right panel of Fig. 3.10 shows that somewhat more corrections are required for convergence in this case; however, the minimum k_0 required for reasonable performance does not increase over the case with only one cluster. In both cases, $k_0 = 5$ now gives convergence behavior very similar to using larger k_0.

3.5.5 The Ring Modulator Problem

In our last example, we consider a stiff nonlinear ODE system of 15 equations. The problem originates from electrical circuit analysis. Specifically, it describes the behavior of the ring modulator [1], and takes the form

$$\frac{d\vec{y}}{dt} = \vec{f}(t, \vec{y}), \quad \vec{y} = \vec{y}_0,$$

with

$$\vec{y} \in \mathbb{R}^{15}, \quad 0 \leq t \leq 10^{-5}.$$

In this equation, the function $\vec{f}$ is defined by

$$\vec{f}(t,\vec{y}) = \begin{pmatrix} C^{-1}(y_8 - 0.5y_{10} + 0.5y_{11} + y_{14} - R^{-1}y_1) \\ C^{-1}(y_9 - 0.5y_{12} + 0.5y_{13} + y_{15} - R^{-1}y_2) \\ C_s^{-1}(y_{10} - q(U_{D1}) + q(U_{D4})) \\ -C_s^{-1}(y_{11} - q(U_{D2}) + q(U_{D3})) \\ C_s^{-1}(y_{12} + q(U_{D1}) - q(U_{D3})) \\ -C_s^{-1}(y_{13} + q(U_{D2}) - q(U_{D4})) \\ C_p^{-1}(-R_p^{-1}y_7 + q(U_{D1}) + q(U_{D2}) - q(U_{D3}) - q(U_{D4})) \\ -L_h^{-1}y_1 \\ -L_h^{-1}y_2 \\ L_{s2}^{-1}(0.5y_1 - y_3 - R_{g2}y_{10}) \\ -L_{s3}^{-1}(0.5y_1 - y_4 + R_{g3}y_{11}) \\ L_{s2}^{-1}(0.5y_2 - y_5 - R_{g2}y_{12}) \\ -L_{s3}^{-1}(0.5y_2 - y_6 + R_{g3}y_{13}) \\ L_{s1}^{-1}(-y_1 + U_{in1}(t) - (R_i + R_{g4})y_{14}) \\ L_{s1}^{-1}(-y_2 - (R_c + R_{g1})y_{15}) \end{pmatrix}.$$

The auxiliary functions $U_{D1}, U_{D2}, U_{D3}, U_{D4}, q, U_{in1}$ and U_{in2} are given by

$$\begin{aligned} U_{D1} &= y_3 - y_5 - y_7 - U_{in2}(t), \\ U_{D2} &= -y_4 + y_6 - y_7 - U_{in2}(t), \\ U_{D3} &= y_4 + y_5 - y_7 + U_{in2}(t), \\ U_{D4} &= -y_3 - y_6 + y_7 + U_{in2}(t), \\ q(U) &= \gamma(e^{\delta U} - 1), \\ U_{in1}(t) &= 0.5\sin(2000\pi t), \\ U_{in2}(t) &= 2\sin(2000\pi t). \end{aligned}$$

The values of the parameters are

C	$=$	$1.6 \cdot 10^{-8}$	R	$=$	25000
C_s	$=$	$2 \cdot 10^{-12}$	R_i	$=$	50
C_p	$=$	10^{-8}	R_p	$=$	50
L_h	$=$	4.45	L_c	$=$	600
L_{s1}	$=$	0.002	R_{g1}	$=$	36.3
L_{s2}	$=$	$5 \cdot 10^{-4}$	R_{g2}	$=$	17.3
L_{s3}	$=$	$5 \cdot 10^{-4}$	R_{g3}	$=$	17.3
γ	$=$	$40.67286402 \cdot 10^{-9}$	δ	$=$	17.7493332

and the initial value $\vec{y}_0$ is given by

$$\vec{y}_0 = \vec{0}.$$

In the simulation, we use the implicit GMRES accelerated SDC method with $p = 7$ Radau IIa nodes. We set $tol_G = 0.1$, $k_0 = p + 1$, and $t_{final} = 10^{-5}$. Our **uniform step** GMRES accelerated SDC method is then compared with available **adaptive** ODE packages described in [1] and the results are shown in table 3.3. In the table, the parameters *rtol*, *atol* and h_0 for each method are chosen experimentally to produce a numerical solution with at least 9 significant digits, which has the fewest possible number of function evaluations.

	G-SDC	DASSL	GAMD	MEBDFI	PSIDE	RADAU	VODE
rtol	1e-8	1e-12	1e-10	1e-9	1e-10	1e-9	1e-11
atol	*	1e-12	1e-10	1e-11	1e-11	1e-10	1e-14
h_0	2.5e-6	*	1e-10	1e-10	*	1e-10	*
rerr	3.0e-9	1.1e-9	3.1e-9	2.9e-9	2.1e-9	2.1e-9	1.3e-9
F	1134	2104	4057	2284	3417	2172	2961
steps	4	1591	76	669	154	47	2277

Table 3.3: A comparison, *- not needed, *rtol*- relative tolerance, *atol*- absolute tolerance, h_0- initial step-size, *rerr*- maximum relative error, *F*- number of function evaluations, *steps*- number of steps taken.

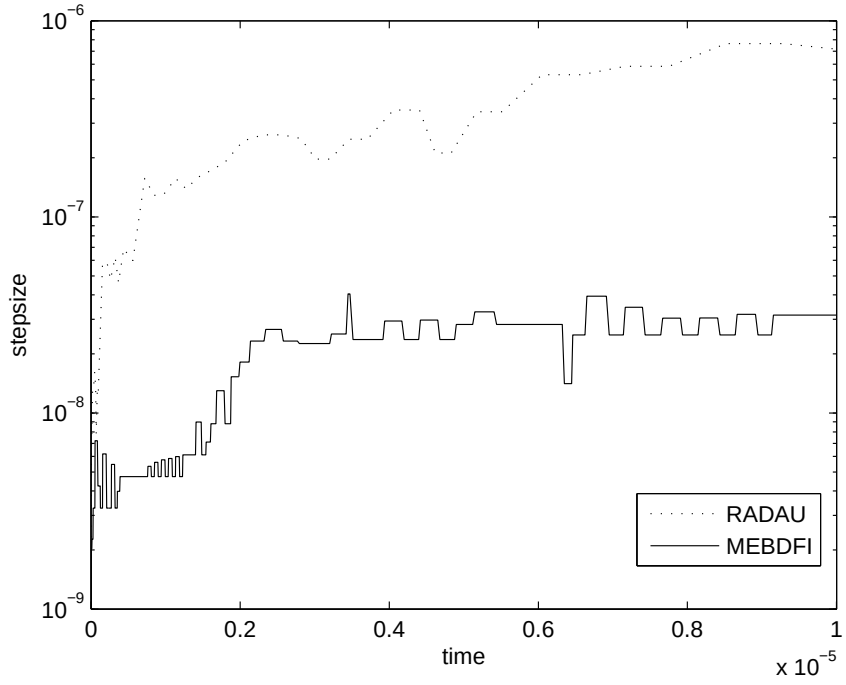

Figure 3.11: Step-sizes selected by RADAU and MEBDFI.

Our results suggest that the new GMRES accelerated SDC method is a very competitive alternative to existing ODE solvers. However, in order to perform more extensive (and convincing) tests, an automatic step-size selection strategy is required for the GMRES accelerated SDC method. In Fig. 3.11, we show the step-sizes used by the adaptive solvers MEBDFI and RADAU. Currently, we are studying strategies for step-size selections along with strategies for computing better initial provisional solutions, for adaptively choosing the parameters k_0 and tol_G, and for adaptively varying the order of the SDC method.

Finally for this section, we want to mention that we have also studied several other problems from the Test Set for IVP solvers [1]. In all cases, the convergence of the original SDC methods is greatly improved by the GMRES accelerated SDC procedure.

Chapter 4

Krylov Deferred Correction Methods for Differential Algebraic Equations

Ordinary differential equations can be considered as a special class of differential algebraic equations (ODEs are index 0 DAEs). In this chapter[1], we extend results for ODE problems from previous chapter to efficient solutions of general differential algebraic equation initial value problems. We refer to the new methods as Krylov deferred correction (KDC) methods.

This chapter is organized as follows. In Sec. 4.1, we introduce the differential algebraic equations. In particular, we discuss the index of a differential algebraic equation and give several examples. In Sec. 4.2, we construct the Krylov deferred correction methods for DAE problems. In Sec. 4.3, a discussion of issues related to the index of an DAE system is presented, including the convergence analysis of the methods. In Sec. 4.4, we present several preliminary numerical results to show the accuracy and efficiency of the new KDC methods for DAE problems.

[1]Parts of this chapter are reprinted from Journal of Computational Physics, In Press, Jingfang Huang, Jun Jia and Michael Minion, "Arbitrary order Krylov deferred correction methods for differential algebraic equations", Copyright 2006, with permission from Elsevier

4.1 Differential Algebraic Equations

In this chapter, we generalize the GMRES accelerated SDC methods to the efficient solution of differential algebraic equation (DAE) systems of the form

$$F(y(t), y'(t), t) = 0. \tag{4.1}$$

DAEs arise naturally in many applications, for example, the discretization of partial differential equations (PDEs) or model reduction and singular perturbations [13]. Compared with ordinary differential equations (ODEs), the numerical solution of DAEs is, in general, a more challenging subject since the algebraic part of the DAE can often be expressed as the infinite stiffness limit of a singular perturbation problem, and such stiffness typically poses difficulty to traditional numerical ODE methods.

An important concept for DAE systems is the index of a differential algebraic equation. There exist several equivalent definitions for this concept, in this thesis, we introduce the definition in [3] defined as follows:

Definition 4.1 *For general DAE systems (4.1), the index along a solution $y(t)$ is the minimum number of differentiations of the system which would be required to solve for y' uniquely in terms of y and t (i.e., to define an ODE for y). Thus, the index is defined in terms of the over-determined system*

$$\begin{aligned} F(y, y', t) &= 0 && (4.2)\\ \frac{d}{dt}F(y, y', y'', t) &= 0 && (4.3)\\ \cdots &= 0 && (4.4)\\ \frac{d^p}{dt^p}F(y, y', \cdots, y^{(p+1)}, t) &= 0 && (4.5) \end{aligned}$$

to be the smallest integer p so that y' in Eq. (4.5) can be solved for in terms of y and t.

We will refer to an ODE system as an index 0 DAE system. In the following we present several examples of more general DAE systems.

(Systems of Index 1) The simplest nontrivial DAE system is of the form

$$y' = f(y, z) \tag{4.6}$$

$$0 = g(y, z) \tag{4.7}$$

where f and g are sufficiently differentiable functions and g_z has a bounded inverse. Note that in this case only one differentiation of g is required to yield z', and hence the system is referred to as an index 1 DAE system.

(Systems of Index 2) The system

$$y' = f(y, z) \tag{4.8}$$

$$0 = g(y) \tag{4.9}$$

is an index 2 DAE system under the assumption that $g_y f_z$ has a bounded inverse. Note that in this case, two differentiations are required in order to derive z'.

As we will show in the following sections, the index plays an important role in finding appropriate numerical schemes for a specific DAE system.

4.2 Krylov Deferred Corrections for DAEs

Existing DAE solvers include the backward differentiation formulas (BDF) based package DASSL developed by Petzold et al. which is applicable to DAE problems of index 0 and 1 [13, 62]; and the Runge-Kutta based RADAU by Hairer et al. which can be applied to DAE problems of index up to 3 [35, 37]. Detailed discussions of these available solvers as well as a test set can be found in [1], and the readers are referred to the references therein.

In this section, we generalize the GMRES accelerated SDC methods for ODE problems to general DAE systems, and construct arbitrary order and stable Krylov deferred correction (KDC) methods. Specifically, the Picard integral formulation and collocation discretization, error equation and deferred correction method, and the use of Newton-Krylov acceleration for DAEs are discussed in order.

4.2.1 Picard Equation and Collocation Formulation for DAEs

In the original SDC method for ODEs, the Picard integral equation (2.3) is used to form the similar integral equation (2.4) for the correction. This formulation has the benefit that it avoids the ill-conditioned differentiation operator. However, it is not immediately clear how to generalize this technique to DAEs since a Picard equation form of the DAE is not readily available.

Toward this end, instead of solving for $y(t)$ in Eq. (4.1) directly, our new formulation uses $y'(t)$ as the unknown variable, which will be denoted by $Y(t)$ in the following discussions. Expressing $y(t)$ as the integral of $Y(t)$, the DAE system $F(y(t), y'(t), t) = 0$ becomes

$$F(y_0 + \int_0^t Y(\tau)d\tau, Y(t), t) = 0.$$

As in Section 3.1, this Picard type equation can be directly discretized using the spectral integration matrix S to yield

$$\vec{F}(\vec{y_0} + \Delta t S \otimes \mathbf{Y}, \mathbf{Y}, \mathbf{t}) = \mathbf{0}, \tag{4.10}$$

where $\mathbf{Y} = [\mathbf{Y_1}, \mathbf{Y_2}, \cdots, \mathbf{Y_p}]^\mathbf{T}$ is the desired solution which approximates $Y(t) = y'(t)$ at the Gaussian nodes. The solution $\mathbf{y} = [y_1, y_2, \cdots, y_p]^T$ is then recovered using the spectral integration matrix

$$\mathbf{y} = \vec{y_0} + \Delta t S \otimes \mathbf{Y}.$$

Eq. (4.10) is the analog of the collocation formulation for ODEs given in Eq. (3.1), and in the following discussions we write Eq. (4.10) symbolically as $\mathbf{H}(\mathbf{Y}) = \mathbf{0}$.

As for the initial values, note that when Gaussian or Radau IIa nodes are used, only $y(0) = y_0$ is required in the collocation formulation. However, when Radau Ia or Lobatto nodes are applied, $Y(0)$ is also required when calculating $\Delta t S \otimes \mathbf{Y}$, which can be derived by solving the equation $F(y(0), Y(0), 0) = 0$.

4.2.2 Error Equation and Modified SDC

Following the procedure in SDC methods for ODEs, given an approximation or provisional solution $\tilde{\mathbf{Y}} = [\tilde{Y}_1, \tilde{Y}_2, \cdots, \tilde{Y}_p]^T$ to the DAE system at the Gaussian nodes $\mathbf{t}$, one can define an equation for the error $\delta(t) = Y(t) - \tilde{Y}(t)$ by

$$F\left(y_0 + \int_0^t \left(\tilde{Y}(\tau) + \delta(\tau)\right) d\tau, \tilde{Y}(t) + \delta(t), t\right) = 0 \tag{4.11}$$

where $\tilde{Y}(t)$ is the polynomial interpolation of $\tilde{\mathbf{Y}}$.

As in the original SDC, we wish to use a low-order method to approximate the error equation (4.11) and improve the provisional solution $\tilde{Y}(t)$. Note that Eq. (4.11) gives the identity

$$F\left(y_0 + \int_0^{t_{m+1}} \tilde{Y}(\tau) d\tau + \left(\int_0^{t_m} + \int_{t_m}^{t_{m+1}}\right) \delta(\tau) d\tau, \tilde{Y}(t_{m+1}) + \delta(t_{m+1}), t_{m+1}\right) = 0. \tag{4.12}$$

A simple time-marching discretization of this equation similar to the explicit (forward) Euler method for ODEs gives a low-order solution $\tilde{\boldsymbol{\delta}} = [\tilde{\delta}_1, \tilde{\delta}_2, \cdots, \tilde{\delta}_p]^T$ by solving

$$F\left(y_0 + [\Delta t S \otimes \tilde{\mathbf{Y}}]_{m+1} + \sum_{l=1}^{m+1} \Delta t_l \tilde{\delta}_{l-1}, \tilde{Y}_{m+1} + \tilde{\delta}_{m+1}, t_{m+1}\right) = 0 \tag{4.13}$$

where $\Delta t_{l+1} = t_{l+1} - t_l$, t_0 and δ_0 are set to 0. Note that this updating formula is in

general implicit since no explicit formula for $\tilde{\delta}_{m+1}$ exists. Similarly, a time-marching scheme based on backward Euler method analogous to Eq. (2.13) is given by

$$F\left(y_0 + [\Delta t S \otimes \tilde{\mathbf{Y}}]_{m+1} + \sum_{l=1}^{m+1} \Delta t_l \tilde{\delta}_l, \tilde{Y}_{m+1} + \tilde{\delta}_{m+1}, t_{m+1}\right) = 0. \tag{4.14}$$

These two methods differ only in the way how the time integral of $\delta(t)$ is approximated. Eq (4.13) is equivalent to the rectangle rule using the left endpoint while Eq (4.14) is the rectangle rule using the right endpoint. A discussion of the advantages of one over the other is presented in the next section.

The two time-stepping methods can be written in matrix form as

$$\vec{F}(\vec{y_0} + \Delta t S \otimes \tilde{\mathbf{Y}} + \Delta t \tilde{S} \otimes \tilde{\boldsymbol{\delta}}, \tilde{\mathbf{Y}} + \tilde{\boldsymbol{\delta}}, \mathbf{t}) = \mathbf{0}, \tag{4.15}$$

where $\Delta t \tilde{S}$ is the lower triangular representation of the rectangle rule approximation of the spectral integration operator S as discussed in Section 3.2. As in the ODE case, the order of accuracy of $\tilde{\mathbf{Y}}$ is increased by iteratively using Eq. (4.15) to approximate the error, and if $\tilde{\mathbf{Y}}$ converges, it converges to the solution of the collocation equation (4.10). Therefore the accuracy of the current methods can be broken down into two separate convergence issues: the convergence of the deferred correction iterations to the collocation solution (as $\tilde{\boldsymbol{\delta}} \to \mathbf{0}$), and the convergence of the solution of the collocation equation to the exact solution (i.e. as $\Delta t \to 0$ or $p \to \infty$). The focus here concerns the acceleration of the first iteration, and we assume the problem is resolved to desired error tolerance by the collocation formulation. We defer further order and accuracy discussions of the collocation methods to Section 4.3.1.

4.2.3 Acceleration using Newton-Krylov Methods

In previous chapter, it is observed that the original SDC method for ODE initial value problems can be considered as a Neumann series expansion for solving the preconditioned system (3.14), where the preconditioner is the spectral deferred correction process. Also in that chapter, a linearly implicit method is applied to general nonlinear ODE problems. In this section, we generalize this idea to DAEs, and explain how Newton-Krylov methods can be directly applied to the preconditioned system instead of using the linearly implicit formulation.

From the discussion in previous section, a low-order method can be considered as a tool for deriving the approximate error $\tilde{\boldsymbol{\delta}}$ as a function of the given provisional solution $\tilde{\mathbf{Y}}$ from the "implicit" function in (4.15). In the following, we use

$$\tilde{\boldsymbol{\delta}} = \tilde{\mathbf{H}}(\tilde{\mathbf{Y}})$$

to represent the explicit form of this implicit function. As a reminder, evaluation of $\tilde{\mathbf{H}}$ is nothing more than one iteration of the deferred correction procedure. Notice that when $\tilde{\boldsymbol{\delta}} = \mathbf{0}$, the solution to Eq. (4.15) is identical to Eq. (4.10). Therefore, solving the collocation formulation $\mathbf{H}(\mathbf{Y}) = \mathbf{0}$ is equivalent to finding the zero of the implicit equation

$$\tilde{\mathbf{H}}(\mathbf{Y}) = \mathbf{0}.$$

Because the low-order method solves an approximation of the collocation formulation, it is not surprising that the explicit function $\tilde{\mathbf{H}}(\mathbf{Y}) = \mathbf{0}$ is better conditioned compared with the original collocation formulation in (4.10) as shown by the following analysis.

Applying the implicit function theorem, the Jacobian matrix $J_{\tilde{\mathbf{H}}}$ of $\tilde{\mathbf{H}}$ is given by

$$\begin{aligned} J_{\tilde{\mathbf{H}}} = \frac{\partial \tilde{\boldsymbol{\delta}}}{\partial \mathbf{Y}} &= -\left(\frac{\partial \vec{F}}{\partial \mathbf{Y}} + \frac{\partial \vec{F}}{\partial \mathbf{y}}\Delta t \tilde{S}\right)^{-1}\left(\frac{\partial \vec{F}}{\partial \mathbf{Y}} + \frac{\partial \vec{F}}{\partial \mathbf{y}}\Delta t S\right) \\ &= -I + \left(\frac{\partial \vec{F}}{\partial \mathbf{Y}} + \frac{\partial \vec{F}}{\partial \mathbf{y}}\Delta t \tilde{S}\right)^{-1}\left(\frac{\partial \vec{F}}{\partial \mathbf{y}}\Delta t(\tilde{S} - S)\right). \end{aligned} \tag{4.16}$$

When $\frac{\partial \vec{F}}{\partial \mathbf{Y}}$ is non-singular (e.g. $\frac{\partial \vec{F}}{\partial \mathbf{Y}} = I$ for ODE systems), since $\tilde{S}$ is an approximation of S, when Δt is small, $J_{\tilde{\mathbf{H}}}$ is close to $-I$. This was the first requirement for the efficient application of Newton-Krylov methods discussed in Section 2.2.2. For comparison, the Jacobian matrix of $\mathbf{H} = \mathbf{0}$ is given by

$$J_{\mathbf{H}} = \frac{\partial \mathbf{H}}{\partial \mathbf{Y}} = \left(\frac{\partial \vec{F}}{\partial \mathbf{Y}} + \frac{\partial \vec{F}}{\partial \mathbf{y}}\Delta t S\right).$$

For higher index DAE problems in which $\frac{\partial \vec{F}}{\partial \mathbf{Y}}$ is singular, the KDC techniques can be applied to some of the unknowns as will be discussed in next section. Using the implicit function theorem, it can be shown for this case as well that the resulting Jacobian matrix is again closer to the identity matrix compared with $J_{\mathbf{H}}$. Also, when any eigenvalue λ of the matrix

$$C = J_{\tilde{\mathbf{H}}} + I = \left(\frac{\partial \vec{F}}{\partial \mathbf{Y}} + \frac{\partial \vec{F}}{\partial \mathbf{y}}\Delta t \tilde{S}\right)^{-1}\left(\frac{\partial \vec{F}}{\partial \mathbf{y}}\Delta t(\tilde{S} - S)\right) \tag{4.17}$$

satisfies $\|\lambda\| \geq 1$ (this may happen to higher index DAE systems independent of the choice of Δt), the original SDC methods (consider the Neumann series for linear problems) become divergent, on the other hand, the Newton-Krylov methods converge efficiently as long as the number of such eigenvalues is small.

Finally, we recall that the second requirement for the efficient application of Newton-Krylov methods discussed in Section 2.2.2 is an efficient procedure for computing the function $\tilde{\mathbf{H}}$ used by the forward difference approximation. As noted earlier, this is simply a deferred correction iteration described succinctly in Eq. (4.15).

4.3 Convergence, Index, and Implementations

Unlike the construction of numerical techniques for ODE initial value problems, which is considered a mature subject in many respects, the efficient and accurate solutions of DAE systems are more challenging in both theory and implementation, especially for higher index systems. The purpose of this section is to present preliminary results on the analytical and numerical properties of the KDC methods, including the convergence analysis and essential techniques describing how a DAE system, depending on its index, can be discretized so that the new KDC methods are more efficient and accurate.

4.3.1 Convergence Analysis

In the KDC methods, the Newton-Krylov schemes are applied to the preconditioned collocation formulations. Therefore the convergence of the methods is determined by (a) the convergence of the collocation formulation and (b) the convergence of the Newton-Krylov methods. Each of these topics has been extensively studied previously and we summarize several results from existing literature in the following. These results, when coupled together, show the convergence of the KDC methods.

(Convergence and Orders of Collocation Formulations) In [35, 34], it was shown that the collocation formulations are equivalent to the Runge-Kutta methods (see p.27 in [34]), and their convergence and orders are determined by the collocation points and the DAE problems to be solved, in particular, the index of the DAE system (see p.18 in [35]). As a general guidance, the collocation formulation for ODE systems (index 0 DAE systems) using p Gaussian points is order $2p$, A-stable and L-stable, symplectic and symmetric, and hence the "optimal" choice. For higher index DAE systems, order reductions can be observed and in general, Radau IIa nodes outperform other choices of collocation points. Interested readers are referred to [35] for further details on the convergence (as well as divergence) of collocation formulations for different DAE problems.

(Convergence of Newton-Krylov Methods) It is well-known that under rather standard assumptions, the original Newton's method converges quadratically when the initial guess is "close" to the real solution. However, in the Newton-Krylov methods (inexact Newton methods), each linear correction equation is only solved approximately and the **local** convergence order is no longer quadratic (but still convergent). It can be shown that super-linear local convergence can be obtained for specially chosen parameters in the Newton-Krylov schemes (see Theorem 6.1.2 in [45]). As for arbitrary initial approximation, continuation/homotopy methods are necessary to accomplish **global** convergence. Interested readers are referred to [46, 45] for further discussions.

Finally we want to mention that the efficiency of the KDC methods can be significantly affected by the different choice of deferred correction based preconditioning strategies including the low order semi-implicit schemes to be discussed in Section 4.3.4. In general, the effective choice of preconditioner requires deep insight into the structure of the DAE system and hence problem dependent. Interested readers are referred to [7] and the references therein for a discussion of general preconditioning techniques.

4.3.2 Differential and Algebraic Components

One implicit assumption here is that the error equation (4.13) or (4.14) for $\tilde{\boldsymbol{\delta}}$ is more efficient to solve compared with a direct solution of the collocation formulation (4.10) or the original DAE system (4.1). Although this assumption is generally true for ODE problems, due to the existence of algebraic equations, it may not be the case for at least some of the unknowns in DAE systems as explained by the following examples.

(Purely Algebraic Equation Systems) For the purpose of intuitive insight, consider first the purely algebraic system

$$F(y, t) = 0.$$

As the derivative never appears in this system (hence y is referred to as an algebraic component), introducing the error equation and spectral integration can neither improve the efficiency nor accuracy. In fact, as spectral integration couples solutions at different node points, simply using Newton's method for the algebraic system at required nodes will in this case be more efficient than using the KDC approach.

(Index 1 Problems) Next consider the index 1 DAE problem

$$\begin{cases} y' = f(y, z) \\ 0 = g(y, z) \end{cases}$$

where f and g are sufficiently differentiable and the inverse of $\frac{\partial g}{\partial z}$ is bounded. In this case, as the derivative of z never appears in the system (hence z is called the algebraic component), it is more efficient to apply spectral integration only to the differential component y by making $\{Y, z\}$ the unknowns. The corresponding discretized collocation formulation becomes

$$\begin{cases} \mathbf{Y} = \tilde{\mathbf{f}}(\tilde{\mathbf{y_0}} + \mathbf{\Delta t S} \otimes \mathbf{Y}, \tilde{\mathbf{z}}), \\ \mathbf{0} = \vec{g}(\vec{y_0} + \Delta t S \otimes \mathbf{Y}, \tilde{\mathbf{z}}), \end{cases}$$

and the error equation is given by

$$\begin{cases} \tilde{\mathbf{Y}} + \tilde{\boldsymbol{\delta}} = \vec{f}(\vec{y_0} + \Delta t S \otimes \tilde{\mathbf{Y}} + \Delta t \tilde{S} \otimes \tilde{\boldsymbol{\delta}}, \vec{z}), \\ \mathbf{0} = \vec{g}(\vec{y_0} + \Delta t S \otimes \tilde{\mathbf{Y}} + \Delta t \tilde{S} \otimes \tilde{\boldsymbol{\delta}}, \vec{z}). \end{cases}$$

The Newton-Krylov procedure is then only applied to $\tilde{\boldsymbol{\delta}}$ corresponding to the Y-component, and the preconditioned system is denoted by $\tilde{\boldsymbol{\delta}} = \tilde{\mathbf{H}}(\mathbf{Y})$.

An immediate advantage of this modified formulation is that the number of unknowns is reduced in $\tilde{\mathbf{H}}(\mathbf{Y}) = \mathbf{0}$ which reduces the cost of the Newton-Krylov solver.

(Higher Index Problems) For higher index problems, we recommend that the differential components and the algebraic ones be treated differently. As a general rule, the

spectral integration technique and the Krylov subspace methods should only be applied to the differential components.

Our preliminary numerical experiments show that due to the reduced number of unknowns in the KDC methods, the modified formulation is more efficient compared with the original formulation introduced in Section 4.2 where KDC strategies are applied to both components, however the accuracy of the solutions are similar for all problems we tested. Rigorous analyses for both formulations as well as implementation details are currently being pursued.

4.3.3 Scaled Newton's Method and the Index

When KDC methods are applied to DAE systems, each marching step (from Gaussian node t_m to t_{m+1}) in one SDC sweep requires the solution of a nonlinear system. When Newton's method is used, unlike the ODE cases, special numerical treatments are required for higher index DAE systems, including the correct scaling when the Jacobian matrix is poorly scaled, e.g.,

$$J = \begin{bmatrix} O(1) & O(1) \\ O(h) & O(h) \end{bmatrix};$$

and different error control strategies as the convergence behavior for different components in the solution may not be the same. These techniques are generally problem (index) dependent, and our experience has shown that these techniques are necessary for the KDC methods to converge efficiently as well. As these techniques are already well-documented in the context of the BDF and Runge-Kutta based methods (See, e.g. Sec. 7 in [35] and Sec. 5.2 in [13]), we neglect the details here.

4.3.4 Semi-implicit KDC Schemes as Preconditioners

Comparing Eqs. (4.13) and (4.14), one may wonder if and when the explicit Euler method is useful since both forms require the solution of a nonlinear system for the unknowns. There are many cases that applying an explicit scheme to some of the equations in the DAE system may improve the efficiency of the numerical method. One such case is the well-known non-stiff ODE systems. A more relevant example is the index 1 DAE system discussed in Section 4.3.2. When an explicit method can be applied to the first equation, z then becomes the only unknown at next time step, and hence Newton's method becomes more efficient. It is also possible to use a "semi-implicit" discretization analogous to the ODE case [58] where some terms in the DAE system are treated with using Eq. (4.13) and some with (4.14). In the KDC methods framework, these techniques can be considered as different preconditioning strategies and we recommend explicit treatment of the non-stiff equations in the system whenever possible.

4.4 Preliminary Numerical Results

In this section, we show some preliminary numerical results for linear and nonlinear DAE problems with different index. The new methods are currently implemented in matlab, and Radau IIa nodes are used in spectral integration.

4.4.1 A Linear DAE System

For the first example, we consider a simple linear DAE system of index 2 (see p. 267 in [3] where α is set to 10)

$$\begin{pmatrix} y_1'(t) \\ y_2'(t) \\ 0 \end{pmatrix} = \begin{pmatrix} 10 - \frac{1}{2-t} & 0 & 10(2-t) \\ \frac{9}{2-t} & -1 & 9 \\ t+2 & t^2-4 & 0 \end{pmatrix} \begin{pmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{pmatrix} + \begin{pmatrix} \frac{3-t}{2-t}e^t \\ 2e^t \\ e^t(2-t-t^2) \end{pmatrix} \quad (4.18)$$

whose analytical solution is given by $\mathbf{y}(t) = (e^t, e^t, -e^t/(2-t))$.

We first demonstrate the convergence behavior of the KDC methods by computing the solution from $t_0 = 0$ to $t_{final} = 1$ using Radau IIa points with $p = 3$, 4, and 5. The full GMRES orthogonalization procedure is applied to the resulting preconditioned linear collocation formulations. The convergence of the error at $t = 1$ versus the time step for the KDC methods is shown in Fig. 4.1 for $y_1(t)$ (left) and $y_3(t)$ (right), respectively. The data in Fig. 4.1 confirms that the KDC method using p Radau IIa nodes is converging with approximate order $2p - 1$.

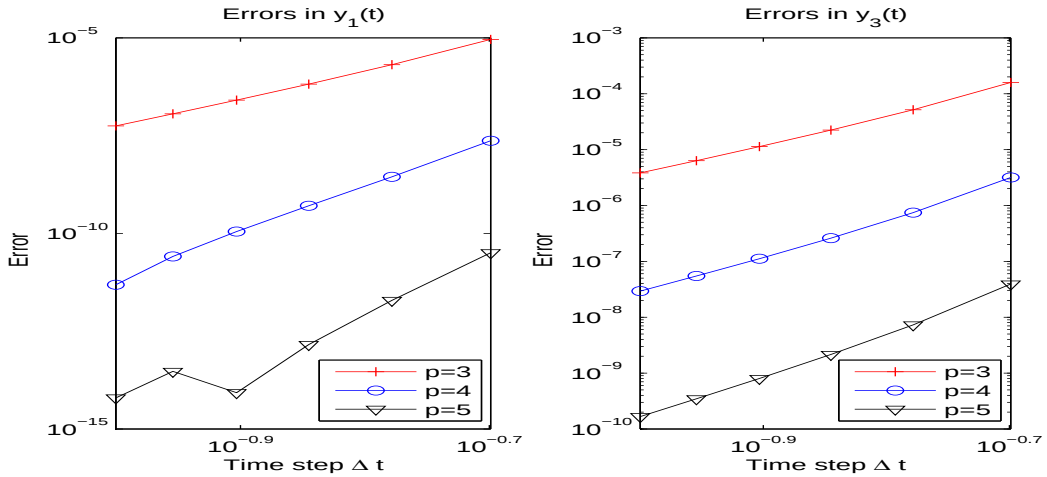

Figure 4.1: Convergence test of KDC methods with full GMRES and Radau IIa points

As a comparison with BDF based methods of orders 2, 3, and 4 (see [3], p268, Fig. 10.2) where a step-size smaller than 10^{-3} is required for 10 digits of accuracy in y_1, the new KDC methods using $p = 5$ only requires a step-size of approximately $10^{-0.9}$ for 14 digits accuracy, with 440 function evaluations.

When the GMRES method is applied to the $N \times N$ linear system $Ax = b$, the memory required increases linearly with the iteration number k, and the number of multiplications scales like $\frac{1}{2}k^2N$. Hence, for large k, the restarted version GMRES(k_0) should be applied. We next study the effect of using GMRES(k_0) on the efficiency of KDC methods.

For the linear system above, we use 16 Radau IIa nodes and set $\Delta t = t_{final} = 1$.

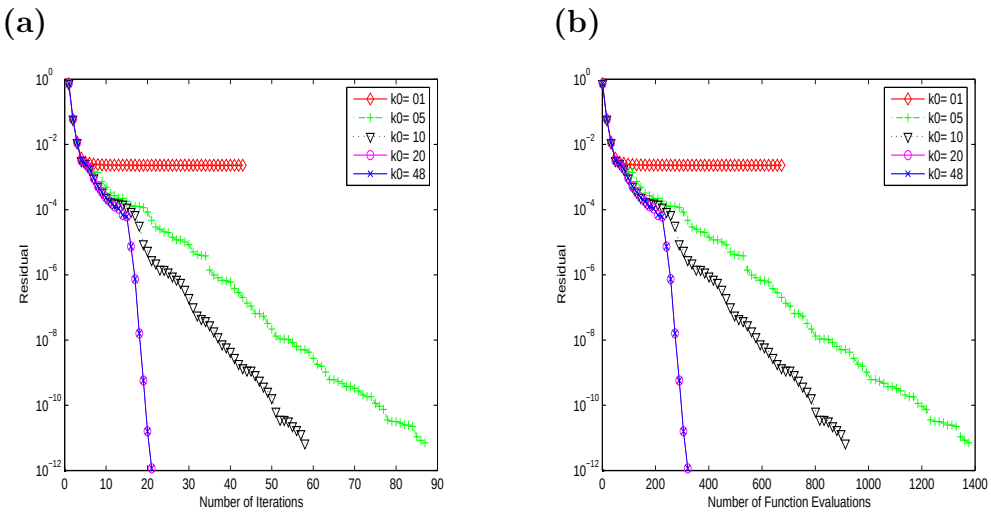

Figure 4.2: Convergence of GMRES(k_0) for different k_0 for a KDC method applied to the linear system as a function of (a) GMRES steps and (b) number of function evaluations.

The total number of unknowns is $N = 16 \times 3$, and hence $k_0 = 48$ is equivalent to the full GMRES procedure. In Fig. 4.2, we study the convergence of the KDC method using different k_0 applied to the preconditioned collocation formulation of Eq. (4.18). Numerical results show that keeping more data in storage (larger k_0) gives better convergence results. However, $k_0 = 20$ results in similar convergence to $k_0 = 48$. In the figure, we plot how the residual decays, as the true error is not readily available from the GMRES subroutine. For this problem, 16 points resolve the solution to 14 significant digits so up to a constant factor, the residual is almost identical to the true error. Also, because Eq. (4.18) is linear, each GMRES iteration (SDC sweep) needs exactly $p = 16$ function evaluations. This explains why the two plots are identical (except for a factor 16 in x-axis).

Finally for this example, note that the KDC method using 9 Radau IIa nodes allows a step-size of 1 (one time step from 0 to 1) for 12 digits of accuracy in y_1 and y_2, with a total number of 162 function evaluations (compared to more than 1000 for the BDF methods in [3]).

4.4.2 The Transistor Amplifier Problem

In our second example, we consider the transistor amplifier problem in [1] which is a stiff DAE system of index 1 consisting of 8 equations given by

$$M\frac{dy}{dt} = f(y), \quad y(0) = y_0, \quad y'(0) = y_0',$$

with $y \in \mathbb{R}^8$ and $0 \leq t \leq 0.2$. The matrix M is of rank 5 and is given by

$$M = \begin{pmatrix} -C_1 & C_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ C_1 & -C_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -C_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -C_3 & C_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & C_3 & -C_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -C_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -C_5 & C_5 \\ 0 & 0 & 0 & 0 & 0 & 0 & C_5 & -C_5 \end{pmatrix}.$$

The function f is defined as

$$f(y) = \begin{pmatrix} -\frac{U_e(t)}{R_0} + \frac{y_1}{R_0} \\ -\frac{U_b}{R_2} + y_2(\frac{1}{R_1} + \frac{1}{R_2}) - (\alpha - 1)g(y_2 - y_3) \\ -g(y_2 - y_3) + \frac{y_3}{R_3} \\ -\frac{U_b}{R_4} + \frac{y_4}{R_4} + \alpha g(y_2 - y_3) \\ -\frac{U_b}{R_6} + y_5(\frac{1}{R_5} + \frac{1}{R_6}) - (\alpha - 1)g(y_5 - y_6) \\ -g(y_5 - y_6) + \frac{y_6}{R_7} \\ -\frac{U_b}{R_8} + \frac{y_7}{R_8} + \alpha g(y_5 - y_6) \\ \frac{y_8}{R_9} \end{pmatrix}$$

where g and U_e are auxiliary functions given by

$$g(x) = \beta(e^{\frac{x}{U_F}} - 1) \text{ and } U_e(t) = 0.1\sin(200\pi t).$$

As in Fig. 4.2 for the linear case, we first consider the effect of using different GMRES(k_0). For this nonlinear system, we use the Newton-GMRES method in which GMRES(k_0) is applied in each Newton iteration to reduce the residual by a factor of η (see Sec. 2.2.2). In the experiment, 16 Radau IIa nodes are used, the step-size is 0.0025 which resolves the solution to 8 digits of accuracy, and η is set to 0.3. In (a) of Fig. 4.3, the residual after each GMRES step (one SDC sweep) is presented for different choices of k_0 and in (b) the residual versus number of function evaluations. It can be seen that $k_0 = 10$ provides results similar to the full GMRES procedure which requires $k_0 = 128$. This is similar to the linear case shown in Fig. 4.2.

For general DAE problems, it is not known how to choose the "optimal" k_0 since the choice depends on the number of Gaussian type nodes and the eigenvalue distribution of the underlying problem. For the remainder of this chapter, we use a simple scheme which selects k_0 to be the smaller of $\{c_1,\ p + N + c_2\}$ where p is the number of node points, N is the number of equations, c_1 is a large constant determined by the computer memory constraints and efficiency considerations, and c_2 is a small constant currently chosen as 5. However this strategy is by no means optimal and better schemes are still being pursued (see the discussion below).

We next provide a comparison of the performance of KDC methods with the MEBDFI and RADAU packages (see [1] for discussions of the two methods). For this problem, adaptive step-size and scheme order selections are essential for optimal efficiency as demonstrated in Fig. 4.4, where for 11 digits of accuracy, the step-sizes used by MEBDFI vary from 10^{-4} to 10^{-14} and those by RADAU from 10^{-3} to 10^{-10}. Nevertheless, we compare the performance of the MEBDFI and RADAU packages with the KDC methods

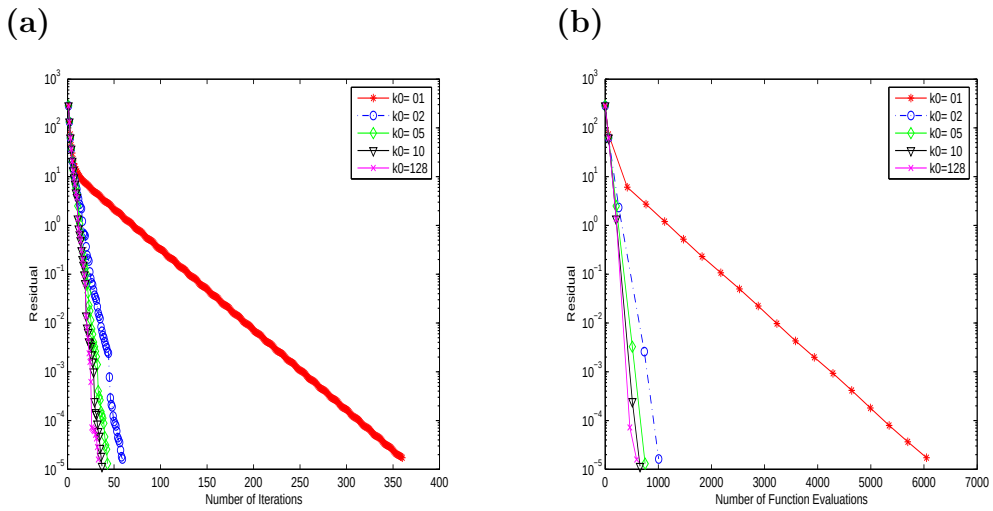

Figure 4.3: Convergence of GMRES(k_0) for different k_0 for a KDC method applied to the nonlinear system as a function of (a) GMRES steps and (b) number of function evaluations.

using uniform time steps. We solve the transistor problem from $t = 0.1$ to $t = 0.2$ to avoid the initial sharp step-size transition region. Fig. 4.5 shows a comparison of the RADAU and MEBDFI packages with KDC methods with $p = 5$, 10, and 20 and a range of fixed time steps.

To give an indication of the disadvantage of using a fixed time step for this example, for the numerical solution using 20 Radau IIa nodes and 200 uniform steps, we compute at each step the Legendre polynomial approximation to the solution (see Section 2.1). We set the error tolerance to 10^{-14} in the Newton-Krylov iterations and hence the solution error mainly comes from the discretization in the collocation formulation (4.10). In the left panel of Fig. 4.6, we plot the magnitude of the coefficient c_{10} for each step and in right panel c_{19}. It can be seen that for most steps, 11 terms in the expansion resolves the solution to 12 digits, however 20 Radau IIa points must be used to resolve the solution to 12 digits in all steps. This indicates that adaptive selection of the number of nodes (or alternatively the size of the time step) would significantly increase the efficiency of the KDC methods for this example.

We are currently studying the issue of adaptively choosing the step-size and scheme

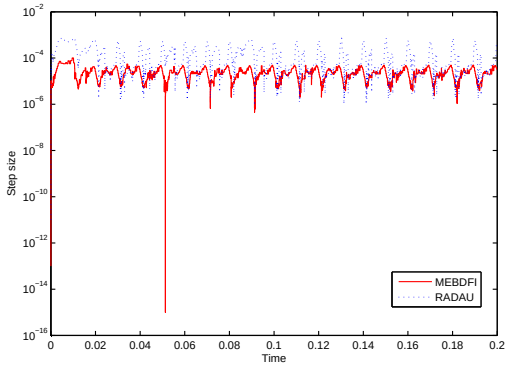

Figure 4.4: Adaptive step-sizes of MEBDFI and RADAU for Transistor problem (10 digits of accuracy).

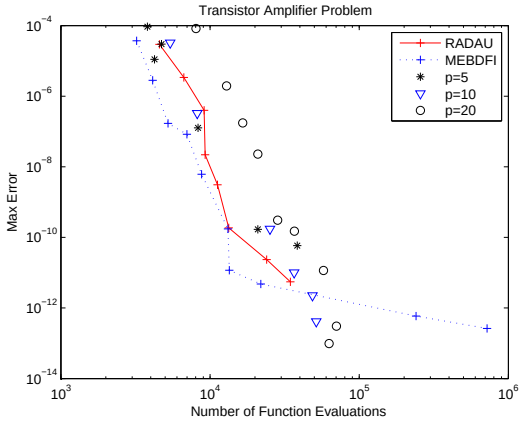

Figure 4.5: Efficiency comparison of the fixed order uniform step KDC method with adaptive RADAU and MEBDFI.

order for KDC methods. This effort must also consider other parameters related to the Newton-Krylov methods (e.g. k_0 and η above). Further possibilites include increasing the number of node points (reflecting the degree of the approximating polynomial) during the Newton-Krylov procedure, and even using different numbers of nodes for different components of the solution vector.

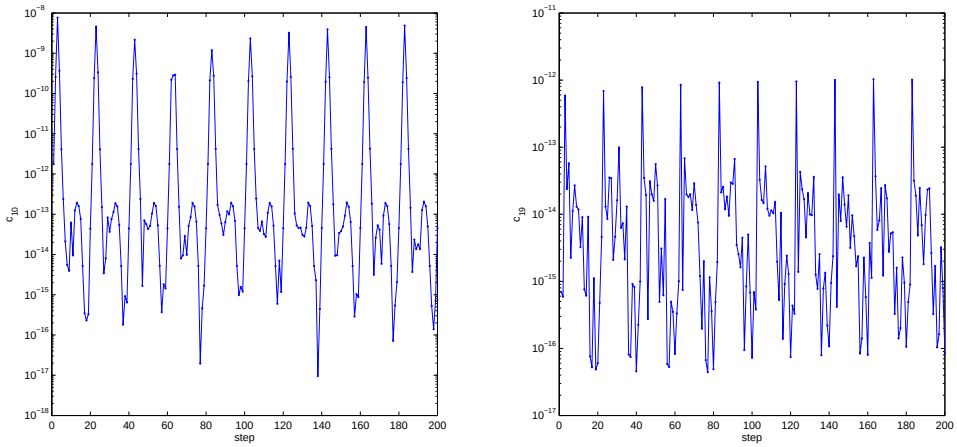

Figure 4.6: Legendre coefficients c_{10} and c_{19}.

4.4.3 The Andrews' Squeezing Mechanism

Andrews' squeezing mechanism describes the motion of 7 rigid bodies connected by joints without friction, which is modeled by a non-stiff second order DAE system of index 3, consisting of 21 differential and 6 algebraic equations, as given by

$$K\frac{dy}{dt} = \Phi(y), \quad , y(0) = y_0, \quad y'(0) = y_0'$$

where

$$y = \begin{pmatrix} q \\ \dot{q} \\ \ddot{q} \\ \lambda \end{pmatrix}, \quad K = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \Phi(y) = \begin{pmatrix} \dot{q} \\ \ddot{q} \\ M(q)\ddot{q} - f(q,\dot{q}) + G^T(q)\lambda \\ g(q) \end{pmatrix}.$$

The index of the unknowns q, $\dot{q}$, $\ddot{q}$ and λ in y is 1, 2, 3, and 3, respectively. We refer interested readers to [1] for explicit forms of functions mentioned above.

As explained in [43], when the original SDC methods are applied to ODE problems, for sufficiently small time step-size Δt, each correction procedure can reduce the residual

by a factor of Δt unless machine precision is reached. However, for most DAE systems we tested, especially for higher index DAE problems, numerical experiments show that the residual from the original SDC methods may no longer converge to zero. In Fig. 4.7, for Andrews' squeezing problem, we use 10 Radau IIa nodes and plot how the residual changes after each original spectral deferred correction in one marching step. Different step-sizes ($\Delta t = 10^{-3}$, 10^{-4}, 10^{-5}, and 10^{-6}) are tested and it can be seen that the residual starts increasing after a few iterations no matter how small the step-size is. In Fig. 4.8, as a comparison, we use similar settings and show the residual after each accelerated Krylov deferred correction, in which the Krylov subspace methods are only applied to the differential components. We notice that for all step-sizes, the residual quickly converges to machine precision.

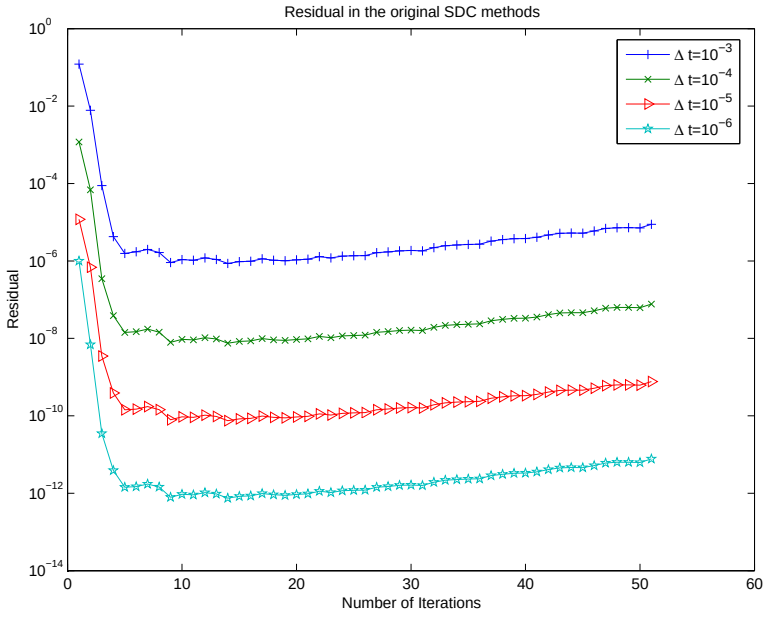

Figure 4.7: Residual in the original SDC method increases after first few corrections.

The convergence of the KDC methods and the divergence of the original SDC techniques can be explained by studying Eq. (4.17). In [43], it was shown that the original SDC methods are equivalent to a Neumann series expansion for the preconditioned system. For ODE problems, as $\frac{\partial \vec{F}}{\partial \mathbf{Y}} = I$, the matrix C in Eq. (4.17) is of order $O(\Delta t)$ as long

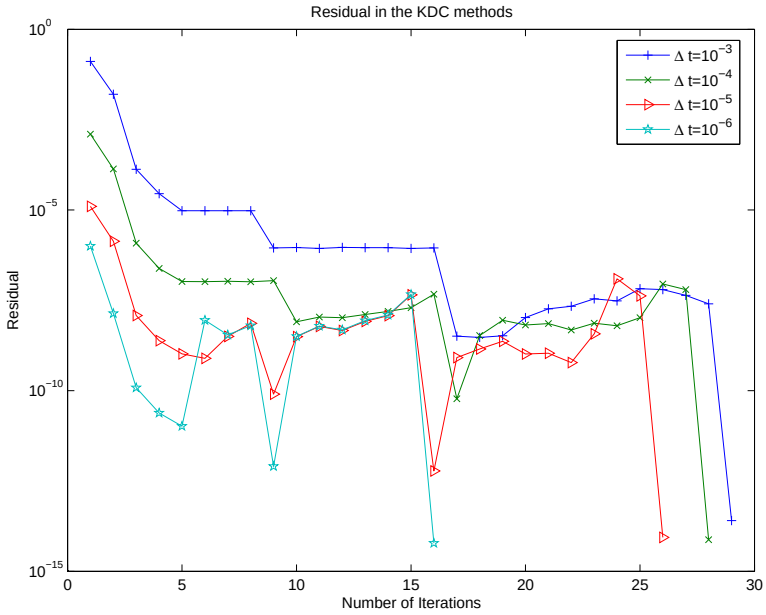

Figure 4.8: Residual in the KDC methods converges to machine precision.

as Δt is sufficiently small. In this case, the residual always converges to zero and each spectral deferred correction increases the residual order by Δt. For DAE problems, due to the existence of algebraic equations, $\frac{\partial \vec{F}}{\partial \mathbf{Y}}$ may be singular, and some eigenvalues in C may be greater than 1 in magnitude, which explains the increasing residual for the original SDC methods. When Krylov subspace methods are applied, however, the existence of such eigenvalues is no longer a problem. Therefore, considering the deferred/defect correction procedures as preconditioners for the collocation formulation and introducing the Newton-Krylov techniques analytically guarantee the local convergence and significantly improve the convergence rate for DAE problems.

Finally for this problem, we want to mention that in the KDC methods, the residuals may increase during iterations as shown in Fig. 4.8. This is due to the inaccuracy in the forward difference approximations in the Newton-Krylov methods and the nonlinearity of the system. Currently we are adapting the Newton-Krylov methods to improve the accuracy of the forward difference approximations. An immediate advantage is that (almost) decreasing residuals will provide better error control strategies.

4.4.4 The Wheelset Problem

In our last example, we consider the wheelset problem described by a DAE system of dimension 17 (also referred to as an implicit differential equation (IDE) system)

$$\frac{dp}{dt} = v, \tag{4.19}$$

$$M(p)\begin{pmatrix} \frac{dv}{dt} \\ \frac{d\beta}{dt} \end{pmatrix} = \begin{pmatrix} f(u) - (\partial g_1(p,q)/\partial p)^T C\lambda \\ d(u) \end{pmatrix}, \tag{4.20}$$

$$0 = g_1(p,q), \tag{4.21}$$

$$0 = g_2(p,q), \tag{4.22}$$

where $u = (p, v, \beta, q, \lambda)^T \in \mathbb{R}^{17}$, $p, v \in \mathbb{R}^5$, $\beta \in \mathbb{R}$, $q \in \mathbb{R}^4$, $\lambda \in \mathbb{R}^2$ and C is a scalar constant. Furthermore, $M : \mathbb{R}^5 \to \mathbb{R}^6 \times \mathbb{R}^6$, $f : \mathbb{R}^{17} \to \mathbb{R}^5$, $d : \mathbb{R}^{17} \to \mathbb{R}$, $g_1 : \mathbb{R}^9 \to \mathbb{R}^2$ and $g_2 : \mathbb{R}^9 \to \mathbb{R}^4$. This problem shows some typical properties of simulation problems in contact mechanics, i.e., friction, contact conditions, stiffness, etc.. It is an index 3 IDE system but can be reduced to index 2. Interested readers are referred to [1] for the initial conditions, the function forms of M, f, d, g_1 and g_2, as well as more detailed discussions of the problem. In the following, similar to [1], we present test results based on the index-2 formulation where Eq. (4.21) is replaced by

$$0 = (\partial g_1(p,q)/\partial p)v.$$

For this test, we march with uniform time-step from $t_0 = 0$ to $t_{final} = 0.002$ (a region in which relative uniform step-sizes are used by the compared methods). A comparison of the performance of KDC methods using 4 and 8 nodes and various fixed time steps with the DASSL, MEBDFI and PSIDE codes is shown in Fig. 4.9. Our numerical experiments show that the MEBDFI method requires 159 function evaluations for 5 digits of accuracy and 581 for 12 digits. On the other hand, the new KDC method with 4 Radau IIa nodes

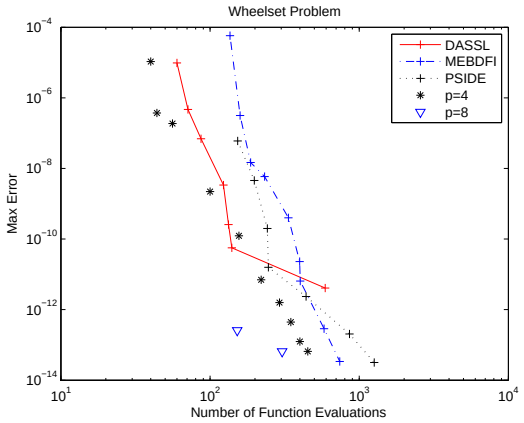

Figure 4.9: Efficiency comparison of the uniform step KDC method with adaptive DASSL, MEBDFI and PSIDE.

requires 40 function evaluations for 5 digits and 348 for 12 digits. The KDC method with 8 nodes uses 152 function evaluations for 12 digits.

Finally, because of the excessive storage requirements of GMRES(k_0), we present here a comparison of alternative Krylov subspace methods applied to the wheelset problem. Specifically, we consider the biconjugate gradients stabilized (BiCGStab) method and transpose-free quasi-minimal residual (TFQMR) algorithm (See [7] for a summary of existing Newton-Krylov methods). The storage required in both methods is independent of iteration number k, and the number of multiplications grows only linearly as a function of k. In Fig. 4.10, for the wheelset problem, we compare the convergence of the full GMRES procedure with BiCGStab and TFQMR in terms of (a) number of iterations and (b) number of function evaluations. In the simulation, we use $p = 4$ Radau IIa nodes and set $\Delta t = t_{final}$. It can be seen that both BiCGStab and TFQMR converge to the prescribed accuracy after numbers of iterations fewer than full GMRES, with similar numbers of function evaluations. Similar numerical results for $p = 8$ are shown in Fig. 4.11. Comparisons of different Krylov subspace methods as well as the optimal choices of different parameters (step-size, k_0, η, etc.) are being studied.

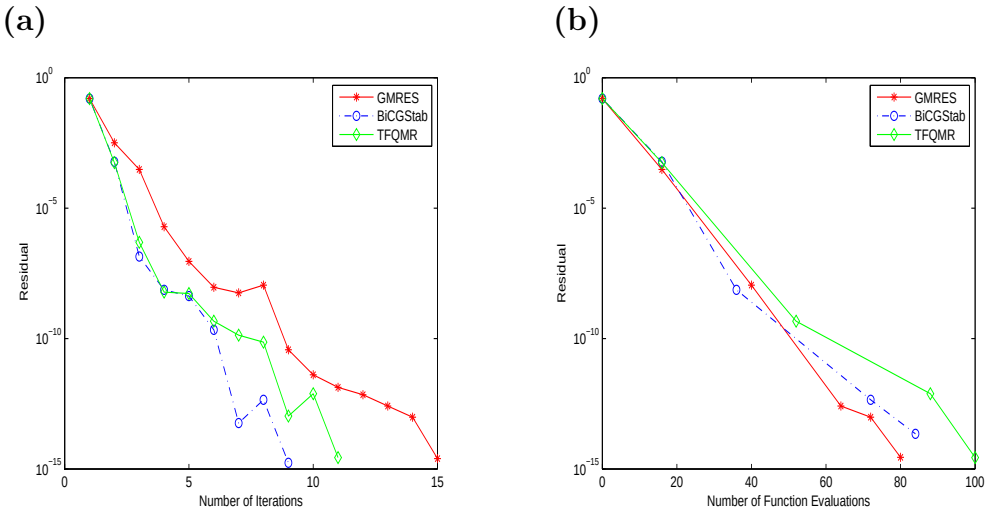

Figure 4.10: Convergence of KDC iterations with 4 Radau IIa points for different Newton-Krylov methods

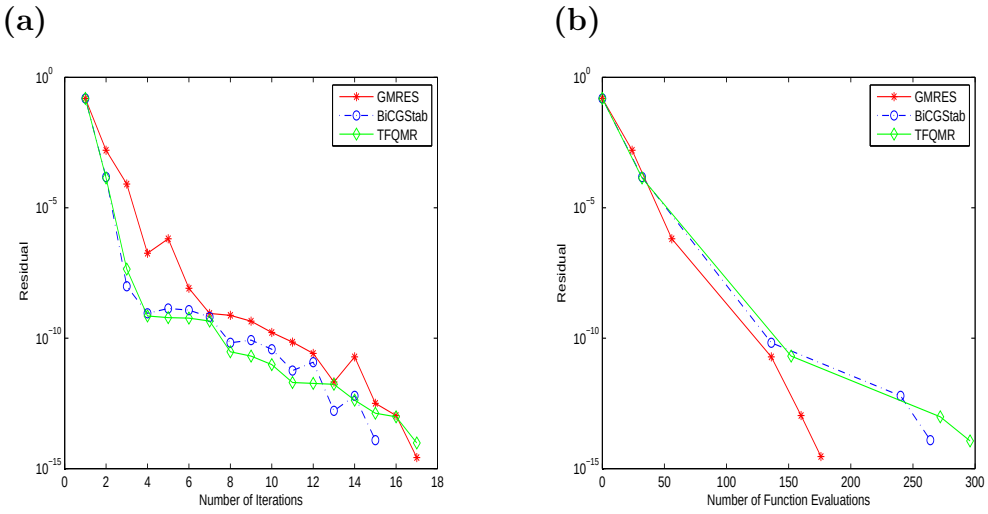

Figure 4.11: Convergence of KDC iterations with 8 Radau IIa points for different Newton-Krylov methods

Chapter 5

Method of Lines Transpose for Partial Differential Equations

In this chapter, we generalize the results from previous chapters and discuss how time dependent PDEs (in particular, parabolic type PDEs) can be efficiently solved by coupling Krylov deferred correction methods and fast elliptic equation solvers in conjunction with method of lines transpose (MoL^T).

This chapter is organized as follows. In Sec. 5.1, we introduce parabolic type partial differential equation problems. MoL^{T}is then implemented in Sec. 5.2, and analyses of the method are briefly discussed in Sec. 5.3. In Sec. 5.4, preliminary numerical results are presented.

5.1 Time Dependent Initial Value PDEs

Efficient and accurate solutions of time dependent initial value partial differential equations are required in simulations of many science and engineering systems. In this chapter, we consider a general parabolic type partial differential equation (PDE) initial value problem of the form

$$\mathcal{L}\left(u_t, u, u_x, u_{xx}\right) = 0 \tag{5.1}$$

where $u = u(x,t)$, $x \in [a,b]$, $t \in [0,T]$, with proper initial and boundary conditions. Specific examples include the well-known diffusion (heat) equation

$$u_t = Du_{xx} + f(x,t),$$

where D is the diffusion coefficient; the diffusion-reaction equation

$$u_t = Du_{xx} + f(u),$$

which models many biological and chemical reaction processes; the nonlinear Schrödinger equation (NLS)

$$i\epsilon u_t + \frac{\epsilon^2}{2}u_{xx} - Vu - f(\|u\|^2)u - \epsilon\tau arg(u)u = 0$$

in solid state physics; and the Richards' equation

$$[c(u) + S_s S_a(u)]u_t = [K_x(u)(u_x + 1)]_x$$

which simulates fluid flow and species transport in subsurface systems. In Eq. (5.1), u may be a vector of unknowns and hence $\mathcal{L} = 0$ contains a system of equations. To simplify the discussions and focus on the ideas, we describe our algorithms and numerical results in $1+1$ dimensions. However, generalization of the analyses and algorithms to higher dimensions is straightforward.

As introduced in Chapter 1, there are typically two discretization strategies for time dependent PDEs. Perhaps due to the popularity and success of numerical methods for solving initial value problems governed by systems of ordinary differential equations (ODEs) and differential algebraic equations (DAEs), the more common practice is method of lines (MoL) technique. To recap, MoL first "discretize" the PDEs in the spatial direction using finite difference, finite element, or spectral methods, and each entry in the

solution vector represents the approximate solution at a specific location (or frequency) for all times. Then available solvers are applied to the resulting ODE/DAE systems. As discussed in Chapter 1, the major drawbacks of this type of schemes are expensive spatial adaptivity and that after the spatial discretization, the equations are no longer "elliptic", hence existing fast elliptic equation solvers can not be easily applied. For all these reasons, we prefer the alternative approach, Rothe's method, which discretizes in time first, leading to an elliptic system to be solved, and marches.

By introducing ideas from Rothe's method and fast integral equation methods for elliptic equations, the purpose of this chapter is to generalize the Krylov deferred correction methods for DAE problems to efficient solutions of Eq. (5.1). In addition to parabolic type problems, hyperbolic type equations could also be considered. Unfortunately, as discontinuity may appear in forms of shocks in hyperbolic systems and the solutions lose smoothness, higher order and spectral methods may no longer be advantageous compared with existing low order schemes.

5.2 Method of Lines Transpose

In this section, we discuss our implementation of MoL^T. For simplicity of discussion, we focus on the $1+1$ dimensional parabolic PDEs described by Eq. (5.1).

5.2.1 Spectral Integration and Collocation Formulation

To march the evolutionary PDE (5.1) from $t = 0$ to Δt, similar to KDC and Rothe's methods, the MoL^{T}first discretizes the PDE in the temporal direction (the transpose direction of traditional MoL) using p nodes $\mathbf{t} = [t_1, t_2, \cdots, t_p]^T$. Instead of uniform node points on which higher order interpolations are unstable, the MoL^{T}uses Gaussian type nodes (including Gaussian, Radau, and Lobatto quadrature nodes) and the Legendre polynomial interpolation. On each node point, to avoid numerically unstable differenti-

ation operator, we define $U_i(x) = U(x, t_i) = u_t(x, t_i)$ as the new unknowns, and recover $u_i(x) = u(x, t_i)$ by integrating the interpolating Legendre polynomial (the coefficients are computed using corresponding quadrature rules). The linear mapping from functions $\{U_i(x), i = 1, \cdots, p\}$ to $\{u_i(x), i = 1, \cdots, p\}$ is also referred to as the "spectral integration" operator, and the corresponding matrix denoted by S is the spectral integration matrix (see [27] and previous chapters). The discretized equation becomes

$$\mathbf{L}\left(\mathbf{U}, \mathbf{u_0} + \Delta t S \otimes \mathbf{U}, \frac{d}{dx}\left(\mathbf{u_0} + \Delta t S \otimes \mathbf{U}\right), \frac{d^2}{dx^2}\left(\mathbf{u_0} + \Delta t S \otimes \mathbf{U}\right)\right) = \mathbf{0} \tag{5.2}$$

where $\mathbf{U} = [U_1(x), U_2(x), \cdots, U_p(x)]^T$ is the desired approximation of $U(x, t)$ at different node points and $\mathbf{u_0} = [u(x, 0), u(x, 0), \cdots, u(x, 0)]^T$ is the initial condition. As Gaussian type nodes are used in the MoL^T, this collocation formulation can be spectrally accurate for non-degenerating parabolic equations. Also note that this discretization preserves the elliptic properties in the spatial direction.

5.2.2 Error Equation and Spectral Deferred Correction

The discretized collocation formulation in Eq. (5.2) consists of a group of elliptic equations coupled by the dense spectral integration matrix S. Its direct solution is in general computationally inefficient. Instead, similar to KDC methods, we assume a provisional solution $\tilde{\mathbf{U}} = [\tilde{U}_1(x), \tilde{U}_2(x), \cdots, \tilde{U}_p(x)]^T$ is given, and define the error as $\boldsymbol{\delta} = \mathbf{U} - \tilde{\mathbf{U}} = [\delta_1, \delta_2, \cdots, \delta_p]^T$. The continuous counterparts of $\boldsymbol{\delta}$ and $\tilde{\mathbf{U}}$ derived by polynomial interpolation in time are denoted as $\delta(x, t)$ and $\tilde{U}(x, t)$, respectively. A simple substitution yields the error equation for $\delta(x, t)$

$$\mathcal{L}\left(\tilde{U} + \delta, u_0 + \int_0^t (\tilde{U} + \delta) d\tau, \frac{d\left(u_0 + \int_0^t (\tilde{U} + \delta) d\tau\right)}{dx}, \frac{d^2\left(u_0 + \int_0^t (\tilde{U} + \delta) d\tau\right)}{dx^2}\right) = 0 \tag{5.3}$$

where $u_0 = u(x,0)$, and the variables in $\tilde{U}(x,t)$ and $\delta(x,t)$ are omitted in the notation. Applying spectral integration to the integrals in this equation, the discretized "error equation" is then given by

$$\mathbf{L}(\tilde{\mathbf{U}}+\boldsymbol{\delta}, \mathbf{u_0}+\Delta tS\otimes(\tilde{\mathbf{U}}+\boldsymbol{\delta}), \frac{d}{dx}(\mathbf{u_0}+\Delta tS\otimes(\tilde{\mathbf{U}}+\boldsymbol{\delta})), \frac{d^2}{dx^2}(\mathbf{u_0}+\Delta tS\otimes(\tilde{\mathbf{U}}+\boldsymbol{\delta}))) = \mathbf{0}, \quad (5.4)$$

where ΔtS is a higher order approximation of the operator $\int_0^t$.

Notice that $\int_0^t$ can also be approximated by lower order integration rules as in KDC methods, such as the rectangular rule using left end point (explicit Euler method), and the rectangular rule using right end point (implicit Euler method). We re-apply a lower order integration $\tilde{S}$ to the unknown $\boldsymbol{\delta}$. A lower order approximation $\bar{\boldsymbol{\delta}}$ of $\boldsymbol{\delta}$ can then be obtained by solving the equation system

$$\mathbf{L}(\tilde{\mathbf{U}}+\bar{\boldsymbol{\delta}}, \mathbf{u_0}+\Delta tS\otimes\tilde{\mathbf{U}}+\Delta t\tilde{S}\otimes\bar{\boldsymbol{\delta}}, \frac{d}{dx}(\mathbf{u_0}+\Delta tS\otimes\tilde{\mathbf{U}}+\Delta t\tilde{S}\otimes\bar{\boldsymbol{\delta}}), \frac{d^2}{dx^2}(\mathbf{u_0}+\Delta tS\otimes\tilde{\mathbf{U}}+\Delta t\tilde{S}\otimes\bar{\boldsymbol{\delta}})) = \mathbf{0}. \quad (5.5)$$

Notice that the equations for $\bar{\boldsymbol{\delta}}$ are now decoupled at different times and the elliptic equation at each node t_i can be solved efficiently using available elliptic equation solvers, which are discussed in Sec. 2.3 for linear problems and in Sec. 5.2.4 for nonlinear cases.

5.2.3 Newton-Krylov Methods and MoL^T

Eq.(5.5) can be considered as an "implicit function" where $\tilde{\mathbf{U}}$ is the input variable and $\bar{\boldsymbol{\delta}}$ is the output function value. Similar to the DAE case, we symbolically denote the explicit form of this function as

$$\bar{\boldsymbol{\delta}} = \tilde{\mathbf{H}}(\tilde{\mathbf{U}}). \quad (5.6)$$

When $\tilde{\mathbf{U}}$ solves the collocation formulation in Eq. (5.2), we have $\bar{\boldsymbol{\delta}} = \mathbf{0}$. Therefore, solving the collocation formulation is equivalent to finding the zero of Eq. (5.6) and we

call (5.7) the "preconditioned" formulation.

$$\bar{\boldsymbol{\delta}} = \tilde{\mathbf{H}}(\tilde{\mathbf{U}}). \tag{5.7}$$

Compared with the original collocation formulation, the "preconditioned" formulation in (5.7) is better conditioned. It is straightforward to show that the Jacobian matrix of the function $\tilde{\mathbf{H}}(\tilde{\mathbf{U}})$ is closer to I and we neglect the details. Hence the Newton-Krylov methods can be efficiently applied to find the zero of $\tilde{\mathbf{H}}(\tilde{\mathbf{U}}) = \mathbf{0}$. As for the second requirement for the efficient implementation of Newton-Krylov methods, when forward difference approximation is applied, each function evaluation is simply one spectral deferred correction in which fast elliptic equation solvers can be applied.

As a summary, the new scheme consists of the following building blocks: (a) Newton-Krylov methods are applied to the preconditioned formulation in Eq. (5.7); (b) each function evaluation is simply one spectral deferred correction in which the elliptic equations are decoupled; and (c) the decoupled elliptic equations are solved efficiently using available fast IEM solvers as discussed in next section.

5.2.4 Fast Nonlinear Two Point Boundary Value ODE Solver

In Chapter 2, we discussed existing fast linear elliptic equation solvers based on integral equation formulations. For nonlinear elliptic equations in the MoL^{T}discretization, Newton's method can be coupled with the fast linear equation solvers as described by the following formulas.

Suppose the boundary value elliptic problem has the form

$$\mathcal{N}(x, u(x), u_x(x), u_{xx}(x)) = 0. \tag{5.8}$$

Define

$$\begin{aligned} F(x) =& \mathcal{N}(x, v(x), v_x(x), v_{xx}(x)), \\ H(x) =& D_v\mathcal{N}(x, v(x), v_x(x), v_{xx}(x)), \\ J(x) =& D_{v_x}\mathcal{N}(x, v(x), v_x(x), v_{xx}(x)), \\ K(x) =& D_{v_{xx}}\mathcal{N}(x, v(x), v_x(x), v_{xx}(x)), \end{aligned} \tag{5.9}$$

where $v(x)$ is an initial approximation of $u(x)$. Then one step of the Newton's method requires the solution of the linearized equation for the Newton correction $e(x)$:

$$K(x)e_{xx}(x) + J(x)e_x(x) + H(x)e(x) + F(x) = 0. \tag{5.10}$$

Note that F, H, J and K are v-dependent. This linear equation is then solved by the linear solver introduced in Chapter 2 and the provisional solution $v(x)$ is updated by $v_{new}(x) = v(x) + e(x)$. This Newton iteration repeats until $e(x)$ or $F(x)$ is sufficiently small or a maximal number of iterations is reached (in which case global convergence can not be achieved and a smaller time step has to be used in the marching scheme), and the final converged $v(x)$ solves Eq. (5.8).

5.3 Algorithm Analysis

As a new approach for time dependent PDEs, many theoretical aspects of MoL^T have not been studied. Some analyses have been done for Rothe's method, and please refer to [10, 49, 65] and the references therein.

In this section, we discuss several available results concerning the accuracy, stability, and efficiency of the method.

The accuracy of the MoL^T is determined by (a) the accuracy of the collocation formulation in Eq. (5.2); (b) the accuracy of the Newton-Krylov methods, and (c) the accuracy

of the fast elliptic equation solvers used. For (a), as the Gaussian type nodes are used, it is not hard to see that the resulting formulation is spectrally accurate. Also, when p Gaussian nodes are used and the elliptic equation system is solved exactly, the resulting scheme has order $2p$ in time for the diffusion equation. When Radau nodes are used, the collocation formulation has order $2p-1$. We want to mention that for PDEs with algebraic constraints, order reduction may be observed and we refer the readers to [35, 34] for details. The convergence and accuracy of the Newton-Krylov methods in (b) have also been widely studied previously. It can be shown that super-linear local convergence can be obtained for specially chosen parameters in the Newton-Krylov schemes (see Theorem 6.1.2 in [45]). For arbitrary initial approximations, continuation/homotopy methods are necessary to accomplish global convergence. Interested readers are referred to [46, 45] for further discussions. For (c), instead of the "order of convergence" concept commonly used in the finite difference and finite element methods, most recently developed fast IEM solvers generate numerical results with prescribed accurate digits, therefore the accuracy in (c) is guaranteed.

In [34], it was shown that for ODE problems, the collocation formulation is equivalent to certain Runge-Kutta methods (see p.27 in [34]). When Gaussian points are used in the temporal discretization, the method is A-stable, B-stable, and L-stable. Based on these existing results and assume the spatial elliptic equations are solved exactly, we conclude that the MoL^{T}is unconditionally stable. Also, as shown in next section, no CFL constraints have been observed in the numerical experiments. Finally, the efficiency of the MoL^{T}is determined by the number of iterations in the Newton-Krylov procedure, which depends on the structure of the problem, the low order preconditioner (Euler, trapezoidal rule, splitting methods, etc.), and the Krylov method (GMRES, BiCGStab, TFQMR, etc.).

5.4 Numerical Experiments

In this section, we show some preliminary numerical results. The new methods are currently implemented in Matlab and Fortran, and Radau IIa nodes are used in the spectral integration for the temporal direction. We utilize available Newton Krylov method packages. For Matlab code, we use the codes from C.T.Kelly [45] while for Fortran we use the NITSOL package by Pernice and Walker [61].

5.4.1 The Diffusion Equation

For the first example, we consider the simplest parabolic type equation

$$\frac{\partial \psi}{\partial t} = \frac{\partial^2 \psi}{\partial x^2} + f(x,t),$$

which is often referred to as the heat or diffusion equation. We assume zero initial condition and time varying boundary conditions, and choose the real solution as $\psi = t\exp(kx - t)$, where $x \in [-1, 1]$. The heat source $f(x,t) = (1 - t - k^2 t)\exp(kx - t)$ is determined accordingly. For this example, we apply the spatial solver mentioned earlier in sections 5.2.4 and 2.3.1. and request the solution in the spatial domain to be fully resolved by setting the error tolerances in the boundary value problem solver and the Newton-Krylov solver to machine precision. The dominant error is hence from the temporal discretization only. The code is written in Fortran.

The numerical tests run from $t = 0$ to $t = 10$. As shown in Fig. 5.1, the method shows desired order $2 \cdot p - 1$ when p Radau IIa points are used in the KDC integrator. We also compare the number of iterations required by the original SDC and the new KDC methods. The results are shown for one time step of the method using step size $\Delta t = 10$ in Fig. 5.2, and $\Delta t = 0.1$ in Fig. 5.3, respectively. Clearly, the KDC methods reduce the number of iterations dramatically compared with the original SDC techniques.

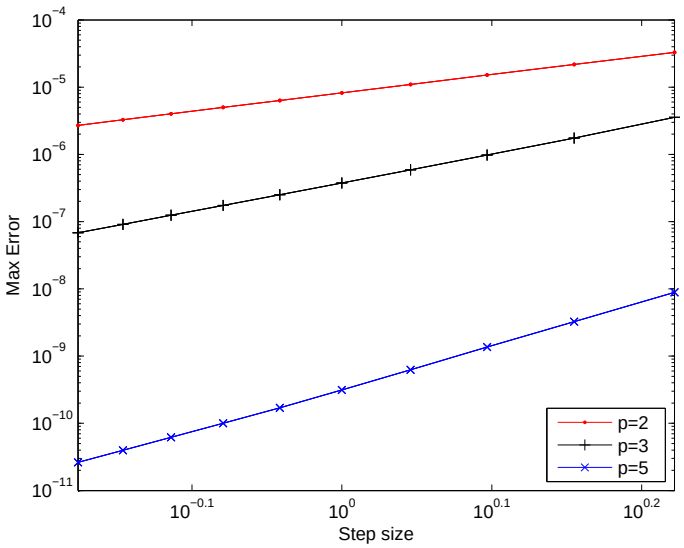

Figure 5.1: Convergence test, p Radau IIa nodes

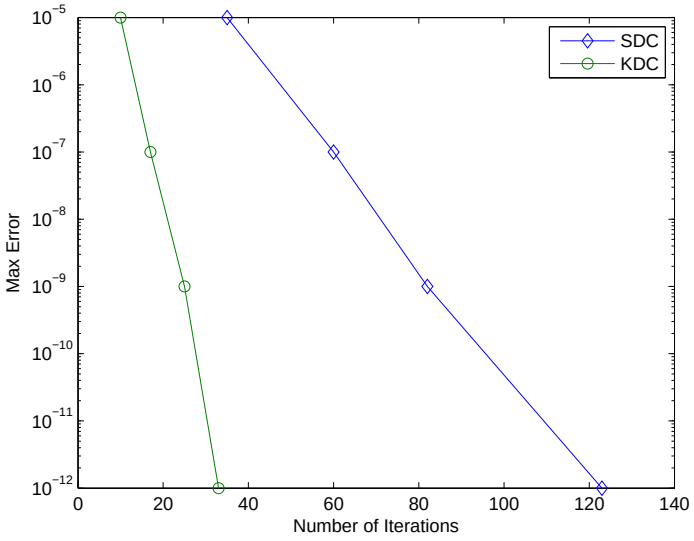

Figure 5.2: Comparison of number of iterations, $p = 20$ Radau IIa nodes

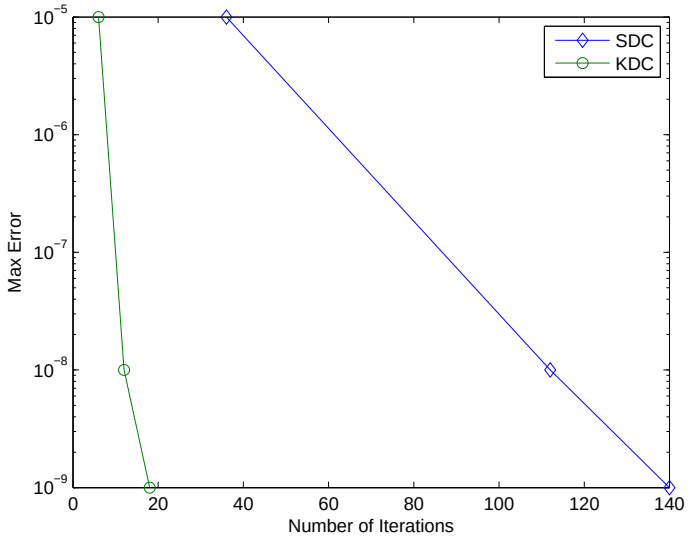

Figure 5.3: Comparison of number of iterations, $p = 5$ Radau IIa nodes

5.4.2 A Variable Coefficient Parabolic Equation

For the second example, we consider a variable coefficient parabolic system defined on $x \in [0, 1], t \in [0, T]$ of the form

$$M(x)\frac{\partial \psi}{\partial t} = N(x)\frac{\partial^2 \psi}{\partial x^2} + f(x, t).$$

We assume periodic boundary conditions and the initial value $\psi_0(x) = e^{\cos(2\pi x)}$ at $t = 0$. The analytical solution is given by

$$\psi(x, t) = e^{\cos(2\pi(x+t^2))-kt},$$

and $M(x)$ and $N(x)$ are defined as

$$M(x) = 2 + \cos(2\pi x),$$

$$N(x) = 2 + \cos(4\pi x).$$

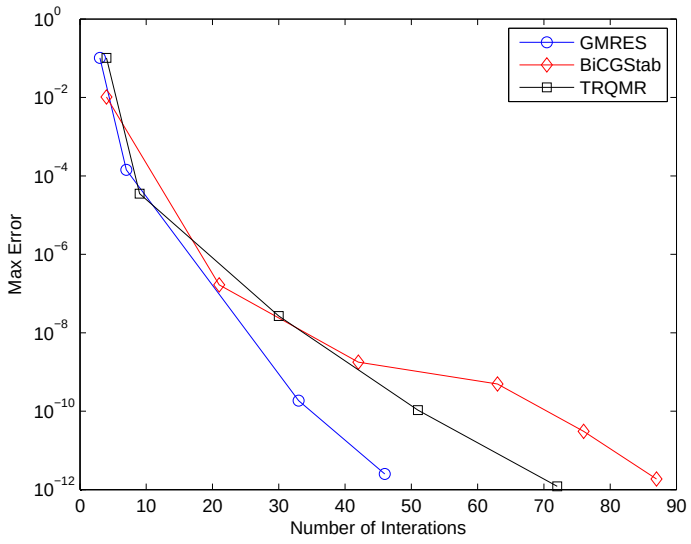

Figure 5.4: Comparison of different Krylov subspace methods, $p = 32$ Radau IIa nodes

$f(x,t)$ is determined accordingly.

Since all the functions are periodic in space, we discretize the spatial domain using equi-spaced grids and approximate the solution using Fourier series expansion whose coefficients are computed using the spectrally accurate trapezoidal rule. The numerical code is implemented in Matlab.

We run the tests from $t = 0$ to $t = 0.5$ for one time step with 32 Radau IIa nodes, and the spatial direction is fully resolved by a truncated Fourier series expansion with 64 terms. Numerical experiments reveal that the MoL^{T}converges more efficiently than the unaccelerated SDC methods for this problem. As the results are very similar to the diffusion equation case, we neglect the details. Instead, as GMRES may require prohibitive storage for PDE problems, in this example, we compare the results using different Krylov subspace methods. As shown in Fig. 5.4, compared with GMRES, alternative BiCGStab and TFQMR methods also provide satisfactory convergence results using less storage.

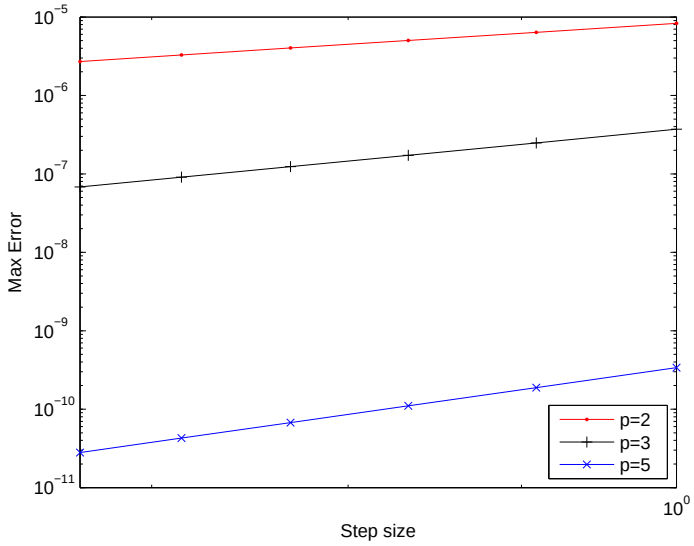

Figure 5.5: Convergence test, p Radau IIa nodes

5.4.3 A Nonlinear Parabolic Equation

Our third example is a nonlinear parabolic problem given by

$$\frac{\partial \psi}{\partial t} = (\psi^{3/2} + 1)\frac{\partial^2 \psi}{\partial x^2} + 2\psi \frac{\partial \psi}{\partial x}(\frac{\partial \psi}{\partial x} + 1) + f(x,t). \tag{5.11}$$

It is designed to mimic the Richards' equation to be discussed in next section. We choose the analytical solution as $\psi = t\exp(kx - t)$ for $x \in [-1, 1]$ to avoid the numerical difficulties related with the sharp fronts in the Richards' equation in this example (the sharp fronts will be discussed in next section). The corresponding initial condition, boundary conditions, and $f(x,t)$ are set accordingly.

For this example, we test the convergence of the methods for nonlinear system. The solution is obtained from $t = 0$ to $t = 10$. As shown in Fig. 5.5, very similar behaviors can be observed as in the linear case. We want to mention that the solution is resolved to machine precision in the spatial domain and solved by the direct method introduced in Sec. 5.2.4.

5.4.4 Richards' Equation

The Richards' Equation (RE) is a highly nonlinear time dependent parabolic system describing flows in porous media [54, 41]. While a standard and frequently used model, the highly nonlinear nature of this equation when solved by typical closure relations and common numerical approximation methods can lead to a series of difficulties, including the loss of mass conservation, poorly resolved fronts, and failure for nonlinear or iterative linear solvers. Although these problems can be resolved by introducing a mass-conservative formulation, finer discretization in space and time, and carefully implemented robust linear and nonlinear equation solvers, these approaches are often computationally expensive, especially when the solution involves sharp fronts in space and time. The efficient and accurate solution of the RE has been a very challenging numerical topic. The RE can be posed in different forms, in the following, we focus on the pressure-head form given by

$$A\frac{\partial \psi}{\partial t} = \frac{\partial}{\partial z}\left[\hat{\rho}K\left(\frac{\partial \psi}{\partial z}+\hat{\rho}\right)\right] \qquad \texttt{in } \Omega \times t \in [0,T] \tag{5.12}$$

$$A = \theta\frac{\partial \hat{\rho}}{\partial \psi}+\hat{\rho}\frac{\partial \theta}{\partial \psi} \tag{5.13}$$

$$\psi = \frac{p}{\rho_0 g} \tag{5.14}$$

$$K = k_r K_s \tag{5.15}$$

$$K_s = \frac{\rho_0 g k}{\mu} \tag{5.16}$$

$$\rho = \rho_0 e^{\beta(p-p_0)} \tag{5.17}$$

$$\hat{\rho} = \frac{\rho}{\rho_0} \tag{5.18}$$

$$S_e = \frac{\theta-\theta_r}{\theta_s-\theta_r} \tag{5.19}$$

and

$$S_e(\psi) = \begin{cases} (1+ \mid \alpha_\nu \psi \mid^{n_\nu})^{-m_\nu} & \text{for } \psi < 0 \\ 1, & \text{for } \psi \geq 0 \end{cases} \tag{5.20}$$

$$k_r = \begin{cases} S_e^{1/2}[1-(1-S_e^{1/m_\nu})^{m_n u}]^2, & \text{for } \psi < 0 \\ 1, & \text{for } \psi \geq 0 \end{cases} \tag{5.21}$$

with initial and boundary conditions

$$\begin{aligned} \psi &= \psi^0 \quad \texttt{in } \Omega, t = 0 \\ \psi &= \psi^b \quad \texttt{on } \Gamma_D, t \in [0,T] \\ q &= q^b \quad \texttt{on } \Gamma_N, t \in [0,T] \end{aligned} \tag{5.22}$$

where ψ^0 is the given initial value; ψ^b is a first-kind boundary condition on boundary Γ_D; the specific discharge q is given by q^b on second-kind boundary Γ_N; and $\Gamma = \Gamma_D \cup \Gamma_N$.

Several numerical schemes have been successfully implemented for this equation, in particular, we want to mention recent work by Miller et al. [57], in which the Richards' equation is solved using adaptive strategies based on traditional spatial discretizations (finite difference and finite element methods) and existing ODE/DAE initial value problem solvers. More specifically, the spatial adaption utilizes a coarse grid solve and a gradient error indicator with the finite difference or discontinuous Galerkin approximation, and the temporal adaption is accomplished using variable order, variable step size approximations based upon the backward difference formulas (BDF) based DAE solvers of orders up to 5.

In this example, we apply the MoL^{T}to the Richards' equation, using the simulation parameter profile of "problem I" described in [57]. In Fig. 5.6, numerical result at a specific time is presented by plotting the solutions on the adaptive spatial grid points used by the fast adaptive elliptic solver. In Fig. 5.7, the solution profiles are presented

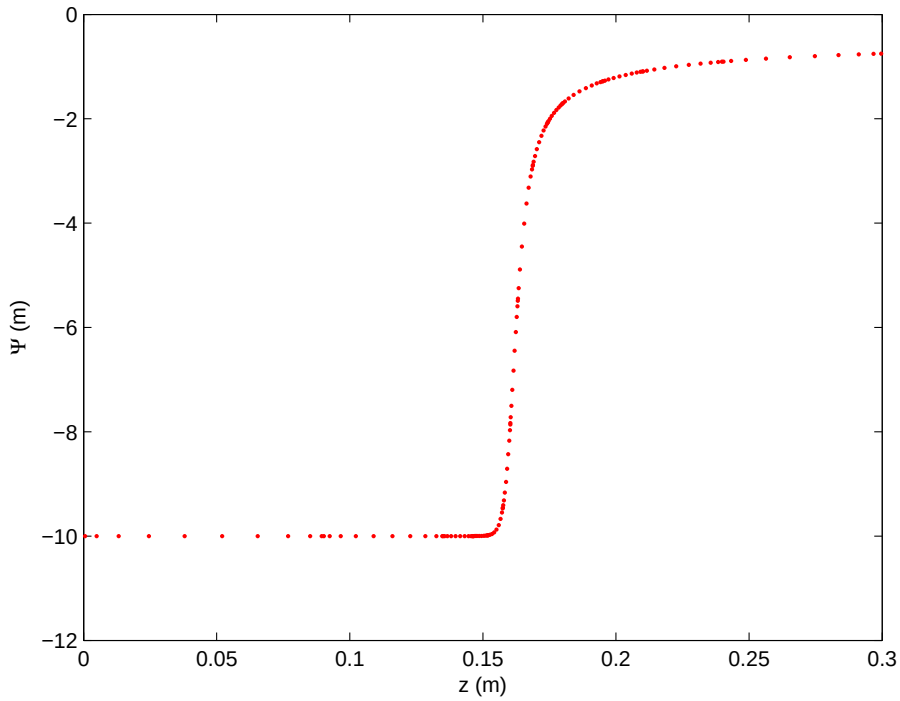

Figure 5.6: Automatic spatial adaption

for different times and it can be seen that the sharp moving front is captured properly.

However, we want to mention that the current code is very primitive. Our preliminary numerical results show that although very promising, the non-optimized code still can not compete with existing state-of-the-art solvers. Currently, we are implementing a fully spatial and temporal adaptive code, and improving the performance by adapting and optimizing existing Newton-Krylov solvers, by generating better initial guess for Newton's methods, and by selecting optimal time stepping and order control strategies. These will be discussed briefly in Chapter 6.

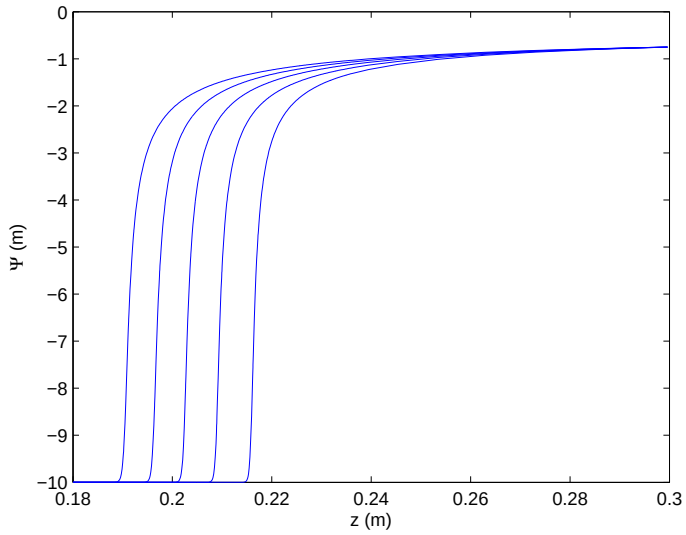

Figure 5.7: Solution profiles for pressure head

Chapter 6

Summary and Recommendations

In this dissertation, a new framework for the efficient and accurate solution of differential equations with algebraic constraints is presented. For ODE and DAE problems, the KDC methods utilize existing spectral deferred correction methods as preconditioners, and apply Newton-Krylov methods to the preconditioned system for optimal efficiency. For PDEs, the MoL^{T}technique first discretizes the temporal direction, and couples KDC methods with the highly efficient and fast IEM solvers for elliptic equations. Due to the use of the Gaussian type nodes, Picard-type integral equation, and spectral integration, the new methods are unconditionally stable and can be spectrally accurate in the temporal direction.

However, further analyses and code optimization are required in order to fully explore the efficiency and accuracy of the new methods. These include more detailed stability and convergence analyses; optimized strategies for adaptive mesh refinements and selection of the orders in time and space; better initial guesses for Newton iterations (especially for highly nonlinear systems such as Richards' equation); proper treatments of differential and algebraic components in an equation; and the optimal choice of different parameters.

BIBLIOGRAPHY

[1] See http://pitagora.dm.uniba.it/testset/.

[2] B. K. Alpert and V. Rokhlin. A fast algorithm for the evaluation of legendre expansions. *SIAM J. on Sci. and Stat. Computing*, 12:1:158–179, 1991.

[3] M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, Philadelphia, 1998.

[4] K. Atkinson. *An Introduction to Advanced Numerical Analysis*. John Wiley, 2nd edition edition, 1989.

[5] W. Auzinger, H. Hofstatter, W. Kreuzer, and E. Weinmuller. Modified defect correction algorithms for odes. part i: General theory. *Numer. Algorithms*, 36:135–156, 2004.

[6] W. Bao, S. Jin, and P. A. Markowich. On time-splitting spectral approximations for the schrodinger equation in the semiclassical regime. *Journal of Computational Physics*, 175(2):487–524, Jan. 2002.

[7] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. M. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. V. der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* Philadalphia: Society for Industrial and Applied Mathematics. Also available as postscript file on http://www.netlib.orgtemplatesTemplates.html, 1994.

[8] R. Barrio. On the a-stability of runge-kutta collocation methods based on orthogonal polynomials. *SIAM Journal on Numerical Analysis*, 36(4):1291–1303, may 1999.

[9] K. Böhmer and H. Stetter, editors. *Defect Correction Methods, Theory and Applications*. Springer-Verlag, Wein, New York, 1984. ., *Defect Correction Methods, Theory and Applications*, Springer-Verlag, Wien-, 1984.

[10] F. A. Bornemann. An adaptive multilevel approach to parabolic equations i. general theory and 1d implementation. *IMPACT of Computing in Science and Engineering*, 2(4):279–317, Dec. 1990.

[11] A. Bourlioux, A. T. Layton, and M. L. Minion. High-order multi-implicit spectral deferred correction methods for problems of reactive flow. *J. Comput. Phys.*, 189(2):651–675, 2003.

[12] J. Boyd. *Chebyshev and Fourier Spectral Methods*. Dover, New York, 2nd edition edition, 2001.

[13] K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*. SIAM, Philadelphia, PA, 1995.

[14] J. Butcher. *The Numerical Analysis of Ordinary Differential Equatinos: Runge-Kutta and General Linear Methods*. Wiley, New York, 1987.

[15] M. P. Calvo and C. Palencia. Avoiding the order reduction of runge-kutta methods for linear initial boundary value problems. *Math. Comput.*, 71:1529–1543, 2002.

[16] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods in Fluid Dynamics*. Springer-Verlag, 1988.

[17] K. Chen. *Preconditioning Techniques and Applications*. Number 19 in Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2005.

[18] H. Cheng, J. Huang, and T. J. Leiterman. An adaptive fast solver for the modified helmholtz equation in two dimensions. *J. Comput. Phys.*, 211(2):616–637, 2006.

[19] K. Dekker and J. G. Verwer. *Stability of Runge-Kutta methods for stiff nonlinear differential equations*. CWI Monographs, North-Holland, 1984.

[20] P. Deuflhard, J. Lang, and U. Nowak. Adaptive algorithms in dynamical process simulation. In H. Neunzert, editor, *Progress in Industrial Mathematics at ECMI 94*, pages 122–137. Wiley-Teubner, 1996.

[21] A. Dutt, L. Greengard, and V. Rokhlin. Spectral deferred correction methods for ordinary differential equations. *BIT*, 40:2:241–266, 2000.

[22] A. Dutt, M. Gu, and V. Rokhlin. Fast algorithms for polynomial interpolation, integration, and differentiation. *SIAM J. Numer. Anal.*, 33(5):1689–1711, 1996.

[23] F. Ethridge and L. Greengard. A new fast-multipole accelerated poisson solver in two dimensions. *SIAM J. Sci. Comput.*, 23(3):741–760, 2001.

[24] B. Fornberg. *A practical guide to pseudospectral methods*. Cambridge University Pres, Cambridge, 1998.

[25] C. W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1971.

[26] D. Gottlieb and S. A. Orszag. *Numerical analysis of spectral methods*. SIAM, Philadelphia, 1977.

[27] L. Greengard. Spectral integration and two-point boundary value problems. *SIAM Journal on Numerical Analysis*, 28(4):1071–1080, aug 1991.

[28] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73(2):325–348, 1987.

[29] L. Greengard and V. Rokhlin. A new version of the fast multipole method for the laplace equation in three dimensions. *Acta Numerica*, 6:229–269, 1997.

[30] W. Hackbusch. A sparse matrix arithmetic based on h-matrices. part i: introduction to h-matrices. *Computing*, 62(2):89–108, 1999.

[31] W. Hackbusch and B. N. Khoromskij. A sparse h -matrix arithmetic. part ii: application to multi-dimensional problems. *Computing*, 64(1):21–47, 2000.

[32] Hagstrom and R. Zhou. On the accuracy of spectral deferred correction of splitting methods for initial value problems. In preparation.

[33] E. Hairer, C. Lubich, and M. Roche. *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods*. Springer-Verlag, 1989.

[34] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer-Verlag, 2002.

[35] E. Hairer, Lubich, Christian, and Roche, Michel. *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods.* Springer-Verlag, 1989.

[36] E. Hairer, S. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I, Non-stiff Problems*. Springer-Verlag, 1991.

[37] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II.* Springer, 1996.

[38] A. C. Hansen and J. Strain. Convergence theory for spectral deferred correction. In preparation.

[39] A. C. Hansen and J. Strain. On the order of deferred correction. *BIT*. Submitted.

[40] D. J. Higham and L. N. Trefethen. Stiffness of ODEs. *BIT*, 33(2):285–303, 1993.

[41] D. Hillel. *Fundamentals of soil physics*. Academic Press, New York, NY, 1980.

[42] J. Huang, J. Jia, and M. Minion. Arbitrary order krylov deferred correction methods for differential algebraic equations. *J. Comput. Phys.*, In Press, Corrected Proof.

[43] J. Huang, J. Jia, and M. Minion. Accelerating the convergence of spectral deferred correction methods. *J. Comput. Phys.*, 214(2):633–656, 2006.

[44] J. Huang and M. Minion. Fast multipole method accelerated iterative schemes for variable coefficient elliptic equations. In preparation.

[45] C. T. Kelly. *Iterative Methods for Linear and Nonlinear Equations*. SIAM, 1995.

[46] C. T. Kelly. *Solving Nonlinear Equations with Newton's Method*. SIAM, 2003.

[47] P. Kolm, S. Jiang, and V. Rokhlin. Quadruple and octuple layer potentials in two dimensions. i. analytical apparatus. *Appl. Comput. Harmon. Anal*, 14(1):47–74, 2003.

[48] J. Lambert. *Numerical Methods for Ordinary Differential Equations*. Wiley, New York, 1991.

[49] J. Lang. Adaptive fem for reaction-diffusion equations. In *Proceedings of international centre for mathematical sciences on Grid adaptation in computational PDES : theory and applications*, pages 105–116, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[50] A. T. Layton and M. L. Minion. Conservative multi-implicit spectral deferred correction methods for reacting gas dynamics. *J. Comput. Phys.*, 194(2):697–715, 2004.

[51] A. T. Layton and M. L. Minion. Implications of the choice of quadrature nodes for picard integral deferred corrections methods for ordinary differential equations. *BIT Numerical*, V45(2):341–373, June 2005.

[52] J.-Y. Lee and L. Greengard. A fast adaptive numerical method for stiff two-point boundary value problems. *SIAM J. Sci. Comput.*, 18(2):403–429, 1997.

[53] B. Lindberg. Error estimation and iterative improvement for discretization algorithms. *BIT Numerical*, V20(4):486–500, Dec. 1980.

[54] G. D. Marsily. *Quantitative hydrogeology: groundwater hydrology for engineers*. Academic Press, San Diego, CA, 1986.

[55] P. G. Martinsson and V. Rokhlin. A fast direct solver for boundary integral equations in two dimensions. *J. Comput. Phys.*, 205(1):1–23, 2005.

[56] B. Mercier, editor. *An Introduction to the Numerical Analysis of Spectral Methods*, 1989.

[57] C. T. Miller, C. Abhishek, and M. W. Farthing. A spatially and temporally adaptive solution of Richards' equation. *Advances in Water Resources*, 29:525–545, Apr. 2006.

[58] M. L. Minion. Semi-implicit spectral deferred correction methods for ordinary differential equations. *Comm. Math. Sci.*, 1:471–500, 2003.

[59] M. L. Minion. Semi-implicit projection methods for incompressible flow based on spectral deferred corrections. *Appl. Numer. Math.*, 48(3-4):369–387, 2004.

[60] V. Pereyna. Iterated deferred corrections for nonlinear boundary value problems. *Numerische*, V11(2):111–125, Feb. 1968.

[61] M. Pernice and H. F. Walker. Nitsol: A newton iterative solver for nonlinear systems. *SIAM J. Sci. Comput.*, 19(1):302–318, 1998.

[62] L. R. Petzold. A description of dassl: A differential-algebraic system solver. Technical Report SAND82-8637, Sandia National Lab, 1982.

[63] A. Rangan. Deferred correction methods for low index differential algebraic equations. Preprint.

[64] A. Rangan. *Adaptive Solvers for Partial Differential and Differential-Algebraic Equations*. PhD thesis, University of California at Berkeley, 2003. Rangan, *Adaptive Solvers for Partial Differential and Differential-Algebraic Equations*, Ph.D. Thesis, University of California at Berkeley, 2003.

[65] E. Rothe. Zweidimensionale parabolische randwertaufgaben als grenzfall eindimensionaler randwertaufgaben. *Annalen*, V102(1):650–670, Dec. 1930.

[66] Y. Saad and M. H. Schultz. Gmres: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, 1986.

[67] J. M. Sanz-Serna, J. G. Verwer, and W. H. Hundsdorfer. Convergence and order reduction of runge-kutta schemes applied to evolutionary problems in partial differential equations. *Numer. Math.*, 50(4):405–418, 1987.

[68] W. Schiesser. *The numerical method of lines: integration of partial differential equations*. Academic Press, San Diego, 1991.

[69] R. D. Skeel. A theoretical framework for proving accuracy results for deferred corrections. *SIAM Journal on Numerical Analysis*, 19(1):171–196, 1982.

[70] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer, 1992.

[71] J. Strain. Fast spectrally-accurate solution of variable-coefficient elliptic problems. *Proceedings of the American Mathematical Society*, 122(3):843–850, nov 1994.

[72] J. Strain. Spectral methods for nonlinear parabolic systems. *J. Comput. Phys.*, 122(1):1–12, 1995.

[73] P. Swarztrauber and R. Sweet. Efficient fortran subprograms for the solution of elliptic partial differential equations. In *Proceedings of the SIGNUM meeting on Software for partial differential equations*, page 30, New York, NY, USA, 1975. ACM Press.

[74] P. N. Swarztrauber. The methods of cyclic reduction, fourier analysis and the facr algorithm for the discrete solution of poisson's equation on a rectangle. *J. Comput. Phys.*, 15:46–54, 1974.

[75] L. N. Trefethen. Is gauss quadrature better than clenshaw-curtis? submitted to SIAM Review.

[76] L. N. Trefethen. *Spectral methods in MatLab*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[77] L. N. Trefethen and M. R. Trummer. An instability phenomenon in spectral methods. *SIAM Journal on Numerical Analysis*, 24(5):1008–1023, oct 1987.

[78] P. K. Vijalapura, J. Strain, and S. Govindjee. Fractional step methods for index-1 differential-algebraic equations. *J. Comput. Phys.*, 203(1):305–320, 2005.

[79] L. Ying, G. Biros, D. Zorin, and H. Langston. A new parallel kernel-independent fast multipole method. In *SC '03: Proceedings of the 2003 ACM/IEEE conference on Supercomputing*, page 14, Washington, DC, USA, 2003. IEEE Computer Society.

[80] P. E. Zadunaisky. A method for the estimation of errors propagated in the numerical solution of a system of ordinary differential equations. In G. Kontopoulos, editor, *IAU Symp. 25: The Theory of Orbits in the Solar System and in Stellar Systems*, pages 281–+, 1966.

[81] P. E. Zadunaisky. On the estimation of errors propagated in the numerical integration of ordinary differential equations. *Numerische*, V27(1):21–39, Mar. 1976.