Mono-Implicit Runge-Kutta-Nyström Methods for Boundary Value Ordinary Differential Equations

P.H. Muir and M. Adams *

Abstract: The successful use of mono-implicit Runge-Kutta methods has been demonstrated by several researchers who have employed these methods in software packages for the numerical solution of boundary value ordinary differential equations. However, these methods are only applicable to first order systems of equations while many boundary value systems involve higher order equations. While it is straightforward to convert such systems to first order, several advantages, including substantial gains in efficiency, higher continuity of the approximate solution and lower storage requirements, are realized when the equations can be treated in their original higher order form. In this paper, we consider generalizations of mono-implicit Runge-Kutta methods, called mono-implicit Runge-Kutta-Nyström methods, suitable for systems of second order ordinary differential equations, having the general form, $y''(t) = f(t, y(t), y'(t))$, and derive optimal symmetric methods of orders two, four, and six. We also introduce continuous mono-implicit Runge-Kutta-Nyström methods which yield interpolants for the discrete methods. Numerical results are included to demonstrate the effectiveness of these methods. Savings of more than 70% are attained in some instances.

Subject Classification: 65L05, 65L10.

Keywords: Runge-Kutta-Nyström methods, boundary value ordinary differential equations, second order equations, efficiency.

1 Introduction

The well-known text, [3], on the numerical solution of boundary value ordinary differential equations (BVODEs) includes a collection of over twenty problems from a wide variety of application areas. About two-thirds of these problems

*Department of Mathematics and Computing Science, Saint Mary's University, Halifax, Nova Scotia, Canada, B3H 3C3. e-mail: muir@stmarys.ca, madams@cs.stmarys.ca. This work was supported by the Natural Sciences and Engineering Research Council of Canada.

involve differential equations of orders two, three, or four, and this is not unusual for problems of this class; many application areas involve models in which higher derivatives appear. In this paper we will consider *second order* systems of ordinary differential equations (ODEs) of the form

$$y''(t) = f(t, y(t), y'(t)), \tag{1}$$

where $t \in [a, b]$, $y : R \to R^n$, and $f : R \times R^n \times R^n \to R^n$. The general form of the boundary conditions we assume is

$$g(y(a), y'(a), y(b), y'(b)) = 0, \tag{2}$$

where $g : R^n \times R^n \times R^n \times R^n \to R^n$, although the specific form of the boundary conditions will not be central to this paper. Necessary and sufficient conditions for the existence and uniqueness of solutions to (1),(2) are discussed in [3].

For the numerical solution of *first order* systems of *initial value* ODEs,

$$y'(t) = f(t, y(t)), \quad y(a) = y_0, \tag{3}$$

a class of methods known as Runge-Kutta (RK) methods have been employed for more than one hundred years; see, e.g., [6] and references within. The application of RK schemes in the numerical solution of *first order* systems of BVODEs, for which the general form is,

$$y'(t) = f(t, y(t)), \quad g(y(a), y(b)) = 0, \tag{4}$$

was discussed over twenty-five years ago in [44]. That paper considered a subclass of the RK methods which are equivalent, for first order systems, to another class of methods known as collocation methods, and the early use of RK methods for BVODEs was largely through the collocation methods [3]. The collocation code, COLSYS [1], and its later modified version, COLNEW [4], are well known software packages of this type. Approximately twenty years ago, in [12], a fourth order method for the numerical solution of (4) was presented. This method, which is equivalent to the fourth order Lobatto collocation method, was from a subclass of the RK methods called mono-implicit Runge-Kutta (MIRK) methods [13]. Shortly afterwards, in [14], a symmetric subclass of the MIRK methods for the numerical solution of (4) was presented and specific methods of orders four, six, and eight were derived. This was followed by [21] and [28] which considered the full class of MIRK methods applied in the numerical solution of (4). The latter paper included a description of an experimental software package called ABVRKS, which employed modified Newton iterations, adaptive mesh refinement, and deferred correction using a family of embedded MIRK methods. These papers were followed by the sequence [9], [10], [15], [16], which described research leading to the development of a general purpose software package, originally called HAGRON and more recently called TWPBVP, for the numerical solution of (4). This package uses MIRK methods of orders four, six, and eight within a deferred correction algorithm, and employs a modified Newton iteration, adaptive mesh refinement, and global error estimation in the computation of a discrete solution approximation, $\{y_i\}_{i=0}^N$, with $y_i \approx y(t_i)$, where $\{t_i\}_{i=0}^N$ is a set of meshpoints partitioning $[a, b]$. More recently, the paper [22] described the software package MIRKDC, which employs MIRK methods within the context of a defect control algorithm for the numerical solution of (4). This package includes a modified Newton iteration, adaptive mesh refinement, and allows a choice of orders two, four, or six. It also provides a continuous solution approximation, $u(t) \approx y(t), t \in [a, b]$, based on the use of continuous MIRK (CMIRK) methods [35]. Through some new work in this area, (4) can now also be treated within the MATLAB [33] environment, using the *bvp4c* code [30], which employs a fourth order MIRK method and defect control, and provides a continuous solution approximation based on a Hermite-Birkhoff type interpolant. The survey [11] provides further discussion of the use of MIRK methods in the numerical solution of BVODEs.

Since RK methods are applicable only to systems of the form (4) it is natural that software packages based on MIRK methods restrict the input problem class to first order systems of BVODEs. However, as noted earlier, it is common for BVODE systems to arise in forms in which one or more of the equations involve higher derivatives. While it is straightforward to convert a higher order system to a first order system, this increases the number of equations and can lead to inefficiencies in the numerical treatment of the problem. The COLSYS/COLNEW packages, because of the flexibility of the collocation method, are able to treat systems of higher order equations directly. Section 5.6.2 of [3] provides a detailed discussion of collocation for such problems.

To our knowledge, relatively little has been done on the treatment of higher order BVODEs using generalizations of RK methods. In the recent papers [19] and [45], it is first assumed that the BVODE systems are converted to first order and then discretized using a numerical method. The approaches described in these papers involve the careful treatment of the resultant linear systems to take advantage of the sparseness arising from the conversion to first order form. For the restricted class of second order BVODEs, $y''(t) = f(t, y(t))$, the paper, [7], describes an approach which uses deferred correction and a generalization of MIRK methods to second order equations, extending the approach mentioned above in [16]. The generalization of the MIRK methods is described in [8]; these methods are two-step methods and are specifically derived for second order equations with the above special structure, $y''(t) = f(t, y(t))$.

The goal of the current paper is to consider a generalization of the MIRK methods to allow direct treatment of systems of second order BVODEs. It is related to some extent to [7] but will consider the more general problem class, (1). It is also related to a considerable body of work on the development of generalizations of RK methods for higher order initial value ODEs, the most well-known of which are the Runge-Kutta-Nyström (RKN) methods, and in particular to the papers, [41], [42], [17], and [40], which investigate MIRKN methods for second order initial value problems.

This paper is organized as follows. In section two, we will begin with a brief review of MIRK and CMIRK methods. We will then consider the mono-implicit Runge-Kutta-Nyström (MIRKN) methods, which are a generalization of MIRK methods to second order equations. This will be followed by a discussion of continuous MIRKN (CMIRKN) methods. This section will also discuss implementation details associated with the use of MIRKN and CMIRKN methods in the BVODE context, including an analysis of the operation counts associated with the key linear algebra computations, and a discussion of various possibilities for defect control. The third section will include a discussion of order conditions for MIRKN and CMIRKN methods. Stage order conditions will also be considered, as will a characterization of symmetry for MIRKN methods. Order and stage order barriers will also be discussed. In section four, specific symmetric, optimal, MIRKN methods of order of accuracy two, four, and six will be derived. In section five, we will derive CMIRKN methods which will yield high order interpolants for the discrete solutions associated with the methods of section four. In section six, we will provide numerical examples to examine the impact of the use of some of the new schemes within the MIRKDC code. Section seven gives our summary, conclusions, and some indications of future work.

The results show that, in general, significant improvements in efficiency are attained when second order BVODE systems are treated directly, using MIRKN methods. Savings in the linear algebra computations are shown to approach 50% and the overall savings in the solution of some of the test problems are more than 70%. Additional advantages such as a more convenient interface for the user (no requirement for conversion to first order system form), higher continuity of the approximate solution, and lower storage requirements are also realized by this approach.

The basic ideas inherent in the generalizations from RK methods to RKN methods, from MIRK methods to MIRKN methods, and to continuous methods, can be extended to methods for higher order ODEs of the form $y^{(m)} = f(t, y(t), y'(t), \ldots, y^{(m-1)}(t))$, and we are currently working on the extension of the results of this paper to these more general equations.

2 MIRK, MIRKN, and CMIRKN Methods

We begin by briefly reviewing MIRK methods since the MIRKN methods are defined in terms of a generalization of these methods.

2.1 Discrete and Continuous MIRK Methods

For the numerical solution of (3) the general form for an s-stage RK method is

$$y_i = y_{i-1} + h \sum_{r=1}^{s} b_r k_r, \tag{5}$$

where the step length is $h = t_i - t_{i-1}$, and where

$$k_r = f\left(t_{i-1} + c_r h, y_{i-1} + h \sum_{j=1}^{s} a_{rj} k_j\right), \tag{6}$$

and the coefficients b_r, a_{rj}, $r, j = 1, \ldots, s$, define a specific method, with $c_r = \sum_{j=1}^{s} a_{rj}$. The k_r's are defined implicitly in terms of each other and it is usually necessary to solve an $s \cdot n$ by $s \cdot n$ nonlinear system to determine them. The coefficients are often presented in the form of a tableau

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array}, \tag{7}$$

where $c = (c_1, \ldots, c_s)^T$, $b = (b_1, \ldots, b_s)^T$, and A is an s by s matrix whose r, jth component is a_{rj}. A RK method is of *order* p if for the *local* initial value problem,

$$y'(t) = f(t, y(t)), \quad y(t_{i-1}) = y_{i-1}, \tag{8}$$

the numerical solution, y_i, given by (5), satisfies $|y(t_i) - y_i| = O(h^{p+1})$ (i.e. has *local error* $O(h^{p+1})$). A RK method of order p is derived by requiring its coefficients to satisfy a set of equations called *order conditions*.

MIRK methods, a subclass of the RK methods, were originally proposed for use in the efficient numerical solution of (3). For an s-stage MIRK method the general form is

$$y_i = y_{i-1} + h \sum_{r=1}^{s} b_r k_r, \tag{9}$$

with

$$k_r = f\left(t_{i-1} + c_r h, (1 - v_r) y_{i-1} + v_r y_i + h \sum_{j=1}^{r-1} x_{rj} k_j\right). \tag{10}$$

A specific method is defined by its coefficients, $\{v_r\}_{r=1}^{s}$, $\{x_{rj}\}_{j=1,r=1}^{r-1,s}$, and $\{b_r\}_{r=1}^{s}$, with $c_r = v_r + \sum_{j=1}^{r-1} x_{rj}$. In the initial value ODE context, the mono-implicit stages, (10), are implicit only in y_i and can be obtained through the solution of an n by n nonlinear system. The BVODE context will be discussed later in this section. The coefficients of a MIRK scheme are usually presented in a tableau of the form,

$$\begin{array}{c|c|c} c & v & X \\ \hline & & b^T \end{array}, \tag{11}$$

where $v = (v_1, \ldots, v_s)^T$ and X is an s by s matrix whose r, jth component is x_{rj}. In [21] it is shown that the MIRK method with the standard tableau given above is equivalent to a RK method with tableau

$$\begin{array}{c|c} c & X + vb^T \\ \hline & b^T \end{array}. \tag{12}$$

A MIRK method of *order* p can be derived by requiring its coefficients to satisfy a set of order conditions, usually given for convenience in a MIRK form, [5].

It is often useful to augment the discrete solution approximation, y_i, obtained at t_i, the endpoint of the ith step, with a continuous solution approximation available over the entire step. The discrete solution yielded by a RK method can be augmented through the use of a continuous RK (CRK) method. A CRK method for the subinterval, $[t_{i-1}, t_i]$, is given by

$$u(t_{i-1} + \theta h) = y_{i-1} + h \sum_{r=1}^{s^*} b_r(\theta) k_r, \tag{13}$$

with the k_r's defined as in (6), $s^* \geq s$, and $0 \leq \theta \leq 1$. In addition to the coefficients which define its stages, the scheme is defined by the weight polynomials, $\{b_r(\theta)\}_{r=1}^{s^*}$. A CRK scheme is of order p if for the local problem, (8), with $u(t)$ as in (13), we have

$$max_{0 \leq \theta \leq 1} |y(t_{i-1} + \theta h) - u(t_{i-1} + \theta h)| = O(h^{p+1}).$$

A pth order CRK scheme can be derived by requiring the coefficients and weight polynomials to satisfy continuous versions of the RK order conditions. Continuous MIRK (CMIRK) schemes are defined analogously, having the general form, (13), but with stages as in (10).

It is also possible to obtain continuous solution approximations for RK methods using Hermite-Birkhoff interpolation in the context of a "boot-strapping" algorithm [20].

2.2 Discrete and Continuous MIRKN Methods

A well-known class of methods for the numerical solution of *second order* initial value ODEs,

$$y''(t) = f(t, y(t), y'(t)), \qquad y(a) = y_0, \quad y'(a) = y'_0, \tag{14}$$

where $y : R \to R^n$, $f : R \times R^n \times R^n \to R^n$, and $y_0, y'_0 \in R^n$, is the class of Runge-Kutta-Nyström (RKN) methods, see, e.g., [29]. An s-stage RKN method has the form,

$$y_i = y_{i-1} + hy'_{i-1} + h^2 \sum_{r=1}^{s} b_r k_r, \quad y'_i = y'_{i-1} + h \sum_{r=1}^{s} b'_r k_r, \tag{15}$$

where

$$k_r = f\left(t_{i-1} + c_r h, y_{i-1} + c_r h y'_{i-1} + h^2 \sum_{j=1}^{s} a_{rj} k_j, y'_{i-1} + h \sum_{j=1}^{s} a'_{rj} k_j\right), \tag{16}$$

and where the coefficients b_r, b'_r, a_{rj}, a'_{rj}, $r, j = 1, \ldots, s$, define a specific method and $c_r = \sum_{j=1}^{s} a'_{rj}$. Much of the research on RKN methods has focused on those designed to handle the second order ODE with the special form, $y''(t) = f(t, y(t))$, since treatment of this form of ODE by an RKN method can be substantially more efficient than the treatment of the equivalent first order system by an RK method; [29] suggests that this advantage is not evident when the more general ODE, $y''(t) = f(t, y(t), y'(t))$ is considered, and, in fact, relatively little work has been done on RKN methods for ODEs having the more general form. However in [24] and [25], it is demonstrated that RKN methods for the more general ODE can be more efficient than RK methods applied to the equivalent first order system. While the number of stages of the two methods must be the same, the extra free coefficients available to the RKN method allow a method with a smaller truncation error coefficient to be obtained (through a process that optimizes the choice of free coefficients to minimize some measure of the truncation error coefficients.) This allows the RKN method to meet the user tolerance while taking larger steps than the RK method, leading to a more efficient method overall. In addition, the use of RKN methods can yield storage savings of 50%. The paper [37] provides further evidence of the superiority of RKN methods for the general second order ODE.

The coefficients of an RKN method are often presented in a tableau of the form

$$\begin{array}{c|c|c} c & A & A' \\ \hline & b^T & b'^T \end{array},$$

where A' is an s by s matrix whose r, jth component is a'_{rj} and $b' = (b'_1, \ldots, b'_s)^T$. A subset of the RKN methods can be obtained from the RK methods; [29] begins its discussion of RKN methods by "generating" a RKN from a RK method. For a RK method with tableau (7), one can generate a RKN method with the tableau

$$\begin{array}{c|c|c} c & A^2 & A \\ \hline & b^T A & b^T \end{array}. \tag{17}$$

The paper [24] describes a different approach, attributed to Verner [43], for generating a RKN method from a RK method. This approach gives a RKN method with tableau

$$\begin{array}{c|c|c} c & \tilde{A} & A \\ \hline & \tilde{b}^T & b^T \end{array},$$

where $\tilde{A}$ is the s by s matrix whose r, jth component is $\tilde{a}_{rj} = (c_r - c_j)a_{rj}$ and $\tilde{b}$ is the vector whose rth component is $\tilde{b}_r = (1 - c_r)b_r$.

A RKN method is of order p if for the local problem,

$$y''(t) = f(t, y(t), y'(t)), \quad y(t_{i-1}) = y_{i-1}, \quad y'(t_{i-1}) = y'_{i-1}, \tag{18}$$

the numerical solution approximation, y_i, and numerical derivative approximation, y'_{i-1}, given by (15), satisfy $|y(t_i) - y_i| = O(h^{p+1})$ and $|y'(t_i) - y'_i| = O(h^{p+1})$. A RKN method of order p can be derived by requiring its coefficients to satisfy RKN order conditions (to be discussed in the next section).

It is natural to consider the notion of mono-implicitness for RKN schemes. The MIRKN methods, a subclass of the RKN methods, have the general form,

$$y_i = y_{i-1} + hy'_{i-1} + h^2 \sum_{r=1}^{s} b_r k_r, \quad y'_i = y'_{i-1} + h \sum_{r=1}^{s} b'_r k_r, \tag{19}$$

where

$$\begin{aligned} k_r = \; & f\Bigg(t_{i-1} + c_r h, \\ & (1 - v_r)y_{i-1} + v_r y_i + h\left((c_r - v_r - w_r)y'_{i-1} + w_r y'_i\right) + h^2 \sum_{j=1}^{r-1} x_{rj} k_j, \\ & (1 - v'_r)y'_{i-1} + v'_r y'_i + h \sum_{j=1}^{r-1} x'_{rj} k_j\Bigg), \end{aligned} \tag{20}$$

and where the coefficients b_r, b'_r, v_r, v'_r, x_{rj}, x'_{rj}, $r, j = 1, \ldots, s$, define a specific method and $c_r = v'_r + \sum_{j=1}^{s} x'_{rj}$. The tableau for a MIRKN method with the above coefficients is

$$\begin{array}{c|c|c|c|c|c} c & v & w & X & v' & X' \\ \hline & & & b^T & & b'^T \end{array}, \tag{21}$$

where $w = (w_1, \ldots, w_s)^T$, $v' = (v'_1, \ldots, v'_s)^T$ and X' is an s by s matrix whose r, jth component is x'_{rj}. By substituting for y_i and y'_i in (20) from (19) and rearranging terms, it can be seen the this MIRKN method is equivalent to the RKN method with tableau

$$\begin{array}{c|c|c} c & X + vb^T + wb'^T & X' + v'b'^T \\ \hline & b^T & b'^T \end{array}. \tag{22}$$

A pth order MIRKN method can be derived by requiring its coefficients to satisfy MIRKN forms of the RKN order conditions (to be discussed in the next section).

Subclasses of the MIRKN methods have recently been considered for the numerical solution of the *special* second order *initial* value ODE, $y''(t) = f(t, y(t))$. Families of fourth order methods with integer abscissa with a restricted form of (20) are considered in [41] and in [42]; in the latter paper the stage restriction is that either $v = \underline{0}$ or $w = \underline{0}$. The question of generating MIRKN methods (with either $v = \underline{0}$ or $w = \underline{0}$) from MIRK methods, using the first of the generation techniques mentioned above, is examined in [17]. In [40] a subclass of the MIRKN methods (with either $v = \underline{0}$ or $w = \underline{0}$), is studied.

The extension of discrete RKN methods to continuous RKN (CRKN) methods has been discussed in, e.g., [31], [26], [18]. Here we consider the extension of MIRKN methods to continuous MIRKN (CMIRKN) methods. A CMIRKN method gives a continuous solution approximation, $u(t)$, and a continuous derivative approximation, $\bar{u}(t)$, for $t \in [t_{i-1}, t_i]$, and has the form

$$u(t_{i-1}+\theta h) = y_{i-1}+\theta h y'_{i-1}+h^2 \sum_{r=1}^{s^*} b_r(\theta) k_r, \quad \bar{u}(t_{i-1}+\theta h) = y'_{i-1}+h \sum_{r=1}^{s^*} \bar{b}_r(\theta) k_r, \tag{23}$$

with the k_r's defined as in (20), $s^* \geq s$, and $0 \leq \theta \leq 1$. In addition to the coefficients which define its stages, the scheme is defined by the weight polynomials, $\{b_r(\theta)\}_{r=1}^{s^*}$, $\{\bar{b}_r(\theta)\}_{r=1}^{s^*}$. A CMIRKN scheme is of order p if for the local problem, (18), with $u(t)$ and $\bar{u}(t)$ as in (23), we have

$$max_{0 \leq \theta \leq 1} |y(t_{i-1} + \theta h) - u(t_{i-1} + \theta h)| = O(h^{p+1}),$$

and

$$max_{0 \leq \theta \leq 1} |y'(t_{i-1} + \theta h) - \bar{u}(t_{i-1} + \theta h)| = O(h^{p+1}).$$

A pth order CMIRKN scheme can be derived by requiring the coefficients and weight polynomials to satisfy continuous versions of the MIRKN order conditions (to be discussed in the next section).

2.3 Generation of MIRKN methods from MIRK methods

As mentioned earlier, in this paper, we shall derive MIRKN methods from a direct application of the order conditions. However, it is always possible to derive a MIRKN method by "generating" it from an existing MIRK method. A corresponding MIRKN method can be obtained using the generation scheme from [29], described earlier in this section. In this subsection we examine this generation scheme and show that the property of mono-implicitness is preserved.

We begin with the tableau of the standard MIRK scheme, (11) and recall that it is equivalent to a RK scheme having the tableau (12). We then apply the generation scheme of [29], which for the standard RK tableau, (7), yields the RKN tableau (17). Substitution for A with $X + vb^T$ in (17) gives an RKN scheme with tableau

$$\begin{array}{c|c|c} c & X(X+vb^T)+vb^T(X+vb^T) & X+vb^T \\ \hline & b^T(X+vb^T) & b^T \end{array}. \tag{24}$$

Letting $\bar{b}^T = b^T(X + vb^T)$ we can simplify (24) to get

$$\begin{array}{c|c|c} c & X^2+Xvb^T+v\bar{b}^T & X+vb^T \\ \hline & \bar{b}^T & b^T \end{array}. \tag{25}$$

We next recall that the a MIRKN method with the standard tableau, (21), is equivalent to an RKN scheme with the tableau, (22). Comparing (25) with the standard MIRKN tableau in RKN form, (22), we see that the generation algorithm leads to a MIRKN scheme provided we choose $w = Xv$. We also note that the matrix X^2 will be strictly lower triangular since the matrix X is strictly lower triangular, ensuring mono-implicitness of the resultant MIRKN scheme. In summary, we have the following result:

Theorem 2.1: Under the generation scheme of [29], (17), a MIRK method with tableau,

$$\begin{array}{c|c|c} c & v & X \\ \hline & & b^T, \end{array}$$

generates a MIRKN method with the tableau

$$\begin{array}{c|c|c|c|c|c} c & v & Xv & X^2 & v & X \\ \hline & & & b^T(X+vb^T) & & b^T \end{array}. \quad \square \tag{26}$$

2.4 Application of MIRKN methods to BVODEs

2.4.1 Newton System Computations

Since the efficiency improvements associated with MIRKN methods in this context are best described in contrast to the associated MIRK method computations, we begin with a brief discussion on the use of MIRK methods in the solution of BVODEs. The basic algorithm proceeds as follows. We assume that the problem interval $[a, b]$ is subdivided by a mesh $\{t_i\}_{i=0}^N$, with $a = t_0 < t_1 < \ldots < t_N = b$ and that an initial guess for the solution at each mesh point is available. A discrete numerical solution for the whole problem interval, $\{y_i\}_{i=0}^N$ with $y_i \approx y(t_i)$, is obtained by applying a modified Newton method to the nonlinear system of equations consisting of the boundary condition equations and n more equations per subinterval which for a MIRK scheme, on the ith subinterval, have the form,

$$y_i - y_{i-1} - h \sum_{r=1}^{s} b_r k_r = 0, \tag{27}$$

with the k_r's given by (10). Since an entire set of solution values, $\{y_i\}_{i=0}^{N}$, is available from the previous iteration, both y_{i-1} and y_i are available for the computation of the k_r's in (27) and the calculation is explicit; thus the solution of a nonlinear system is not required and the computational costs are lower; see [21].

We next consider modifications to the above algorithm that arise when MIRK methods are replaced by MIRKN methods for the solution of systems of second order BVODEs. The main differences are that we have discrete numerical solution *and* derivative values for the whole problem interval, $\{y_i, y_i'\}_{i=0}^{N}$, where $y_i \approx y(t_i)$, and $y_i' \approx y'(t_i)$, and that the nonlinear equations for the ith subinterval become

$$\begin{bmatrix} y_i - y_{i-1} - h y_{i-1}' - h^2 \sum_{r=1}^{s} b_r k_r \\ y_i' - y_{i-1}' - h \sum_{r=1}^{s} b_r' k_r \end{bmatrix} = 0, \tag{28}$$

with the k_r's given by (20). Since an entire set of approximate values, $\{y_i, y_i'\}_{i=0}^{N}$, is available from the previous iteration, y_{i-1}, y_{i-1}', y_i, and y_i' are available for the computation of the k_r's in (28) and the calculation is explicit, as before.

As mentioned earlier, one of the advantages of the direct treatment of second order equations is that the user is provided with the convenience of being able to describe the ODE system in its original second order form, rather than having to convert it to first order. Two other advantages are savings in storage (since the stage vectors, k_r, will be half as long) and in continuity of the approximate continuous solution ($u(t) \in C^2[a,b]$ whereas the first order system form leads to $u(t) \in C^1[a,b]$). A third advantage, in computational efficiency, arises from two sources. The first comes from the presence of extra free coefficients in the MIRKN formula which can lead to smaller truncation error coefficients, which in turn can lead to larger steps (subintervals) over the problem interval, yielding improved efficiency (computational costs are linear in N the number of subintervals). The second source of improved efficiency comes from the implementation details for the Newton matrix calculations associated with the MIRK and MIRKN methods, which we now consider.

The dominant cost within a BVODE code is the construction of the Newton matrices, which have an almost block diagonal structure. The ith subinterval contributes a left block, L_i and a right block, R_i, to the Newton matrix. When the discretization formula is a MIRK method, we have

$$L_i = -I - h\sum_{r=1}^{s} b_r \frac{\partial k_r}{\partial y_{i-1}}, \quad R_i = I - h\sum_{r=1}^{s} b_r \frac{\partial k_r}{\partial y_i}, \tag{29}$$

where

$$\frac{\partial k_r}{\partial y_{i-1}} = \left[\frac{\partial f}{\partial y}\right]_r \left((1-v_r)I + h\sum_{j=1}^{r-1} x_{rj}\frac{\partial k_j}{\partial y_{i-1}}\right), \tag{30}$$

and

$$\frac{\partial k_r}{\partial y_i} = \left[\frac{\partial f}{\partial y}\right]_r \left(v_r I + h\sum_{j=1}^{r-1} x_{rj}\frac{\partial k_j}{\partial y_i}\right), \tag{31}$$

for $r = 1, \ldots, s$. The matrix, $\left[\frac{\partial f}{\partial y}\right]_r$, the Jacobian of f, is evaluated at $t_{i-1}+c_r h$, $(1-v_r)y_{i-1} + v_r y_i + h\sum_{j=1}^{r-1} x_{rj}k_j$. For a system of q equations, all partial derivatives are q by q matrices, and the dominant costs are the matrix additions and multiplications. The leading terms in the total cost for constructing L_i and R_i are $q^2s^2 + 2q^3s$ additions and multiplications. We refer the reader to [21] for further details.

When the discretization formula is a MIRKN method, we have

$$L_i = \left[\begin{array}{c|c} -I - h^2\sum_{r=1}^{s} b_r \frac{\partial k_r}{\partial y_{i-1}} & -hI - h^2\sum_{r=1}^{s} b_r \frac{\partial k_r}{\partial y'_{i-1}} \\ \hline -h\sum_{r=1}^{s} b'_r \frac{\partial k_r}{\partial y_{i-1}} & -I - h\sum_{r=1}^{s} b'_r \frac{\partial k_r}{\partial y'_{i-1}} \end{array}\right], \tag{32}$$

and

$$R_i = \left[\begin{array}{c|c} I - h^2\sum_{r=1}^{s} b_r \frac{\partial k_r}{\partial y_i} & -h^2\sum_{r=1}^{s} b_r \frac{\partial k_r}{\partial y'_i} \\ \hline -h\sum_{r=1}^{s} b'_r \frac{\partial k_r}{\partial y_i} & I - h\sum_{r=1}^{s} b'_r \frac{\partial k_r}{\partial y'_i} \end{array}\right], \tag{33}$$

where

$$\frac{\partial k_r}{\partial y_{i-1}} = \left[\frac{\partial f}{\partial y}\right]_r \left((1-v_r)I + h^2\sum_{j=1}^{r-1} x_{rj}\frac{\partial k_j}{\partial y_{i-1}}\right) + \left[\frac{\partial f}{\partial y'}\right]_r \left(h\sum_{j=1}^{r-1} x'_{rj}\frac{\partial k_j}{\partial y_{i-1}}\right), \tag{34}$$

$$\frac{\partial k_r}{\partial y'_{i-1}} = \left[\frac{\partial f}{\partial y}\right]_r \left(h(c_r - v_r - w_r)I + h^2\sum_{j=1}^{r-1} x_{rj}\frac{\partial k_j}{\partial y'_{i-1}}\right) + \tag{35}$$

$$\left[\frac{\partial f}{\partial y'}\right]_r \left((1-v'_r)I + h\sum_{j=1}^{r-1} x'_{rj}\frac{\partial k_j}{\partial y'_{i-1}}\right), \tag{36}$$

$$\frac{\partial k_r}{\partial y_i} = \left[\frac{\partial f}{\partial y}\right]_r \left(v_r I + h^2\sum_{j=1}^{r-1} x_{rj}\frac{\partial k_j}{\partial y_i}\right) + \left[\frac{\partial f}{\partial y'}\right]_r \left(h\sum_{j=1}^{r-1} x'_{rj}\frac{\partial k_j}{\partial y_i}\right), \tag{37}$$

and

$$\frac{\partial k_r}{\partial y'_i} = \left[\frac{\partial f}{\partial y}\right]_r \left(hw_r I + h^2\sum_{j=1}^{r-1} x_{rj}\frac{\partial k_j}{\partial y'_i}\right) + \left[\frac{\partial f}{\partial y'}\right]_r \left(v'_r I + h\sum_{j=1}^{r-1} x'_{rj}\frac{\partial k_j}{\partial y'_i}\right), \tag{38}$$

for $r = 1, \ldots, s$. The matrices, $\left[\frac{\partial f}{\partial y}\right]_r$, $\left[\frac{\partial f}{\partial y'}\right]_r$, are evaluated at $t_{i-1} + c_r h$, $(1-v_r)y_{i-1} + v_r y_i + h((c_r - v_r - w_r)y'_{i-1} + w_r y'_i) + h^2\sum_{j=1}^{r-1} x_{rj}k_j$, $(1-v'_r)y'_{i-1} + v'_r y'_i + h\sum_{j=1}^{r-1} x'_{rj}k_j$. For a system of q second order equations, all partial derivatives are q by q matrices, and the dominant costs are the matrix additions and multiplications. The leading terms in the total cost for constructing L_i and R_i in this case are $4q^2s^2 + 8q^3s$ additions and multiplications. Assuming a system of n second order equations, we let $q = 2n$ for the MIRK method operation count and $q = n$ for the MIRKN method operation count; this gives $4n^2s^2 + 16n^3s$ and $4n^2s^2 + 8n^3s$, respectively. For a large system of equations ($n >> s$), we see that the expected computational savings, from this source, associated with the use of the MIRKN methods, approaches 50%. For example, with three second order equations ($n = 3$) and five stage MIRK and MIRKN methods ($s = 5$), the operation counts are 3060 and 1980, respectively, indicating a saving for the MIRKN methods, from this source, of 33%.

2.4.2 Defect Computations

For first order systems, $y'(t) = f(t, y)$, a main use of the CMIRK methods is in the defect control algorithm. The CMIRK method yields a continuous solution approximation, $u(t)$, and the defect is given by

$$\delta(t) = u'(t) - f(t, u(t)). \tag{39}$$

Since $u(t)$ is locally $O(h^{p+1})$ for a pth order CMIRK method, $u'(t)$ is locally $O(h^p)$, and with a Lipschitz assumption on f, we have a defect, (39), that is $O(h^p)$, locally and globally (since $u(t)$ and $u'(t)$ are both $O(h^p)$ globally).

For second order systems, $y''(t) = f(t, y(t), y'(t))$, the CMIRKN method yields a solution approximation, $u(t)$, and a derivative approximation, $\bar{u}(t)$. We can approximate $y'(t)$ with either $u'(t)$ or $\bar{u}(t)$ and $y''(t)$ with either $u''(t)$ or $\bar{u}'(t)$. This leads to the following candidates for the defect:

$$\delta_1(t) = u''(t) - f(t, u(t), u'(t)), \quad \delta_2(t) = \bar{u}'(t) - f(t, u(t), u'(t)), \tag{40}$$

$$\delta_3(t) = u''(t) - f(t, u(t), \bar{u}(t)), \quad \delta_4(t) = \bar{u}'(t) - f(t, u(t), \bar{u}(t)). \tag{41}$$

Since $u(t)$ is locally $O(h^{p+1})$, it follows that $u''(t)$ will be only $O(h^{p-1})$. The global error of the discrete endpoint approximations is $O(h^p)$; hence the local error associated with the $u''(t)$ approximation will dominate the global error, and the defects, $\delta_1(t)$ and $\delta_3(t)$ will be only $O(h^{p-1})$. On the other hand, since both $u(t)$ and $\bar{u}(t)$ are locally $O(h^{p+1})$, $u'(t)$ and $\bar{u}'(t)$ will be locally $O(h^p)$, consistent with the global error. Thus, employing a Lipschitz assumption on f, the defects, $\delta_2(t)$ and $\delta_4(t)$ will be $O(h^p)$, and thus are to be preferred to $\delta_1(t)$ and $\delta_3(t)$.

Unfortunately the choice of defects, $\delta_2(t)$ or $\delta_4(t)$, as the basis for a defect control algorithm, will imply that only three of the four quantities $u(t)$, $\bar{u}(t)$, $u'(t)$, and $\bar{u}'(t)$ will be controlled. This concern suggests two further candidates for the defect:

$$\delta_5(t) = \left[\begin{array}{c} u'(t) - \bar{u}(t) \\ \bar{u}'(t) - f(t, u(t), u'(t)) \end{array} \right] \quad \text{and} \quad \delta_6(t) = \left[\begin{array}{c} u'(t) - \bar{u}(t) \\ \bar{u}'(t) - f(t, u(t), \bar{u}(t)) \end{array} \right]. \tag{42}$$

These expressions generalize $\delta_2(t)$ and $\delta_4(t)$, respectively, and make a closer connection to the form of the defect for the associated first order system. Of these, $\delta_6(t)$ might be preferred since it employs the $O(h^{p+1})$ $\bar{u}(t)$ value rather than the less accurate $O(h^p)$ $u'(t)$ value for the $y'(t)$ argument in $f(t, y(t), y'(t))$. We thus employ $\delta_6(t)$ as the basis for the defect control algorithm for which numerical results are presented, later in this paper.

A criticism of the choice of $\delta_6(t)$ for the defect is that it is no longer as straightforward to explain to a user exactly what quantity is being controlled; $\delta_1(t)$ is actually a better choice in this regard, but as we have mentioned, it provides only an $O(h^{p-1})$ defect. We are currently conducting investigation of the idea of employing a locally $O(h^{p+2})$ continuous solution approximation, $\tilde{u}(t)$ (obtained through the use of an order $p+1$ CMIRKN method, or possibly through the use of a boot-strapping algorithm [20].) Then $\tilde{u}'(t)$ will be $O(h^{p+1})$ locally and $\tilde{u}''(t)$ will be $O(h^p)$ locally. Then

$$\delta_7(t) = \tilde{u}''(t) - f(t, \tilde{u}(t), \tilde{u}'(t)),$$

will be a $O(h^p)$ expression for the defect, which can be explained in the usual straightforward way, as for first order systems, in terms of the solution approximation, $\tilde{u}(t)$. We note that the computation of a locally $O(h^{p+2})$ continuous solution approximation represents a somewhat more expensive solution approximation than that described in this paper.

3 Order Conditions, Order Barriers, Symmetry

3.1 Order Conditions for RKN and MIRKN Methods

The order conditions for RKN methods involve a generalization of the theory for RK methods; see, e.g., [29]. A detailed list of RKN order conditions up to order six is given in [24]. Order conditions up to order four, for example, are

Order Conditions: RKN Form

Order	Order Conditions for y_i	Order Conditions for y_i'
1	–	$b'^T e = 1$,
2	$b^T e = \frac{1}{2}$,	$b'^T c = \frac{1}{2}$,
3	$b^T c = \frac{1}{6}$,	$b'^T c^2 = \frac{1}{3}$, $\quad b'^T A' c = \frac{1}{6}$,
4	$b^T c^2 = \frac{1}{12}$, $\quad b^T A' c = \frac{1}{24}$.	$b'^T c^3 = \frac{1}{4}$, $\quad b'^T A' c^2 = \frac{1}{12}$,
		$b'^T A'^2 c = \frac{1}{24}$, $\quad b'^T c A' c = \frac{1}{8}$,
		$b'^T A c = \frac{1}{24}$.

(43)

Note from (15) that the approximation for y_i is at least first order (local error $O(h^2)$); i.e., it is guaranteed to agree up to the $O(h)$ term with the Taylor series for the local solution since it explicitly includes the hy'_{i-1} term. Thus order conditions for the coefficients of the RKN method associated with the expression for y_i only become relevant for order 2 and higher.

MIRKN methods are a subset of the RKN methods and thus methods of a desired order can be derived by applying the RKN order conditions but it is usually more convenient to convert the order conditions so that they explicitly depend on the coefficients $\{c,v,w,X,b,v',X',b'\}$ of the MIRKN method rather than the coefficients $\{c,A,b,A',b'\}$ of the RKN method, as in (43). This is done by substituting, in the RKN order conditions, for A with $X+vb^T+wb'^T$ and for A' with $X'+v'b'^T$ and simplifying. We defer listing MIRKN order conditions to a later section.

3.2 Stage Order Conditions

In [27] it is shown that if a RK method is applied to a system of stiff differential equations *order reduction* can result. This means that a pth order method will behave as if it is of lower order, q or $q+1$, where q is the *stage order* (e.g. [6]) of the method, independent of the classical order of the method.

For RKN methods, there are two sets of stage order conditions,

$$A'c^{j-1} = \frac{c^j}{j} \quad \text{and} \quad Ac^{j-1} = \frac{c^{j+1}}{j(j+1)} \quad j = 1, 2, \dots \quad . \tag{44}$$

If a method has stage order q then the stage arguments, $\hat{y}_r = y_{i-1} + hy'_{i-1} + h^2 \sum_{j=1}^{s} a_{rj} k_j$ and $\hat{y}'_r = y'_{i-1} + h \sum_{j=1}^{s} a'_{rj} k_j$ are at least $O(h^{q+1})$ approximations to the local solution and derivative, respectively, evaluated at $t_{i-1} + c_r h$. As we saw above for the RKN order conditions, the stage order conditions are also applied differently for the y and y' approximations. The y argument to the rth stage, $y_i + c_r h y'_i + \sum_{j=1}^{s} b_j k_j$, is guaranteed to be at least a first order (local error $O(h^2)$) approximation to the local solution at $t_{i-1} + c_r h$ since it agrees with the Taylor series expansion of the local solution at $t_{i-1} + c_r h$ up to at least the $O(h)$ term for all choices of the coefficients. Thus the stage order conditions for y do not become relevant until stage order two is considered. Thus the stage order conditions for RKN methods up to stage order four are

Stage Order Conditions: RKN Form

Stage Order	Stage Order Conditions for $\hat{y}_r$	Stage Order Conditions for $\hat{y}'_r$
1	–	$A'e = c,$
2	$Ae = \frac{c^2}{2},$	$A'c = \frac{c^2}{2},$
3	$Ac = \frac{c^3}{6},$	$A'c^2 = \frac{c^3}{3},$
4	$Ac^2 = \frac{c^4}{12}.$	$A'c^3 = \frac{c^4}{4}.$

(45)

(As we shall see later, an application of stage order conditions in the order specified in the table above corresponds to the order in which the stage order conditions actually arise in the order conditions.) One can immediately obtain the stage order conditions for MIRKN methods by employing the substitutions of $X + vb^T + wb'^T$ for A and $X' + v'b'^T$ for A' and simplifying. Up to stage order four, they are

Stage Order Conditions: MIRKN Form

Stage Order	Stage Order Conditions for $\hat{y}_r$	Stage Order Conditions for $\hat{y}'_r$
1	–	$X'e + v' = c,$
2	$Xe + \frac{v}{2} + w = \frac{c^2}{2},$	$X'c + \frac{v'}{2} = \frac{c^2}{2},$
3	$Xc + \frac{v}{6} + \frac{w}{2} = \frac{c^3}{6},$	$X'c^2 + \frac{v'}{3} = \frac{c^3}{3},$
4	$Xc^2 + \frac{v}{12} + \frac{w}{3} = \frac{c^4}{12}.$	$X'c^3 + \frac{v'}{4} = \frac{c^4}{4},$

(46)

and the obvious general forms, for stage order $j + 2$, are

$$Xc^j + \frac{v}{(j+1)(j+2)} + \frac{w}{j+1} = \frac{c^{j+2}}{(j+1)(j+2)} \quad \text{and} \quad X'c^{j+1} + \frac{v'}{j+2} = \frac{c^{j+2}}{j+2}. \tag{47}$$

3.3 Alternative Form of the Order Conditions

In [2], an alternative form of the standard RK order conditions is presented. While this set of order conditions can be obtained from a linear combination of the standard order conditions, the alternative set is sometimes a more convenient representation. In the alternative form, one is able to express all the RK order conditions as either quadrature conditions, $b^T c^j = \frac{1}{j+1}$, or conditions which depend on stage order conditions. For RKN methods the same kind of alternative representation for the order conditions is possible; all order conditions are expressed either as quadrature conditions, $b^T c^j = \frac{1}{j(j+1)}$, $b'^T c^j = \frac{1}{j+1}$, or conditions involving the stage order conditions for an RKN method, (44). All of the RKN order conditions given in [24] can be written in this alternative form. For example, the order conditions up to fourth order for an RKN method, (43), can be rewritten as

Order Conditions: RKN Alt. Form

Order	Order Conditions for y	Order Conditions for y'
1	–	$b'^T e = 1,$
2	$b^T e = \frac{1}{2},$	$b'^T c = \frac{1}{2},$
3	$b^T c = \frac{1}{6},$	$b'^T c^2 = \frac{1}{3}, b'^T(A'c - \frac{c^2}{2}) = 0,$
4	$b^T c^2 = \frac{1}{12}, b^T(A'c - \frac{c^2}{2}) = 0.$	$b'^T c^3 = \frac{1}{4}, b'^T(A'c^2 - \frac{c^3}{3}) = 0,$
		$b'^T A'(A'c - \frac{c^2}{2}) = 0, b'^T c(A'c - \frac{c^2}{2}) = 0,$
		$b'^T(Ac - \frac{c^3}{6}) = 0.$

(48)

It is then clear from the above tables that by requiring a method to satisfy some of the stage order conditions, we can reduce the number of order conditions that must be satisfied. For example, an RKN method of stage order three, would have

$$A'e = c, A'c = \frac{c^2}{2}, A'c^2 = \frac{c^3}{3}, \quad \text{and} \quad Ae = \frac{c^2}{2}, Ac = \frac{c^3}{6}, \tag{49}$$

and then all of the order conditions for a fourth order RKN method except the quadrature conditions are satisfied.

Similarly, the order conditions for an MIRKN method can be expressed as either quadrature conditions, $b^T c^j = \frac{1}{j(j+1)}$ or $b'^T c^j = \frac{1}{j+1}$, or as conditions involving the stage order conditions (47). Up to order four, we have

Order Conditions for Orders 1 to 4: MIRKN Alt. Form

Order	Order Conditions for y_i	Order Conditions for y_i'
1	–	$b'^T e = 1,$
2	$b^T e = \frac{1}{2},$	$b'^T c = \frac{1}{2},$
3	$b^T c = \frac{1}{6},$	$b'^T c^2 = \frac{1}{3}, \quad b'^T(X'c + \frac{v'}{2} - \frac{c^2}{2}) = 0,$
4	$b^T c^2 = \frac{1}{12},$	$b'^T c^3 = \frac{1}{4}, \quad b'^T(X'c^2 + \frac{v'}{3} - \frac{c^3}{3}) = 0,$
	$b^T(X'c + \frac{v'}{2} - \frac{c^2}{2}) = 0.$	$b'^T X'(X'c + \frac{v'}{2} - \frac{c^2}{2}) = 0,$
		$b'^T c(X'c + \frac{v'}{2} - \frac{c^2}{2}) = 0,$
		$b'^T(Xc + \frac{v}{6} + \frac{w}{2} - \frac{c^3}{6}) = 0.$

(50)

In a later section of this paper, we will derive MIRKN and CMIRKN methods of orders five and six for which we will assume stage order three. The order conditions appearing above reduce to only the quadrature conditions; the remaining fifth and sixth order conditions are

Order Conditions for Orders 5 and 6, with Stage Order 3: MIRKN Alt. Form

Order	Order Conditions for y_i	Order Conditions for y_i'
5	$b^T c^3 = \frac{1}{20},$	$b'^T c^4 = \frac{1}{5}, b'^T(X'c^3 + \frac{v'}{4} - \frac{c^4}{4}) = 0,$ $b'^T(Xc^2 + \frac{v}{12} + \frac{w}{3} - \frac{c^4}{12}) = 0,$
6	$b^T c^4 = \frac{1}{30},$ $b^T(X'c^3 + \frac{v'}{4} - \frac{c^4}{4}) = 0,$	$b'^T c^5 = \frac{1}{6},$ $b'^T(X'c^4 + \frac{v'}{5} - \frac{c^5}{5}) = 0,$
	$b^T(Xc^2 + \frac{v}{12} + \frac{w}{3} - \frac{c^4}{12}) = 0.$	$b'^T(Xc^3 + \frac{v}{20} + \frac{w}{4} - \frac{c^5}{20}) = 0,$
		$b'^T c(X'c^3 + \frac{v'}{4} - \frac{c^4}{4}) = 0,$
		$b'^T c(Xc^2 + \frac{v}{12} + \frac{w}{3} - \frac{c^4}{12}) = 0,$
		$b'^T X'(X'c^3 + \frac{v'}{4} - \frac{c^4}{4}) = 0,$
		$b'^T X'(Xc^2 + \frac{v}{12} + \frac{w}{3} - \frac{c^4}{12}) = 0.$

(51)

3.4 Order Conditions for CMIRKN Methods

In [31] order conditions for CRKN methods are considered. Since the CMIRKN methods are a subclass of the CRKN methods, it is possible to obtain order conditions for CMIRKN methods from those for CRKN methods, using the substitutions for A with $X+vb^T+wb'^T$ and for A' with $X'+v'b'^T$. Furthermore these can be rewritten in terms of quadrature conditions and conditions which depend on stage order conditions, as is the case for discrete MIRKN methods. Given an order condition for a MIRKN method, one can obtain the corresponding order condition for a CMIRKN method simply by substituting for b with $b(\theta)$ and for b' with $\bar{b}(\theta)$, where $b(\theta) = [b_1(\theta), \ldots, b_{s^*}(\theta)]^T$ and $\bar{b}(\theta) = [\bar{b}_1(\theta), \ldots, \bar{b}_{s^*}(\theta)]^T$. If the order condition is the quadrature condition of order p, one must also introduce a factor of θ^p in the numerator of the right hand side. Specific continuous order conditions will be presented, as needed, in later sections of this paper.

3.5 Symmetry

In the numerical solution of BVODEs a key property of the underlying RK method is symmetry ([3], section 10.3.2). One characterization of symmetry, [39], is that the RK method is equal to its own reflection [38]. For MIRK methods it is straightforward to characterize symmetric schemes in terms of a restricted general form of the tableau [34]. Here we examine this question for MIRKN schemes.

For a given RK method, the reflected method is obtained by applying the RK method, starting at t_i and stepping *backwards* a distance of h to t_{i-1}. When the RK method is a MIRK method, the formula for the reflected method can be obtained from the original by substituting t_i for t_{i-1}, $-h$ for h, and swapping y_{i-1} and y_i. If the formula is invariant under these operations, it is symmetric. For a MIRKN method, the formula for the reflected method will be obtained by substituting t_i for t_{i-1}, $-h$ for h, swapping y_{i-1} and y_i, and swapping y'_{i-1} and y'_i. A comparison of the formulas for the original and reflected MIRKN methods will then lead to a characterization of symmetry in terms of the general form of the tableau for a MIRKN method, as follows. Recall that the general form for a MIRKN method is given by

$$y_i = y_{i-1} + hy'_{i-1} + h^2\sum_{r=1}^{s} b_r k_r, \quad y'_i = y'_{i-1} + h\sum_{r=1}^{s} b'_r k_r, \tag{52}$$

where

$$k_r = f\left(t_{i-1} + c_r h, \right.$$
$$\left.(1-v_r)y_{i-1} + v_r y_i + h\left((c_r - v_r - w_r)\, y'_{i-1} + w_r y'_i\right) + h^2\sum_{j=1}^{r-1} x_{rj} k_j, \right.$$

$$
(1-v'_r)y'_{i-1} + v'_r y'_i + h\sum_{j=1}^{r-1} x'_{rj}k_j \Bigg) . \tag{53}
$$

Performing the reflection substitutions and rearranging gives the formula for the reflected method,

$$
y_i = y_{i-1} + hy'_{i-1} + h^2 \sum_{r=1}^{s} (b'_r - b_r)k_r, \quad y'_i = y'_{i-1} + h\sum_{r=1}^{s} b'_r k_r, \tag{54}
$$

where

$$
\begin{aligned}
k_r = \quad & f\Bigg(t_{i-1} + (1-c_r)h, \\
& v_r y_{i-1} + (1-v_r)y_i + h\left((-w_r)y'_{i-1} + (w_r + v_r - c_r)y'_i\right) + h^2 \sum_{j=1}^{r-1} x_{rj}k_j, \\
& \qquad v'_r y'_{i-1} + (1-v'_r)y'_i + h\sum_{j=1}^{r-1}(-x'_{rj})k_j \Bigg) .
\end{aligned} \tag{55}
$$

In order for a MIRKN method to be symmetric, the original and reflected methods must be the same, up to a relabeling of the stages. That is, the MIRKN method is symmetric if there exists a permutation matrix P such that

$$
\begin{gathered}
c = P(e-c), \quad v = P(e-v), \quad w = P(w+v-c), \quad X = PXP^T, \quad b = P(b'-b), \\
v' = P(e-v'), \quad X' = P(-X')P^T, \quad b' = Pb'.
\end{gathered} \tag{56}
$$

Since X and X' must be strictly upper triangular, the symmetry conditions on these matrices are least restrictive if adjacent pairs of stages map onto each other under the reflection substitutions and swaps. In the case of a method with an odd number of stages, the last stage can map onto itself. This latter case introduces substantial restrictions on the coefficients, e.g. $c_r = \frac{1}{2}$, and in practical symmetric MIRKN methods only one such stage will be included. These symmetry conditions allow us to derive restrictions on the coefficients of the method.

Below we give the resultant general forms for the tableaus of symmetric MIRKN methods having $s = 1, \ldots, 5$.

$$
\begin{array}{c|c|c|c|c|c}
\frac{1}{2} & \frac{1}{2} & w_1 & 0 & \frac{1}{2} & 0 \\
\hline
 & & & \frac{1}{2}b'_1 & & b'_1
\end{array} \tag{57}
$$

$$
\begin{array}{c|c|c|cc|c|cc}
c_1 & v_1 & w_1 & 0 & 0 & v'_1 & 0 & 0 \\
1-c_1 & 1-v_1 & w_1+v_1-c_1 & 0 & 0 & 1-v'_1 & 0 & 0 \\
\hline
 & & & b_1 & b'_1 - b_1 & & b'_1 & b'_1
\end{array} \tag{58}
$$

$$
\begin{array}{c|c|c|ccc|c|ccc}
c_1 & v_1 & w_1 & 0 & 0 & 0 & v_1' & 0 & 0 & 0 \\
1-c_1 & 1-v_1 & w_1+v_1-c_1 & 0 & 0 & 0 & 1-v_1' & 0 & 0 & 0 \\
\frac{1}{2} & \frac{1}{2} & w_3 & x_{31} & x_{31} & 0 & \frac{1}{2} & x_{31}' & -x_{31}' & 0 \\
\hline
 & & & b_1 & b_1'-b_1 & \frac{1}{2}b_3' & & b_1' & b_1' & b_3'
\end{array}
\tag{59}
$$

$$
\begin{array}{c|c|c|ccccc}
c_1 & v_1 & w_1 & 0 & 0 & 0 & 0 & \ldots \\
1-c_1 & 1-v_1 & w_1+v_1-c_1 & 0 & 0 & 0 & 0 & \ldots \\
c_3 & v_3 & w_3 & x_{31} & x_{32} & 0 & 0 & \ldots \\
1-c_3 & 1-v_3 & w_3+v_3-c_3 & x_{32} & x_{31} & 0 & 0 & \ldots \\
\hline
 & & & b_1 & b_1'-b_1 & b_3 & b_3'-b_3 & \ldots
\end{array}
$$

$$
\begin{array}{c|c|cccc}
\ldots & v_1' & 0 & 0 & 0 & 0 \\
\ldots & 1-v_1' & 0 & 0 & 0 & 0 \\
\ldots & v_3' & x_{31}' & x_{32}' & 0 & 0 \\
\ldots & 1-v_3' & -x_{32}' & -x_{31}' & 0 & 0 \\
\hline
 & & b_1' & b_1' & b_3' & b_3'
\end{array}
\tag{60}
$$

$$
\begin{array}{c|c|c|cccccc}
c_1 & v_1 & w_1 & 0 & 0 & 0 & 0 & 0 & \ldots \\
1-c_1 & 1-v_1 & w_1+v_1-c_1 & 0 & 0 & 0 & 0 & 0 & \ldots \\
c_3 & v_3 & w_3 & x_{31} & x_{32} & 0 & 0 & 0 & \ldots \\
1-c_3 & 1-v_3 & w_3+v_3-c_3 & x_{32} & x_{31} & 0 & 0 & 0 & \ldots \\
\frac{1}{2} & \frac{1}{2} & w_5 & x_{51} & x_{51} & x_{53} & x_{53} & 0 & \ldots \\
\hline
 & & & b_1 & b_1'-b_1 & b_3 & b_3'-b_3 & \frac{1}{2}b_5' & \ldots
\end{array}
$$

$$
\begin{array}{c|c|ccccc}
\ldots & v_1' & 0 & 0 & 0 & 0 & 0 \\
\ldots & 1-v_1' & 0 & 0 & 0 & 0 & 0 \\
\ldots & v_3' & x_{31}' & x_{32}' & 0 & 0 & 0 \\
\ldots & 1-v_3' & -x_{32}' & -x_{31}' & 0 & 0 & 0 \\
\ldots & \frac{1}{2} & x_{51}' & -x_{51}' & x_{53}' & -x_{53}' & 0 \\
\hline
 & & b_1' & b_1' & b_3' & b_3' & b_5'
\end{array}
\tag{61}
$$

3.6 Order and Stage Order Barriers

The order and stage order barriers for MIRKN and CMIRKN methods follow from the corresponding results for MIRK and CMIRK methods.

Theorem 3.1: Assuming stage order conditions for MIRKN methods as given in (47), the maximum stage order of an s stage MIRKN method (19), (20), is $max(s,3)$.

Proof: We consider the stage order conditions from (47) involving b', and observe that, with X' replaced by X, v' replaced by v, and b' replaced by b, these conditions are identical to those for a MIRK method. The result then follows from Theorem 2.1 part (iii) of [5]. □

Theorem 3.2: The maximum order of an s-stage MIRKN method is $s+1$.

Proof: We consider the subset of the MIRKN order conditions up to order p involving b'. A subset of these conditions involve only the coefficients c, v', X', and b'. This subset of conditions, with the substitutions of X for X', v for v', and b for b', is identical to the set of order conditions for a pth order MIRK method. The result then follows from Theorem 3.3 of [5]. □

We note that for $s = 1, \ldots, 5$, it is possible to derive MIRKN methods meeting the barrier of Theorem 3.2; such methods are given later in this paper.

Theorem 3.3: The maximum order of an s-stage CMIRKN method is less than or equal to that of an s-stage CMIRK method.

Proof: The order conditions for a pth order continuous CMIRKN method include a subset which involve only the coefficients c, v', X', and $\bar{b}(\theta)$. With the substitutions of X for X', v for v', and $b(\theta)$ for $\bar{b}(\theta)$, these conditions become identical to the order conditions for a pth order CMIRK method. Also, as noted above, the stage order conditions for a CMIRKN method include a subset that are equivalent to those for a CMIRK method. Hence the order barriers for CMIRK methods given in [35] also apply to CMIRKN methods. □

We note that for $p = 1, \ldots, 6$, the bounds for CMIRK methods can also be met by CMIRKN methods; this will be demonstrated in a subsequent section of this paper.

4 Discrete, Symmetric MIRKN Methods

In this section we derive optimal, symmetric, MIRKN methods of orders two, four, and six. The general forms for the tableaus of the symmetric methods and the order and stage order conditions to be applied have been presented in the previous section. Some methods will have free parameters remaining after the order and stage order conditions have been satisfied; for a pth order method, we will select such parameters to minimize, C_{p+1}, a measure of the coefficients of the leading local error terms. (C_{p+1}^2 is a weighted sum of the (unsatisfied) order conditions for order $p+1$.) We will choose the free parameters to minimize C_{p+1}, subject to the constraint that C_{p+2}/C_{p+1} is not too large. See [25] for further details. We employ MAPLE [32] and the Fortran package RKTREE [36] to assist us in conducting this minimization process.

4.1 Second Order

We begin with the general family of one stage symmetric methods for which the tableau is identified in the previous section. With one stage it is possible to achieve order two but only stage order one. The resultant one parameter family has the tableau

$$\begin{array}{c|c|c|c|c|c} \frac{1}{2} & \frac{1}{2} & w_1 & 0 & \frac{1}{2} & 0 \\ \hline & & & \frac{1}{2} & & 1 \end{array}.$$

Analysis of C_3 gives a minimum for $w_1 = -\frac{1}{12}$. With this choice, $C_3 \approx 0.19$, $C_4 \approx 0.23$, and $C_4/C_3 \approx 1.2$.

We next consider the general family of two stage symmetric methods, beginning with the general form for the tableau of a two stage symmetric method given earlier. The maximum order and stage order are two; applying these conditions gives the two parameter family with the tableau

$$\begin{array}{c|c|c|cc|c|cc} 0 & v_1 & -\frac{v_1}{2} & 0 & 0 & 0 & 0 & 0 \\ 1 & 1-v_1 & \frac{v_1}{2} & 0 & 0 & 1 & 0 & 0 \\ \hline & & & b_1 & \frac{1}{2}-b_1 & & \frac{1}{2} & \frac{1}{2} \end{array}.$$

We analyze C_3 and find that it is minimized for $b_1 = \frac{1}{3}$ and that it does not depend on v_1; we will choose, arbitrarily, $v_1 = 0$. This method has $C_3 \approx 0.25$, $C_4 \approx 0.46$, and $C_4/C_3 \approx 1.8$.

4.2 Fourth Order

The minimum number of stages required to obtain a fourth order method is three. It is also possible to achieve the maximum stage order of three. Applying the stage order three conditions and the resultant fourth order conditions to the 3-stage symmetric tableau given earlier we obtain the one parameter family with tableau

$$\begin{array}{c|c|c|ccc|c|ccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & w_3 & -\frac{1}{16}-\frac{w_3}{2} & -\frac{1}{16}-\frac{w_3}{2} & 0 & \frac{1}{2} & \frac{1}{8} & -\frac{1}{8} & 0 \\ \hline & & & \frac{1}{6} & 0 & \frac{1}{3} & & \frac{1}{6} & \frac{1}{6} & \frac{2}{3} \end{array}.$$

Analysis of C_5 gives a minimum for $w_3 = -\frac{3}{20}$ (giving $x_{31} = x_{32} = \frac{1}{80}$). Then $C_5 \approx 0.022$, $C_6 \approx 0.042$, and $C_6/C_5 \approx 1.9$.

4.3 Sixth Order

The minimum number of stages required to obtain a sixth order method is five. It is also possible to achieve the maximum stage order of three. Applying the stage order three conditions and the resultant sixth order conditions to the 5-stage symmetric tableau given earlier we obtain a four parameter family with tableau

$$
\begin{array}{c|c|c|cccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \dots \\
c_3 & v_3 & w_3 & x_{31} & x_{32} & 0 & 0 & 0 & \dots \\
1-c_3 & 1-v_3 & w_3+v_3-c_3 & x_{32} & x_{31} & 0 & 0 & 0 & \dots \\
\frac{1}{2} & \frac{1}{2} & w_5 & x_{51} & x_{51} & x_{53} & x_{53} & 0 & \dots \\
\hline
 & & & b_1 & 0 & b_3 & b_4 & b_5 & \dots
\end{array}
$$

$$
\begin{array}{c|c|ccccc}
\dots & 0 & 0 & 0 & 0 & 0 & 0 \\
\dots & 1 & 0 & 0 & 0 & 0 & 0 \\
\dots & c_3^2(3-2c_3) & c_3(c_3-1)^2 & c_3^2(c_3-1) & 0 & 0 & 0 \\
\dots & (2c_3+1)(c_3-1) & -c_3^2(c_3-1) & -c_3(c_3-1)^2 & 0 & 0 & 0 \\
\dots & \frac{1}{2} & x'_{51} & -x'_{51} & x'_{53} & -x'_{53} & 0 \\
\hline
\dots & & b'_1 & b'_1 & b'_3 & b'_3 & b'_5
\end{array},
$$

where

$$x_{31} = -\frac{1}{3}v_3 - \frac{1}{2}w_3 - \frac{1}{6}{c_3}^3 + \frac{1}{2}{c_3}^2, \quad x_{32} = -\frac{1}{6}v_3 - \frac{1}{2}w_3 + \frac{1}{6}{c_3}^3,$$

$$x_{51} = \frac{1}{384}\frac{\alpha_1}{{c_3}^2\left(5\,{c_3}^2 - 5\,c_3 + 1\right)\left(-1 + c_3\right)^2},$$

$$x_{53} = -\frac{1}{384}\frac{\alpha_2}{{c_3}^2\left(5\,{c_3}^2 - 5\,c_3 + 1\right)\left(-1 + c_3\right)^2},$$

$$\alpha_1 = 2\,w_3 + 5\,c_3 + 216\,{c_3}^3 - 408\,{c_3}^4 - 54\,{c_3}^2 + v_3 - 120\,{c_3}^6 + 360\,{c_3}^5 - 384\,{c_3}^2 w_5 +$$
$$32\,c_3 w_5 + 1664\,{c_3}^3 w_5 - 3232\,{c_3}^4 w_5 - 960\,{c_3}^6 w_5 + 2880\,{c_3}^5 w_5,$$

$$\alpha_2 = 2\,w_3 + 5\,c_3 + 48\,{c_3}^3 - 24\,{c_3}^4 - 30\,{c_3}^2 + v_3 - 192\,{c_3}^2 w_5 +$$
$$32\,c_3 w_5 + 320\,{c_3}^3 w_5 - 160\,{c_3}^4 w_5,$$

$$x'_{51} = \frac{1}{64}\,\frac{40\,{c_3}^4 - 80\,{c_3}^3 + 52\,{c_3}^2 + 1 - 12\,c_3}{c_3\left(-1 + c_3\right)\left(5\,{c_3}^2 - 5\,c_3 + 1\right)},$$

$$x'_{53} = \frac{1}{64}\,\frac{2\,c_3 - 1}{c_3\left(-1 + c_3\right)\left(5\,{c_3}^2 - 5\,c_3 + 1\right)},$$

$$b_1 = \frac{1}{60}\,\frac{1 + 10\,{c_3}^2 - 10\,c_3}{c_3\left(-1 + c_3\right)}, \quad b_3 = \frac{1}{60}\,\frac{1}{\left(2\,c_3 - 1\right)^2 c_3},$$

$$b_4 = -\frac{1}{60}\,\frac{1}{\left(-1 + c_3\right)\left(2\,c_3 - 1\right)^2}, \quad b_5 = \frac{4}{15}\,\frac{5\,{c_3}^2 - 5\,c_3 + 1}{\left(2\,c_3 - 1\right)^2},$$

$$b_1' = \frac{1}{60}\,\frac{1+10\,{c_3}^2-10\,c_3}{c_3\,(-1+c_3)}, \quad b_3' = -\frac{1}{60}\,\frac{1}{c_3\,(-1+c_3)\,(2\,c_3-1)^2},$$

$$b_5' = \frac{8}{15}\,\frac{5\,{c_3}^2-5\,c_3+1}{(2\,c_3-1)^2}.$$

Analysis of C_7 gives a minimum for $c_3=\frac{1}{5}$, $v_3=\frac{1}{10}$, $w_3=-\frac{1}{50}$, and $w_5=-\frac{1}{5}$. Then $C_7\approx 0.0016$, $C_8\approx 0.0038$, and $C_8/C_7\approx 2.4$. These choices for the free parameters give the method with tableau

$$\begin{array}{c|c|c|ccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\frac{1}{5} & \frac{1}{10} & -\frac{1}{50} & -\frac{7}{1500} & -\frac{2}{375} & 0 & 0 & 0 & \cdots \\
\frac{4}{5} & \frac{9}{10} & -\frac{3}{25} & -\frac{2}{375} & -\frac{7}{1500} & 0 & 0 & 0 & \cdots \\
\frac{1}{2} & \frac{1}{2} & -\frac{1}{5} & \frac{2329}{61440} & \frac{2329}{61440} & -\frac{5}{12288} & -\frac{5}{12288} & 0 & \cdots \\
\hline
 & & & \frac{1}{16} & 0 & \frac{25}{108} & \frac{25}{432} & \frac{4}{27} & \cdots
\end{array}$$

$$\begin{array}{c|c|ccccc}
\cdots & 0 & 0 & 0 & 0 & 0 & 0 \\
\cdots & 1 & 0 & 0 & 0 & 0 & 0 \\
\cdots & \frac{13}{125} & \frac{16}{125} & -\frac{4}{125} & 0 & 0 & 0 \\
\cdots & \frac{112}{125} & \frac{4}{125} & -\frac{16}{125} & 0 & 0 & 0 \\
\cdots & \frac{1}{2} & -\frac{13}{256} & \frac{13}{256} & \frac{75}{256} & -\frac{75}{256} & 0 \\
\hline
\cdots & & \frac{1}{16} & \frac{1}{16} & \frac{125}{432} & \frac{125}{432} & \frac{8}{27}
\end{array}.$$

4.4 Comparison of Error Coefficients

In [29], as mentioned earlier, a given RK method generates an associated RKN method through the treatment of a second order ODE as a system of two first order equations. In this section we will compare each of the MIRKN methods we have derived with a "reference" MIRKN method, (26), generated from a MIRK method. The general approach has been described in an earlier section. The MIRK methods we will use for reference are the midpoint scheme, the trapezoidal scheme, and the fourth and sixth order MIRK methods employed in MIRKDC and given in [22].

In the table below, we compare the values for the error coefficients of our derived methods to those of the reference MIRKN methods.

Type	p	s	C_{p+1}	C_{p+2}
Derived	2	1	0.19	0.23
Reference	2	1	0.21	0.26
Derived	2	2	0.25	0.46
Reference	2	2	0.28	0.52
Derived	4	3	0.022	0.042
Reference	4	3	0.022	0.043
Derived	6	5	0.0016	0.0038
Reference	6	5	0.0017	0.0040

. (62)

The above table suggests that when we compare MIRK and MIRKN methods we should expect to see some differences in performance, associated with differences in accuracy, for second and sixth order, but no differences for fourth order.

5 CMIRKN Methods

In this section we will derive interpolants for the discrete solutions obtained from the computations based on the MIRKN methods presented in the previous section. These interpolants will be based on local polynomial interpolants for the solution and derivative for each subinterval. Over the whole problem domain, the piecewise polynomial interpolant for the solution will have C^2 continuity, while that for the derivative will have C^1 continuity. The framework of CMIRKN methods will be employed. Each CMIRKN method will include all the stages of the associated discrete MIRKN method; the discrete method is said to be *embedded* within the continuous one.

For a discrete method of order p, it is frequently sufficient to augment the discrete solution approximation with an interpolant of order $p-1$. The interpolant of order $p-1$ has a local error of $O(h^p)$ which agrees with the order of the *global* error associated with the discrete solution approximation. However, in the context of defect control (as in MIRKDC), it is necessary to employ an interpolant whose local error agrees with the *local* error of the discrete method. That is, both the discrete and continuous methods must be order p. See, e.g., [23], for further discussion. Therefore, for each discrete method of order p derived in the previous section, this section will include the derivation of CMIRKN methods of order $p-1$ and p.

In addition to satisfying the appropriate order conditions, the CMIRKN methods we derive must also satisfy certain continuity constraints, as indicated above. Because the degrees of the polynomials for each subinterval will be chosen to be as low as possible, the continuity constraints will sometimes be sufficient to ensure a desired order of accuracy. (That is, the order of accuracy following from standard interpolation theory will sometimes be sufficient to guarantee the desired order.)

We now present the continuity conditions that each CMIRKN method must satisfy. Assuming the general form given in (23) for the solution and derivative approximations, and assuming that

$$k_1 = f(t_{i-1}, y_{i-1}, y'_{i-1}) \quad \text{and} \quad k_2 = f(t_i, y_i, y'_i), \tag{63}$$

C^2 continuity for the continuous solution approximation, u, requires

$$\{b_r(0) = 0, b'_r(0) = 0\}_{r=1}^{s^*}, \quad b''_1(0) = 1, \{b''_r(0) = 0\}_{r=2}^{s^*},$$

and

$$\{b_r(1) = b_r, b'_r(1) = b'_r\}_{r=1}^{s}, \quad \{b_r(1) = 0, b'_r(1) = 0\}_{r=s+1}^{s^*},$$
$$b''_1(1) = 0, b''_2(1) = 1, \{b''_r(0) = 0\}_{r=3}^{s^*},$$

where $\{b_r\}_{r=1}^{s}$ and $\{b'_r\}_{r=1}^{s}$ are the weights for the solution approximation and derivative approximations, respectively, of the embedded discrete MIRKN method (19). C^1 continuity for the continuous derivative approximation, $\bar{u}$, requires

$$\{\bar{b}_r(0) = 0\}_{r=1}^{s^*}, \quad \bar{b}'_1(0) = 1, \{\bar{b}'_r(0) = 0\}_{r=2}^{s^*},$$

and

$$\{\bar{b}_r(1) = b'_r\}_{r=1}^{s}, \{\bar{b}_r(1) = 0\}_{r=s+1}^{s^*}, \quad \bar{b}'_1(1) = 0, \bar{b}'_2(1) = 1, \{\bar{b}'_r(0) = 0\}_{r=3}^{s^*}.$$

In some instances, the application of order and/or continuity conditions will lead to a family of CMIRKN methods with free parameters. As in the discrete case, it is possible to consider an error coefficient for the CMIRKN method that is similar to that for the MIRKN method except that it depends on θ; for the $O(h^{p+1})$ term, it is $C_{p+1}(\theta)$. We will choose the free parameters to minimize $C_{p+1} \equiv max_{0 \leq \theta \leq 1} C_{p+1}(\theta)$, subject to the constraint that C_{p+2}/C_{p+1} is not too large.

5.1 CMIRKN methods for 2nd Order MIRKN Methods

5.1.1 A CMIRKN method for the One Stage MIRKN Method

We will embed the one stage, second order, discrete MIRKN method of the previous section in a CMIRKN method with tableau,

$$\begin{array}{c|c|c|ccc|c|ccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\frac{1}{2} & \frac{1}{2} & -\frac{1}{12} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 \\
\hline
 & & & b_1(\theta) & b_2(\theta) & b_3(\theta) & & \bar{b}_1(\theta) & \bar{b}_2(\theta) & \bar{b}_3(\theta)
\end{array}.$$

The first two stages are the end point stages, (63); the third stage is the single stage from the discrete method. The actual cost is one extra stage per subinterval since end point stages can be shared between adjacent subintervals. (Since the endpoints stages are introduced for the CMIRKN method and are not included in the embedded discrete method, the description above for application of the continuity conditions needs to be modified slightly, in an obvious fashion.) The weight polynomials are determined by the continuity conditions; application of these conditions gives

$$b_1(\theta) = \frac{1}{2}\theta^2(1-\theta)^3, \quad b_2(\theta) = \frac{1}{2}\theta^3(1-\theta)^2, \quad b_3(\theta) = \frac{1}{2}\theta^3(2-\theta),$$
$$\bar{b}_1(\theta) = \theta(\theta-1)^2, \quad \bar{b}_2(\theta) = \theta^2(\theta-1), \quad \bar{b}_3(\theta) = \theta^2(3-2\theta).$$

These conditions force this CMIRKN method to be order two. It has $C_3 \approx 0.22$ and $C_4 \approx 0.32$, giving $C_4/C_3 \approx 1.5$.

5.1.2 A CMIRKN method for the Two Stage MIRKN Method

We next embed the two stage, second order, discrete method of the previous section in a CMIRKN method. Its tableau is

$$\begin{array}{c|c|c|cc|c|cc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
\hline
 & & & b_1(\theta) & b_2(\theta) & & \bar{b}_1(\theta) & \bar{b}_2(\theta)
\end{array}.$$

The weight polynomials are determined by the continuity conditions; we obtain

$$b_1(\theta) = \frac{1}{6}\theta^2(3-\theta), \quad b_2(\theta) = \frac{1}{6}\theta^3, \quad \bar{b}_1(\theta) = \frac{1}{2}\theta(2-\theta), \quad \bar{b}_2(\theta) = \frac{1}{2}\theta^2.$$

These conditions force this method to be order two. It has $C_3 \approx 0.25$ and $C_4 \approx 0.46$, giving $C_4/C_3 \approx 1.8$.

5.2 CMIRKN methods for the Fourth Order MIRKN Method

5.2.1 A Third Order CMIRKN method

We consider the three stage, fourth order, discrete MIRKN method of the previous section and begin by embedding it in a CMIRKN of third order (local error $O(h^4)$). The tableau of the resultant CMIRKN method is

$$\begin{array}{c|c|c|ccc|c|ccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\frac{1}{2} & \frac{1}{2} & -\frac{3}{20} & \frac{1}{80} & \frac{1}{80} & 0 & \frac{1}{2} & \frac{1}{8} & -\frac{1}{8} & 0 \\
\hline
 & & & b_1(\theta) & b_2(\theta) & b_3(\theta) & & \bar{b}_1(\theta) & \bar{b}_2(\theta) & \bar{b}_3(\theta)
\end{array}.$$

The continuity conditions are used to determine the weight polynomials; we obtain

$$b_1(\theta) = \frac{1}{6}\theta^2(\theta^2-3\theta+3), \quad b_2(\theta) = \frac{1}{6}\theta^3(\theta-1), \quad b_3(\theta) = \frac{1}{3}\theta^3(2-\theta),$$

$$\bar{b}_1(\theta) = \frac{1}{6}\theta(4\theta^2 - 9\theta + 6), \quad \bar{b}_2(\theta) = \frac{1}{6}\theta^2(4\theta - 3), \quad \bar{b}_3(\theta) = \frac{2}{3}\theta^2(3 - 2\theta).$$

The continuity conditions are sufficient to ensure that the method has third order. It has $C_4 \approx 0.024$ and $C_5 \approx 0.044$, giving $C_5/C_4 \approx 1.8$.

5.2.2 A Fourth Order CMIRKN method

We next embed the fourth order discrete method in a fourth order CMIRKN method (local error $O(h^5)$). In this case the order conditions for the determination of the derivative approximation are more restrictive than the continuity conditions and in fact require us to introduce an additional stage in order to satisfy them. This is the minimum number of stages for a fourth order CMIRKN method. The extra stage must also be required to satisfy the stage order three conditions. The free parameters are c_4, v_4, w_4, x_{41}, and v_4'. The weight polynomials for the derivative, $\{\bar{b}_r(\theta)\}$, determined from the order conditions, are expressible in terms of c_4. For the solution approximation, the continuity conditions are more restrictive and we use them to determine the corresponding weight polynomials; we obtain

$$b_1(\theta) = \frac{1}{6}\theta^2(\theta^2 - 3\theta + 3), \quad b_2(\theta) = \frac{1}{6}\theta^3(\theta - 1), \quad b_3(\theta) = \frac{1}{3}\theta^3(2 - \theta), \quad b_4(\theta) \equiv 0.$$

The extra stage is not required for the solution approximation and thus we have set the corresponding weight polynomial to zero. For a fourth order method with stage order three, the order conditions associated with the derivative approximation are just the quadrature conditions, $\bar{b}(\theta)^T c^j = \frac{\theta^{j+1}}{j+1}, j = 0, \ldots, 3$. Application of these conditions gives a five parameter family. An analysis of C_5 indicates that suitable choices for the free parameters are $c_4 = v_4 = \frac{3}{10}$, $w_4 = -1$, $x_{41} = \frac{4}{25}$, and $v_4' = \frac{2}{5}$. The tableau of the resultant CMIRKN method is

$$\begin{array}{c|c|c|cccc|c|cccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & \frac{1}{2} & -\frac{3}{20} & \frac{1}{80} & \frac{1}{80} & 0 & 0 & \frac{1}{2} & \frac{1}{8} & -\frac{1}{8} & 0 & 0 \\
\frac{3}{10} & \frac{3}{10} & -1 & \frac{4}{25} & \frac{87}{500} & \frac{561}{1000} & 0 & \frac{2}{5} & \frac{349}{3000} & -\frac{281}{3000} & -\frac{46}{375} & 0 \\
\hline
 & & & b_1(\theta) & b_2(\theta) & b_3(\theta) & 0 & & \bar{b}_1(\theta) & \bar{b}_2(\theta) & \bar{b}_3(\theta) & \bar{b}_4(\theta)
\end{array},$$

with $\{b_r(\theta)\}$ as above and with

$$\bar{b}_1(\theta) = -\frac{1}{6}\theta(10\theta^3 - 24\theta^2 + 19\theta - 6), \quad \bar{b}_2(\theta) = \frac{1}{42}\theta^2(30\theta^2 - 32\theta + 9),$$

$$\bar{b}_3(\theta) = -\frac{1}{3}\theta^2(15\theta^2 - 26\theta + 9), \quad \bar{b}_4(\theta) = \frac{125}{21}\theta^2(\theta - 1)^2.$$

This method has $C_5 \approx 0.022$, $C_6 \approx 0.042$, and $C_6/C_5 \approx 1.9$.

5.3 CMIRKN Methods for the Sixth Order MIRKN Method

5.3.1 A Fifth Order CMIRKN Method

For fifth order (local error $O(h^6)$), we will apply the order conditions involving the weight polynomials associated with the derivative, $\{\bar{b}_r(\theta)\}$, since they are more restrictive than the continuity conditions associated with those polynomials. For the weight polynomials associated with the solution, $\{b_r(\theta)\}$, the continuity conditions are more restrictive and we will therefore determine these weight polynomials using the continuity conditions. The application of the continuity conditions to the $\{b_r(\theta)\}$ polynomials gives

$$b_1(\theta) = -\frac{1}{16}\theta^2(5\theta^3 - 16\theta^2 + 18\theta - 8), \quad b_2(\theta) = \frac{1}{16}\theta^3(5\theta - 4)(\theta - 1),$$

$$b_3(\theta) = \frac{25}{432}\theta^3(9\theta^2 - 25\theta + 20), \quad b_4(\theta) = -\frac{25}{432}\theta^3(9\theta^2 - 20\theta + 10),$$

$$b_5(\theta) = \frac{4}{27}\theta^3(2 - \theta).$$

The derivation of the $\{\bar{b}_r(\theta)\}$ polynomials requires a closer examination of the order conditions. For fifth order, there are seven order conditions which would imply that the CMIRKN method must employ seven stages, i.e. two extra stages. However, it is possible to reduce the number of required stages to six, the minimum number of stages for a fifth order CMIRKN method.

We will require the new sixth stage to have stage order six and then examine the two fifth order conditions

$$\bar{b}(\theta)^T \left(X'c^3 + \frac{v'}{4} - \frac{c^4}{4}\right) = 0, \quad \bar{b}(\theta)^T \left(Xc^2 + \frac{v}{12} + \frac{w}{3} - \frac{c^4}{12}\right) = 0.$$

For the embedded sixth order discrete method obtained in the previous section, the vectors multiplying $\bar{b}(\theta)$ above are multiples of each other, and these two order conditions reduce to only one. This remaining condition together with the five quadrature conditions, $\bar{b}(\theta)^T c^j = \frac{\theta^{j+1}}{j+1}, j = 0, \ldots, 4$; can then be used to determine the $\{\bar{b}_r(\theta)\}$ polynomials. We get

$$\bar{b}_1(\theta) = \frac{1}{48}\theta(80\theta^4 - 275\theta^3 + 325\theta^2 - 202\theta + 48),$$

$$\bar{b}_2(\theta) = \frac{1}{16}\theta^2(4\theta - 3)(15\theta^2 - 20\theta + 6),$$

$$\bar{b}_3(\theta) = -\frac{125}{432}\theta^2(12\theta^3 - 39\theta^2 + 44\theta - 18), \bar{b}_4(\theta) = -\frac{125}{432}\theta^2(48\theta^3 - 111\theta^2 + 80\theta - 18),$$

$$\bar{b}_5(\theta) = -\frac{8}{27}\theta^2(30\theta^3 - 75\theta^2 + 62\theta - 18), \bar{b}_6(\theta) = \frac{125}{12}\theta^2(2\theta - 1)(\theta - 1)^2.$$

Application of the stage order six conditions to the new sixth stage leaves free parameters v_6, w_6, and v_6'. However, the error coefficients do not depend on these free parameters; we choose them arbitrarily to be $v_6 = v_6' = \frac{3}{5}$ and $w_6 = -\frac{3}{5}$. This CMIRKN method has $C_6 \approx 0.0023$, $C_7 \approx 0.0056$, and $C_7/C_6 \approx 2.5$. The tableau of this method is

$$
\begin{array}{c|c|c|cccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \;\ldots \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \;\ldots \\
\frac{1}{5} & \frac{1}{10} & -\frac{1}{50} & -\frac{7}{1500} & -\frac{2}{375} & 0 & 0 & 0 & 0 \;\ldots \\
\frac{4}{5} & \frac{9}{10} & -\frac{3}{25} & -\frac{2}{375} & -\frac{7}{1500} & 0 & 0 & 0 & 0 \;\ldots \\
\frac{3}{5} & \frac{3}{5} & -\frac{3}{5} & \frac{1827}{50000} & \frac{1857}{50000} & \frac{2767}{18000} & \frac{2509}{18000} & \frac{3184}{28125} & 0 \;\ldots \\
\hline
 & & & b_1(\theta) & b_2(\theta) & b_3(\theta) & b_4(\theta) & b_5(\theta) & 0 \;\ldots
\end{array}
$$

$$
\begin{array}{c|c|cccccc}
\ldots & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\ldots & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\ldots & \frac{13}{125} & \frac{16}{125} & -\frac{4}{125} & 0 & 0 & 0 & 0 \\
\ldots & \frac{112}{125} & \frac{4}{125} & -\frac{16}{125} & 0 & 0 & 0 & 0 \\
\ldots & \frac{3}{5} & \frac{117}{5000} & -\frac{183}{5000} & \frac{223}{1800} & -\frac{317}{1800} & \frac{368}{5625} & 0 \\
\hline
\ldots & & \bar{b}_1(\theta) & \bar{b}_2(\theta) & \bar{b}_3(\theta) & \bar{b}_4(\theta) & \bar{b}_5(\theta) & \bar{b}_6(\theta)
\end{array},
$$

with the weight polynomials as given above. The sixth stage is not required for the solution approximation and we have $b_6(\theta) \equiv 0$.

5.3.2 A Sixth Order CMIRKN Method

Here we derive a sixth order (local error $O(h^7)$) CMIRKN method within which we embed the discrete sixth order MIRKN method of the previous section. In this case, for both the $\{b_r(\theta)\}$ and $\{\bar{b}_r(\theta)\}$ polynomials, the order conditions involve more restrictions than do the continuity conditions, and we will thus proceed with an analysis of the order conditions.

The order conditions involving the $\{b_r(\theta)\}$ polynomials are the same seven conditions we considered in the fifth order case, except that all $\bar{b}(\theta)$ vectors are replaced by $b(\theta)$ vectors. As before, assuming that all new stages we add have at least stage order four, the two nonquadrature order conditions reduce to one. This condition, together with the remaining five, can be used to determine the $\{b_r(\theta)\}$ polynomials required for the solution approximation. We have

$$b_1(\theta) = \frac{1}{48}\theta^2(20\theta^4 - 75\theta^3 + 108\theta^2 - 74\theta + 24), \quad b_2(\theta) = \frac{1}{48}\theta^3(\theta - 1)(20\theta^2 - 25\theta + 8),$$

$$b_3(\theta) = -\frac{25}{432}\theta^3(20\theta^3 - 69\theta^2 + 85\theta - 40), \quad b_4(\theta) = -\frac{25}{432}\theta^3(20\theta^3 - 51\theta^2 + 40\theta - 10),$$

$$b_5(\theta) = -\frac{4}{27}\theta^3(8\theta^3 - 24\theta^2 + 25\theta - 10), \quad b_6(\theta) = \frac{8}{3}\theta^3(\theta - 1)^3.$$

There are fourteen order conditions involving the $\{\bar{b}_r(\theta)\}$ polynomials. We will require all new stages to have at least stage order five. Six of the resultant order conditions are

$$\bar{b}(\theta)^T\left(X'c^3+\frac{v'}{4}-\frac{c^4}{4}\right)=0,\quad \bar{b}(\theta)^T\left(Xc^2+\frac{v}{12}+\frac{w}{3}-\frac{c^4}{12}\right)=0,$$

$$\bar{b}(\theta)^T\left(X'c^4+\frac{v'}{5}-\frac{c^5}{5}\right)=0,\quad \bar{b}(\theta)^T\left(Xc^3+\frac{v}{20}+\frac{w}{4}-\frac{c^5}{20}\right)=0,$$

$$\bar{b}(\theta)^T c\left(X'c^3+\frac{v'}{4}-\frac{c^4}{4}\right)=0,\quad \bar{b}(\theta)^T c\left(Xc^2+\frac{v}{12}+\frac{w}{3}-\frac{c^4}{12}\right)=0.$$

An examination of the vectors multiplying $\bar{b}(\theta)$ in these equations shows that only two of the six are independent; we can thus reduce these six conditions to only

$$\bar{b}(\theta)^T\left(X'c^3+\frac{v'}{4}-\frac{c^4}{4}\right)=0,\quad \bar{b}(\theta)^T\left(X'c^4+\frac{v'}{5}-\frac{c^5}{5}\right)=0. \tag{64}$$

We next examine two other order conditions,

$$\bar{b}(\theta)^T X'\left(X'c^3+\frac{v'}{4}-\frac{c^4}{4}\right)=0,\quad \bar{b}(\theta)^T X'\left(Xc^2+\frac{v}{12}+\frac{w}{3}-\frac{c^4}{12}\right)=0.$$

An examination of the vectors multiplying $\bar{b}(\theta)^T X'$ in these conditions shows that they are multiples, and we can thus disregard the second one. The remaining condition, because of the sparsity structure of X' and because all but the third through fifth entries of $\left(X'c^3+\frac{v'}{4}-\frac{c^4}{4}\right)$ are zero, can be satisfied, independently of the $\{\bar{b}_r(\theta)\}$ polynomials, provided we impose the following extra constraint on the coefficients of the new stages

$$\frac{64}{125}\left(x'_{r3}+x'_{r4}\right)=x'_{r5},\quad r=6,\ldots,s^*. \tag{65}$$

Thus the first eight order conditions we have considered reduce to the two conditions, (64). In addition, we have, the six quadrature conditions, $\bar{b}(\theta)^T c^j = \frac{\theta^{j+1}}{j+1}, j=0,\ldots,5$; this implies that the CMIRKN method will require eight stages altogether and thus three extra stages in addition to the five of the embedded MIRKN method. This is the minimum number of stages that a sixth order CMIRKN method can have. In addition to satisfying (65), we will require the new stages to satisfy the stage order five conditions. Also, the sixth stage will be required to satisfy the first of the stage order six conditions, $X'c^5+\frac{v}{6}=\frac{c^6}{6}$, while stages seven and eight are required to satisfy both stage order six conditions. This leaves the free parameters v_6, w_6, v'_6, v_7, w_7, x_{75}, x'_{75}, c_8, v_8, w_8, x_{81}, x_{85}, and x'_{85}. A numerical investigation of C_7 gives optimal choices for these parameters of $v_6=\frac{9}{25}$, $w_6=-\frac{1}{4}$, $v'_6=\frac{12}{25}$, $v_7=\frac{4}{5}$, $w_7=0$,

$x_{75} = 0$, $x'_{75} = \frac{14}{25}$, $c_8 = \frac{4}{25}$, $v_8 = \frac{2}{5}$, $w_8 = \frac{9}{25}$, $x_{81} = 0$, $x_{85} = 0$, and $x'_{85} = \frac{1}{2}$. This gives $C_7 \approx 0.0016$, $C_8 \approx 0.038$, and $C_8/C_7 \approx 2.4$. The tableau of the CMIRKN method is

$$
\begin{array}{c|c|c|cccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \ldots \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \ldots \\
\frac{1}{5} & \frac{1}{10} & -\frac{1}{50} & -\frac{7}{1500} & -\frac{2}{375} & 0 & 0 & 0 & 0 & 0 & 0 & \ldots \\
\frac{4}{5} & \frac{9}{10} & -\frac{3}{25} & -\frac{2}{375} & -\frac{7}{1500} & 0 & 0 & 0 & 0 & 0 & 0 & \ldots \\
\frac{1}{2} & \frac{1}{2} & -\frac{1}{5} & \frac{2329}{61440} & \frac{2329}{61440} & -\frac{5}{12288} & -\frac{5}{12288} & 0 & 0 & 0 & 0 & \ldots \\
\frac{1}{2} & \frac{9}{25} & -\frac{1}{4} & \frac{911}{38400} & \frac{23}{1536} & \frac{1073}{13824} & \frac{737}{13824} & \frac{137}{5400} & 0 & 0 & 0 & \ldots \\
\frac{2}{5} & \frac{4}{5} & 0 & -\frac{317}{12500} & -\frac{3}{3125} & -\frac{428}{3375} & -\frac{581}{13500} & 0 & -\frac{10448}{84375} & 0 & 0 & \ldots \\
\frac{4}{25} & \frac{2}{5} & \frac{9}{25} & x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & 0 & \ldots \\
\hline
 & & & b_1(\theta) & b_2(\theta) & b_3(\theta) & b_4(\theta) & b_5(\theta) & b_6(\theta) & 0 & 0 & \ldots
\end{array}
$$

$$
\begin{array}{c|c|cccccccc}
\ldots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\ldots & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\ldots & \frac{13}{125} & \frac{16}{125} & -\frac{4}{125} & 0 & 0 & 0 & 0 & 0 & 0 \\
\ldots & \frac{112}{125} & \frac{4}{125} & -\frac{16}{125} & 0 & 0 & 0 & 0 & 0 & 0 \\
\ldots & \frac{1}{2} & -\frac{13}{256} & \frac{13}{256} & \frac{75}{256} & -\frac{75}{256} & 0 & 0 & 0 & 0 \\
\ldots & \frac{12}{25} & \frac{183}{6400} & -\frac{167}{6400} & \frac{1165}{6912} & -\frac{1085}{6912} & \frac{4}{675} & 0 & 0 & 0 \\
\ldots & -\frac{17497}{12500} & \frac{29817}{200000} & \frac{17817}{200000} & \frac{223}{320} & \frac{127}{320} & \frac{14}{25} & -\frac{288}{3125} & 0 & 0 \\
\ldots & v'_8 & x'_{81} & x'_{82} & x'_{83} & x'_{84} & x'_{85} & x'_{86} & x'_{87} & 0 \\
\hline
\ldots & & \bar{b}_1(\theta) & \bar{b}_2(\theta) & \bar{b}_3(\theta) & \bar{b}_4(\theta) & \bar{b}_5(\theta) & \bar{b}_6(\theta) & \bar{b}_7(\theta) & \bar{b}_8(\theta)
\end{array},
$$

where $v'_8 = -\frac{44493991}{31250000}$, the last row of X is

$$\left[\begin{array}{cccccccc} 0 & -\frac{7701259}{15625000} & -\frac{38497579}{93750000} & -\frac{745411}{46875000} & 0 & -\frac{7158056}{9765625} & \frac{10337749}{15625000} & 0 \end{array}\right],$$

the last row of X' is

$$\left[\begin{array}{cccccccc} \frac{76329639}{500000000} & \frac{45632679}{500000000} & \frac{11534329}{20000000} & \frac{7996921}{20000000} & \frac{1}{2} & \frac{56448}{1953125} & -\frac{12936}{78125} & 0 \end{array}\right],$$

and

$$\bar{b}_1(\theta) = -\frac{1}{176}\theta(1250\theta^5 - 4190\theta^4 + 5465\theta^3 - 3514\theta^2 + 1154\theta - 176),$$

$$\bar{b}_2(\theta) = \frac{1}{33264}\theta^2(107500\theta^4 - 239340\theta^3 + 191415\theta^2 - 65704\theta + 8208),$$

$$\bar{b}_3(\theta) = -\frac{125}{4752}\theta^2(125\theta^2 - 184\theta + 48)(2\theta - 1)^2,$$

$$\bar{b}_4(\theta) = -\frac{125}{4752}\theta^2(125\theta^2 - 184\theta + 48)(2\theta - 1)^2,$$

$$\bar{b}_5(\theta) = -\frac{8}{297}\theta^2(125\theta^2 - 184\theta + 48)(2\theta - 1)^2,$$

$$\bar{b}_6(\theta) = \frac{16}{187}\theta^2(\theta - 1)^2(875\theta^2 - 688\theta + 128),$$

$$\bar{b}_7(\theta) = -\frac{125}{1188}\theta^2(\theta - 1)^2(25\theta - 7)(25\theta - 18),$$

$$\bar{b}_8(\theta) = \frac{390625}{282744}\theta^2(\theta - 1)^2(25\theta^2 - 25\theta + 9).$$

Since the solution approximation does not need to use the seventh and eighth stages, we have $b_7(\theta) \equiv b_8(\theta) \equiv 0$.

6 Numerical Results

6.1 Solution Profiles

In this section we provide "solution profiles" for the code execution by MIRKDC using MIRK methods to treat first order systems and a modified version of MIRKDC, using MIRKN methods, to treat second order systems directly. The MIRKDC code (and all typical BVODE codes) solve a nonlinear BVODE system using the following general approach. For a given mesh of points which partition the problem interval, a numerical method, e.g. a Runge-Kutta method, is used to discretize the system of ODEs. The resultant equations, coupled with the boundary conditions, lead to a nonlinear system of algebraic equations, for which the unknowns are usually solution approximations at the meshpoints. The nonlinear system is then solved using a modified Newton iteration. This computation is usually coupled with an algorithm for estimating the global error, or, in the case of MIRKDC, an algorithm for estimating the defect of the solution. When the global error or defect estimate is not sufficiently small, its distribution over the subintervals of the mesh is used as the basis for generating a new mesh, and the algorithm is repeated, beginning with this new mesh.

We can provide an indication of the work undertaken by a BVODE code in solving a BVODE system by recording the corresponding solution profile; this is a sequence of ordered pairs of integers; the first integer gives the number of subintervals of the mesh and the second gives the number of Newton iterations performed on the nonlinear system associated with that mesh. We will consider several dozen test problems, characterized by parameter values, from two families of problems given in [3]. The families are nonlinear and do not have, in general, closed form solutions. In each family, the parameter, ϵ, controls the problem difficulty which increases as ϵ decreases.

The first family (P1) consists of one second order equation:

$$y''(t) = \left(\frac{\frac{1}{2} + \frac{\gamma}{2} - \epsilon A'(t)}{\epsilon A(t)}\right) y'(t) - \frac{y'(t)}{\epsilon A(t) y^2(t)}$$

$$-\frac{A'(t)}{\epsilon A^2(t)y(t)}\left(1-\frac{\gamma-1}{2}y^2(t)\right), \tag{66}$$

where $\gamma = 1.4$, $A(t) = 1 + t^2$, and where the boundary conditions are

$$y(0) = 0.9129, \quad y(1) = 0.375. \tag{67}$$

In our numerical testing, we also require the first order system form of this family:

$$\begin{aligned} u_1'(t) = u_2(t), \quad y''(t) &= \left(\frac{\frac{1}{2}+\frac{\gamma}{2}-\epsilon A'(t)}{\epsilon A(t)}\right) y'(t) - \frac{y'(t)}{\epsilon A(t)y^2(t)} \\ &-\frac{A'(t)}{\epsilon A^2(t)y(t)}\left(1-\frac{\gamma-1}{2}y^2(t)\right), \end{aligned} \tag{68}$$

with γ and $A(t)$ as before and with the boundary conditions

$$u_1(0) = 0.9129, \quad u_1(1) = 0.375. \tag{69}$$

For this family we will request a defect tolerance of 10^{-6}.

The second family (P2) consists of one fourth order and one second order ODE; the former we convert to two second order equations, obtaining the following second order BVODE system:

$$\begin{aligned} y_1''(x) = y_2(x), \quad \epsilon y_2''(x) &= -y_1(x)y_2'(x) - y_3(x)y_3'(x), \\ \epsilon y_3''(x) &= y_1'(x)y_3(x) - y_1(x)y_3'(x), \end{aligned} \tag{70}$$

with boundary conditions

$$y_1(0) = y_1'(0) = y_1(1) = y_1'(1) = 0, \qquad y_3(0) = -1, \quad y_3(1) = 1. \tag{71}$$

The first order system form of this family is:

$$\begin{gathered} u_1'(x) = u_2(x), \quad u_2'(x) = u_3(x), \quad y_3'(x) = u_4(x), \\ \epsilon u_4'(x) = -u_1(x)u_4(x) - u_5(x)u_6(x), \\ u_5'(x) = u_6(x), \quad \epsilon u_6'(x) = u_2(x)u_1(x) - u_1(x)u_6(x), \end{gathered} \tag{72}$$

with boundary conditions

$$u_1(0) = u_2(0) = u_1(1) = u_2(1) = 0, \qquad u_5(0) = -1, \quad u_5(1) = 1. \tag{73}$$

For this family we will request a defect tolerance of 10^{-5}.

The solution profile results obtained from applying the original MIRKDC code, using MIRK schemes to treat the first order systems, and from applying a modified version of the MIRKDC code, using MIRKN schemes to treat second order systems, are given in Tables 1, 2, and 3, for the family of test problems, P1, for methods of order 2,4, and 6. Tables 4, 5, and 6 provide the corresponding results for the second family of test problems, P2. In some cases a failed Newton iteration was encountered during the attempted solution of a nonlinear system associated with a given mesh (indicated by a *) but the code was able to recover and eventually obtain a solution on a finer mesh. Results for each family and each method are given for decreasing ϵ values until the code fails, with no recovery.

ϵ	Method	Profile
1.0	MIRK	(5,1),(20,1),(80,1),(247,1),(359,1)
1.0	MIRKN	(5,1),(20,1),(80,1),(242,1),(349,1)
0.1	MIRK	(5,2),(10,2),(20,2),(80,1),(320,1),(1280,1),(2051,1)
0.1	MIRKN	(5,2),(10,2),(20,2),(80,1),(320,1),(1220,1),(1903,1)

Table 1: Solution Profiles, Problem Family P1, Second Order

ϵ	Method	Profile
1.0	MIRK	(5,1),(15,1),(19,1)
1.0	MIRKN	(5,1),(15,1),(19,1)
0.1	MIRK	(5,2),(20,1),(57,1),(70,1)
0.1	MIRKN	(5,2),(20,1),(56,1),(69,1)
0.01	MIRK	(5,4),(10,9)*,(20,14)*,(40,18),(80,20),(160,19),(198,1),(222,1),(244,1)
0.01	MIRKN	(5,4)*,(10,8)*,(20,12)*,(40,19),(80,20),(178,1),(218,1)
0.008	MIRK	(5,6)*,(10,6)*,(20,13)*,(40,24),(80,24),(183,1),(230,1),(253,1)
0.008	MIRKN	(5,12)*,(10,12)*,(20,15)*,(40,16)*,(80,29),(183,1),(231,1),(254,1)
0.005	MIRK	(5,8)*,(10,9)*,(20,7)*,(40,20)*,(80,34)*,(160,40),(221,1),(264,1),(290,1)
0.005	MIRKN	(5,24)*,(10,9)*,(20,13)*,(40,19)*,(80,33)*,(160,40),(226,1),(267,1),(293,1)

Table 2: Solution Profiles, Problem Family P1, Fourth Order

An examination of the solution profile results indicates that, especially for the second and sixth order methods, the MIRKDC code generally converges more quickly to an appropriately adapted, coarser, mesh when MIRKN methods are employed for the direct treatment of second order systems. For fourth order we see relatively little difference between the methods. These results are consistent with those we obtained earlier in which the magnitudes of the local error coefficients for MIRK and MIRKN methods were assessed (section 4.4).

ϵ	Method	Profile
1.0	MIRK	(5,1),(8,1)
1.0	MIRKN	(5,1),(7,1)
0.1	MIRK	(5,2),(20,1),(29,1)
0.1	MIRKN	(5,2),(18,1),(23,1)
0.01	MIRK	(5,10)*,(10,14)*,(20,14)*,(40,19),(80,20),(107,1),(117,1)
0.01	MIRKN	(5,2)*,(10,13)*,(20,23),(40,19),(70,1),(79,1)
0.008	MIRK	(5,7)*,(10,19)*,(20,21)*,(40,26),(80,24),(117,1),(128,1)
0.008	MIRKN	(5,4)*,(10,14)*,(20,18)*,(40,21)*,(80,24),(88,1),(96,1)
0.005	MIRK	(5,11)*,(10,2)*,(20,11)*,(40,29)*,(80,41)*,(160,40),(176,1),(157,1)
0.005	MIRKN	(5,11)*,(10,15)*,(20,19)*,(40,25)*,(80,39),(92,1),(101,1),(111,1)

Table 3: Solution Profiles, Problem Family P1, Sixth Order

ϵ	Method	Profile
1.0	MIRK	(5,1), (20,1), (80,1), (181,1), (238,1)
1.0	MIRKN	(5,1), (20,1), (73,1), (110,1)
0.5	MIRK	(5,1),(20,1),(80,1),(221,1),(310,1)
0.5	MIRKN	(5,1),(20,1),(80,1),(143,1)
0.1	MIRK	(5,1),(10,1),(40,1),(160,1),(434,1),(604,1)
0.1	MIRKN	(5,1),(20,1),(80,1),(252,1),(368,1)
0.05	MIRK	(5,1),(10,1),(40,1),(160,1),(535,1),(798,1)
0.05	MIRKN	(5,1),(10,1),(40,1),(160,1),(435,1),(606,1)
0.01	MIRK	(5,1),(10,1),(20,1),(40,1),(80,1),(320,1),(1070,1),(1598,1)
0.01	MIRKN	(5,1),(10,1),(20,1),(40,1),(80,1),(320,1),(975,1),(1414,1)
0.005	MIRK	(5,3),(10,1),(20,1),(40,1),(80,1),(320,1),(1280,1),(2340,1), (2861,1)
0.005	MIRKN	(5,1),(10,1),(20,1),(40,1),(80,1),(320,1),(1280,1),(2144,1), (2626,1)

Table 4: Solution Profiles, Problem Family P2, Second Order

6.2 Newton Matrix Block Timings

In this section we report on the total execution times for the computation of the Newton matrices, as discussed in section 2.3.1. We compare the costs associated with applying MIRKDC, using MIRK methods, to the first order system version of test problems P1 and P2, with those associated with applying MIRKDC, using MIRKN methods, to the second order system version of each problem. Tables 7 and 8 give these results for problem families P1 and P2, respectively. All times are given in seconds.

The efficiency improvements associated with the linear algebra computations become more significant for larger systems of ODEs; for the small problem P1 (1 second order equation), we see from Table 7 that the savings for the total Newton matrix computations average about 20%. From Table 8 we see that for problem family P2 (3 second order equations) the savings are about 30%-70% for $p = 2$ and 25%-75% for $p = 6$, while for $p = 4$ the variance in code performance, as indicated by the solution profiles presented earlier, dominates the potential savings associated with the Newton matrix computations, and thus the execution times of both fourth order methods are comparable.

The reductions in the total Newton matrix computation costs associated with MIRKN methods arise from two sources: (i) savings in the computations of each pair of Newton blocks, L_i, R_i, for each subinterval, and (ii) savings associated with treating meshes having smaller numbers of subintervals (i.e. fewer blocks need be computed). In addition to the results for the total Newton matrix calculations, we are also interested in examining the relative costs for computing each L_i, R_i pair (i.e. cost (i)), with each of the classes of methods for orders 2,4, and 6. This cost will depend only on the number of ODEs, the number of stages of the method, and the type of method, MIRK or MIRKN (as discussed in section 2.3.1). Table 9 gives this execution time for each method, for each order, for each test problem. This data is obtained by computing for each test problem, the total number of Newton blocks computed (i.e. number of subintervals times number of Newton iterations, from the solution profile data) and then dividing the total Newton matrix time by this value.

The results of Table 9 show that the savings from the use of MIRKN methods for *one* Newton block pair, L_i, R_i, for problem family P1 (1 second order equation) are about 1%-6%, while for problem family P2 (3 second order equations) the savings are about 25%-28%. These results are consistent with the operation count analysis of section 2.3.1. Later in this section, we will return to this analysis for larger systems of BVODEs.

6.3 Overall Timings

Results from the previous subsection demonstrate that one source of improved efficiency from the use of MIRKN methods comes from a more efficient computation of the Newton matrix blocks. Another source of improvement in efficiency is implied by the results of subsection 6.1. Since more free parameters are available when MIRKN and CMIRKN methods are employed, it is usually possible to derive more accurate formulas and interpolants. The improved accuracy allows the MIRKDC code to employ coarser meshes (i.e. meshes with fewer subintervals) and still obtain a solution to the desired defect tolerance. In some instances there is also a reduction in the number of different meshes that the code needs to consider in order to obtain the solution to a given problem. (More accurate interpolants lead to sharper defect estimates which in turn lead to faster convergence to an appropriately adapted mesh.). Tables 1-6, given previously, exhibiting the solution profiles for various code executions, demonstrate these effects.

Since the costs of the calculations performed by MIRKDC (and by BVODE codes in general) depend significantly on the number of subintervals in each mesh and the number of different meshes that need to be treated, decreases in these quantities lead to significant decreases in the overall costs. In Tables 10 and 11 we report the overall execution times for MIRKDC using MIRK and MIRKN methods for order 2, 4, and 6, for the two test families.

Problem family P1, considered in Table 10, consists of only one second order equation and thus we do not expect there to be much advantage for MIRKN methods. The MIRK and MIRKN methods lead to comparable performance overall for orders 2 and 4, while for order 6, the MIRKN methods yield on average an efficiency improvement of about 20%. From Table 11, we observe that for problem family P2 and for second or sixth order methods, the savings associated with the MIRKN methods are about 30%-35%, while for fourth order no significant savings are attained (consistent with the results of section 4.4 which indicated that there is no significant difference between the fourth order methods.) We note that for the sixth order methods, the relative efficiency of the MIRKN methods improves as the problems become more difficult; we see savings of about 40%.

6.4 Results for Larger ODE Systems

In this section, we briefly examine the efficiency improvements available when the number of ODEs is somewhat larger. As indicated in section 2.3.1, the efficiency advantage of the MIRKN methods associated with the computation of the Newton matrices becomes more significant as the number of differential equations becomes larger.

We will consider the problem family (P3) which is based on the second problem family P2: P3 is constructed from four "copies" of P2. This leads to a test family representable as a system of twelve second order equations or twenty four first order equations. The solution profiles for the P3 family are the same as those for P2.

The total costs associated with the construction of the Newton matrices for test problem P3 are given in Table 12. The reductions in the times for the total Newton matrix computations associated with the use of the MIRKN methods are about 45%-70% for $p = 2$ and $p = 6$; for $p = 4$, the results indicate savings for the MIRKN method in the range of 40%-60% generally, but there a two problems, with $\epsilon = 10^{-5}$ and 10^{-6}, where the MIRK method has substantially better performance. This occurs because the Newton iteration associated with the calculation based on the MIRKN method for these problems has difficulty converging leading to more iterations and finer meshes. The problems are very poorly conditioned and even slight differences in the associated numerical computations can lead to significant differences overall. We also include, in Table 13, the costs per Newton matrix block associated with the problem P3 for MIRK and MIRKN methods. We observe that savings of about 43% are obtained.

We conclude by reporting the overall timings for the use of the MIRK and MIRKN methods in the solution of the first and second order system versions of problem P3. The results are provided in Table 14; the improvements associated with the use of the MIRKN methods are much more significant, in general. For $p = 2$, the overall savings are about 38%-42%, and for $p = 6$, the savings are about 40%-55%, while for $p = 4$, the performances of the two methods are comparable.

All tests were performed on a SUN UltraSparc 10 using the standard UNIX f77 compiler with full optimization.

7 Summary and Conclusions

This paper considers the direct treatment of second order systems of BVODEs by Runge-Kutta-Nyström methods. Based on the appropriate order and stage order conditions, and on symmetry conditions, optimal mono-implicit Runge-Kutta-Nyström methods of orders two, four, and six are derived. Continuous mono-implicit Runge-Kutta-Nyström methods are then introduced to provide continuous solution and derivative approximations; optimal schemes of orders two through six are derived. Order and stage order barrier results for the discrete and continuous schemes are provided. The specific structure of the Newton matrices arising from the use of the MIRKN methods is examined and it is shown that efficiency improvements, from this source, of up to 50% can be expected for larger ODE systems. Various forms for the appropriate defect for second order systems are also discussed.

A modified version of the MIRKDC software package has been developed to implement the new methods. The numerical results demonstrate efficiency improvements from two sources; the savings associated with the linear algebra computations, approaching 50%, are confirmed. The numerical results also demonstrate that a more significant source of improvement is associated with the optimality of the derived schemes; the extra free coefficients in the Nyström schemes allow more accurate schemes to be obtained and the resultant schemes lead to computations which employ coarser meshes and require fewer Newton iterations, particularly for higher order methods on difficult problems. In some instances, the combined efficiency improvements lead to overall savings of more than 70%.

There are a variety of areas for further investigation. Development of higher order discrete and continuous schemes would be useful. Further investigation of the various possibilities for the defect presented in this paper could be helpful. Generalizations of the methods and ideas presented here to higher order and mixed order systems appears to be an interesting area for further investigation. Application of the continuous methods to provide efficient superconvergent approximations for collocation solutions (see [23]) would also be interesting.

The analysis and results presented in this paper show that the use of classes of Runge-Kutta methods specifically adapted to the order of the ODEs can lead to significant improvements in computational efficiency as well as providing further convenience to the code user in providing the BVODE description, and through the higher continuity of the approximate solution and lower storage requirements. These results suggest that further investigation of these types of methods is warranted.

References

[1] U. Ascher, J. Christiansen, and R.D. Russell, Collocation software for boundary value ODEs, Trans. Math. Softw., 7, 1981, 209-222.

[2] The Runge-Kutta theory in a nutshell, SIAM J. Numer. Anal., 33, 1996, 1712-1735.

[3] U.M. Ascher, R.M.M. Mattheij, R.D. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, Classics in Applied Mathematics Series, Society for Industrial and Applied Mathematics, Philadelphia, 1995.

[4] G. Bader and U. Ascher, A new basis implementation for a mixed order boundary value ode solver, SIAM J. Sci. Stat. Comp., 8, 1987, 483-500.

[5] K. Burrage, F.H. Chipman, P.H. Muir, Order results for mono-implicit Runge-Kutta methods, SIAM J. Numer. Anal., 31, 1994, 876-891.

[6] J.C. Butcher, *The Numerical Analysis of Ordinary Differential Equations*, Wiley, Chichester, 1987.

[7] J.C. Butcher, J.R. Cash, G. Moore, and R.D. Russell, Defect correction for two-point boundary value problems on nonequidistant meshes, Math. Comp, 64, 1995, 629-648.

[8] J.R. Cash, High order P-stable formulae for the numerical integration of periodic initial value problems, Numer. Math., 37, 1981, 355-370.

[9] J.R. Cash, On the numerical integration of nonlinear two-point boundary value problems using iterated deferred corrections, Part 1: A survey and comparison of some one-step formulae, Comput. Math. Appl., 12a, 1986, 1029-1048.

[10] J.R. Cash, On the numerical integration of nonlinear two-point boundary value problems using iterated deferred corrections, Part 2: The development and analysis of highly stable deferred correction formulae, SIAM J. Numer. Anal., 25, 1988, 862-882.

[11] J.R. Cash, Runge-Kutta methods for the solution of stiff boundary value problems, Appl. Numer. Math., 22, 1996, 165-177.

[12] J.R. Cash and D.R. Moore, A high order method for the numerical solution of two-point boundary value problems, BIT, 20, 1980, 44-52.

[13] J.R. Cash and A. Singhal, Mono-implicit Runge-Kutta formulae for the numerical integration of stiff differential systems, IMA J. Numer. Anal., 2, 1982, 211-227.

[14] J.R. Cash and A. Singhal, High order methods for the numerical solution of two-point boundary value problems, BIT, 22, 1982, 184-199.

[15] J.R. Cash and M.H. Wright, Implementation issues in solving nonlinear equations for two-point boundary value problems, Computing, 45, 1990, 17-37.

[16] J.R. Cash and M.H. Wright, A deferred correction method for nonlinear two-point boundary value problems: implementation and numerical evaluation, SIAM J. Sci. Stat. Comput., 12, 1991, 971-989.

[17] H. De Meyer, G. Vanden Berghe, T. Van Hecke and M. Van Daele, On the generation of mono-implicit Runge-Kutta-Nyström methods by mono-implicit Runge-Kutta methods, J. Comp. Appl. Math., 87, 1997, 147-167.

[18] J.R. Dormand and P.J. Prince, Runge-Kutta-Nyström triples, Comput. Math. Applic., 13, 1987, 937-949.

[19] W.H. Enright and M. Hu, Improving performance when solving high-order and mixed-order boundary value problems in ODEs, Numer. Alg., 16, 1997, 107-116.

[20] W.H. Enright, K.R. Jackson, S.P. Nörsett, and P.G. Thomsen, Interpolants for Runge-Kutta formulas, ACM Trans. Math. Softw., 12, 1986, 193-218.

[21] W.H. Enright and P.H. Muir, Efficient classes of Runge-Kutta methods for two-point boundary value problems, Computing, 37, 1986, 315-334.

[22] W.H. Enright and P.H. Muir, Runge-Kutta software with defect control for boundary value ODEs, SIAM J. Sci. Comput., 17, 1996, 479-497.

[23] W.H. Enright and P.H. Muir, Superconvergent continuous Runge-Kutta schemes for discrete collocation solutions, SIAM J. Sci. Comp., 21, 1999, 227-254.

[24] Fine, J.M., Low order practical Runge-Kutta-Nyström methods with interpolants, Dept. of Comp. Sci. Tech. Rep. No. 183/85, University of Toronto, 1985.

[25] Fine, J.M., Low order Runge-Kutta-Nyström methods, Computing, 38, 1987, 281-297.

[26] Fine, J.M., Interpolants for Runge-Kutta-Nyström methods, Computing, 39, 1987, 27-42.

[27] R. Frank, J. Schneid, and C.W. Ueberhuber, Order results for implicit Runge-Kutta methods applied to stiff systems, SIAM J. Numer. Anal., 22, 1985, 515-534.

[28] S. Gupta, An adaptive boundary value Runge-Kutta solver for first order boundary value problems, SIAM J. Numer. Anal., 22, 1985, 114-126.

[29] Hairer, E., Nörsett, S. P., and Wanner, G., *Solving ordinary differential equations. I. Nonstiff problems*, Second edition, Springer Series in Computational Mathematics, 8, Springer-Verlag, Berlin, 1993.

[30] J. Kierzenka and L.F. Shampine, A BVP Solver based on Residual Control and the MATLAB PSE, SMU Math Report 99-001, Department of Mathematics, Southern Methodist University, Dallas, TX, USA, 1999.

[31] A. Marthinsen, Continuous extensions to Nyström methods for second order initial value problems, BIT, 36, 1996, 309-332.

[32] MAPLE, Waterloo Maple Inc., Waterloo, Ont., Canada.

[33] MATLAB, Mathworks, Inc., Natick, MA, USA.

[34] P.H. Muir, Optimal discrete and continuous mono-implicit Runge-Kutta schemes for BVODEs, Adv. Comp. Math., 10, 1999, 135-167.

[35] P.H. Muir and B. Owren, Order barriers and characterizations for continuous mono-implicit Runge-Kutta schemes, Math. Comp., 61, 1993, 675-699.

[36] P.W. Sharp, Fortran software for computation of error coefficients of Runge-Kutta and Runge-Kutta-Nyström methods, private communication, University of Auckland, Auckland, New Zealand, 1995.

[37] P.W. Sharp and Fine, J.M., Eight stage (5,6) Nyström pairs for y"=f(x,y,y'), Dept. of Comp. Sci. Tech. Rep. No. 215/88, University of Toronto, 1988.

[38] R. Scherer and H. Türke, Reflected and transposed methods, BIT, 23, 1983, 262-266.

[39] H. Stetter, *The Analysis of Discretization Methods for Ordinary Differential Equations*, Springer-Verlag, Berlin, 1973.

[40] M. Van Daele, H. De Meyer, T. Van Hecke and G. Vanden Berghe, On a class of P-stable mono-implicit Runge-Kutta-Nyström methods, Appl. Numer. Math., 27, 1998, 69-82.

[41] T. Van Hecke, M. Van Daele, G. Vanden Berghe, and H. De Meyer, A mono-implicit Runge-Kutta-Nyström modification of the Numerov method, J. Comp. Appl. Math., 78, 1997, 161-178.

[42] T. Van Hecke, M. Van Daele, G. Vanden Berghe, and H. De Meyer, P-stable mono-implicit Runge-Kutta-Nyström modifications of the Numerov method, in Numerical Analysis and Its Applications; First International Workshop, proceedings WNAA'96, Rousse, Bulgaria 1996; Ed L. Vulkov, J. Wasniewski and P. Yalamov, (Springer, Berlin); Lecture notes in Computer Science, 1196, 1997, 536–545.

[43] J.H. Verner, The order of some implicit Runge-Kutta methods, Numer. Math., 13, 14-23.

[44] R. Weiss, The application of implicit Runge-Kutta and collocation methods to boundary value problems, Math. Comp., 28, 1974, 449-464.

[45] K. Zuberi, Improving the efficiency of Runge-Kutta methods for the solution of BVPs for higher-order ODEs, M.Sc. Thesis, Computer Science Dept., Univ. of Toronto, 1996.

ϵ	Method	Profile
0.5	MIRK	(5,1),(9,1)
0.5	MIRKN	(5,1),(9,1)
0.1	MIRK	(5,1),(16,1),(21,1)
0.1	MIRKN	(5,1),(17,1),(22,1)
0.05	MIRK	(5,1),(20,1),(28,1)
0.05	MIRKN	(5,1),(20,1),(29,1)
0.01	MIRK	(5,1),(20,1),(39,1),(45,1)
0.01	MIRKN	(5,1),(20,1),(40,1),(45,1)
0.005	MIRK	(5,1),(10,1),(40,1),(66,1),(73,1)
0.005	MIRKN	(5,2),(10,1),(40,1),(61,1),(67,1)
0.001	MIRK	(5,5)*,(10,2),(20,2),(79,1),(107,1),(117,1)
0.001	MIRKN	(5,5)*,(10,2),(20,2),(77,1),(106,1),(116,1)
0.0005	MIRK	(5,4)*,(10,4),(20,3),(103,1),(130,1),(143,1)
0.0005	MIRKN	(5,2),(10,4),(20,3),(40,3),(102,1),(129,1),(141,1),(155,1)
0.0001	MIRK	(5,5),(10,9)*,(20,7)*,(40,5),(80,5),(151,1),(177,1),(194,1)
0.0001	MIRKN	(5,3),(10,5),(20,5),(40,5),(80,5),(151,1),(176,1),(193,1)
5×10^{-5}	MIRK	(5,4), (10,11)*, (20,15)*, (40,6), (80,7), (157,1), (195,1), (214,1)
5×10^{-5}	MIRKN	(5,7), (10,6)*, (20,4), (40,7), (80,7), (157,1), (187,1), (205,2)
10^{-5}	MIRK	(5,4), (10,6) , (20,17)* , (40,3)* , (80,13) , (160,12) , (320,12) , (279,1) , (228,1) , (456,12)
10^{-5}	MIRKN	(5,5) , (10,4)* , (20,9)* , (40,9) , (80,12) , (160,12) , (320,12) , (215,1) , (236,1)* , (472,19)* , (944,12)* , (1888,14)
5×10^{-6}	MIRK	(5,4) , (10,7) , (20,6)* , (40,7)* , (80,13) , (160,13) , (320,13) , (284,7)* , (568,14) , (308,1) , (243,1) , (486,6)* ,(972,17)* , (1944,13)
5×10^{-6}	MIRKN	(5,6) , (10,5)* , (20,15)* , (40,13)* , (80,14) , (160,13) , (320,13) , (212,3) , (233,16)* , (466,12)
10^{-6}	MIRK	(5,1)* , (10,3)* , (20,5)* , (40,4)* , (80,17)* , (160,13) , (320,13) , (640,13) , (403,1) , (305,8)* , (610,15)
10^{-6}	MIRKN	(5,2)* , (10,4) , (20,11)* , (40,9)* , (80,21) , (160,14) , (320,13) , (640,13) , (320,6)* , (640,15) , (320,2)* , (640,13)* , (1280,15)
5×10^{-7}	MIRK	(5,1)* , (10,4)* , (20,4)* , (40,4)* , (80,7)* , (160,7)* , (320,13) , (640,13) , (1280,13) , (666,1) , (365,5)* , (730,14)
5×10^{-7}	MIRKN	FAIL
10^{-7}	MIRK	(5,1)* , (10,1)* , (20,1)* , (40,1)* , (80,1)* , (160,2)* , (320,6)* , (640,13) , (1280,14) , (2560,13) , (1280,6)* , (2560,15) , (1280,1)
10^{-7}	MIRKN	(5,1)* , (10,1)* , (20,1)* , (40,2)* , (80,1)* , (160,5)* , (320,15) , (640,13) , (1280,15) , (2560,13) , (1280,1) , (640,1)

Table 5: Solution Profiles, Problem Family P2, Fourth Order

ϵ	Method	Profile
0.1	MIRK	(5,1),(9,1)
0.1	MIRKN	(5,1),(8,1)
0.05	MIRK	(5,1),(11,1),(13,1)
0.05	MIRKN	(5,1),(10,1)
0.01	MIRK	(5,1),(20,1),(25,1)
0.01	MIRKN	(5,1),(18,1),(22,1)
0.005	MIRK	(5,1),(10,1),(31,1),(36,1)
0.005	MIRKN	(5,1),(10,1),(26,1)
0.001	MIRK	(5,5),(10,3),(20,2),(48,1),(55,1)
0.001	MIRKN	(5,3),(10,4),(20,2),(38,1)
0.0005	MIRK	(5,7)*,(10,4),(20,3),(40,3),(62,1),(68,1)
0.0005	MIRKN	(5,4)*,(10,3),(20,3),(40,3),(52,1),(57,1),(51,1)
0.0001	MIRK	(5,4)*,(10,6)*,(20,6),(40,5),(80,5),(108,1),(97,1)
0.0001	MIRKN	(5,10),(10,8)*,(20,6),(40,5),(80,5),(88,1),(72,1)
5×10^{-5}	MIRK	(5,12)*,(10,5)*,(20,4)*,(40,7),(80,7),(160,7),(132,1),(105,1)
5×10^{-5}	MIRKN	(5,1)*,(10,7)*,(20,5)*,(40,7),(80,7),(160,7),(104,1),(78,1)
10^{-5}	MIRK	(5,9)*,(10,10)*,(20,2)*,(40,19),(80,14),(160,12),(320,12), (268,1),(156,2),(312,13),(222,1)
10^{-5}	MIRKN	(5,2)*,(10,2)*,(20,4)*,(40,4)*,(80,11)*,(160,13),(320,12), (160,1)
5×10^{-6}	MIRK	(5,7),(10,2)*,(20,7)*,(40,2)*,(80,18),(160,13),(320,13), (640,12),(471,1), (235,3)*,(470,18)
5×10^{-6}	MIRKN	(5,1)*,(10,5)*,(20,3)*,(40,7)*,(80,16),(160,13),(320,13), (160,1),(106,1)
10^{-6}	MIRK	(5,6),(10,1)*,(20,3)*,(40,3)*,(80,4)*,(160,21),(320,13), (640,13),(463,1), (231,3)*,(462,16)
10^{-6}	MIRKN	(5,1)*,(10,2)*,(20,2)*,(40,2)*,(80,17)*,(160,17),(320,13), (640,13),(320,1)
5×10^{-7}	MIRK	(5,6),(10,1)*,(20,1)*,(40,2)*,(80,3)*,(160,26)*,(320,14), (640,13),(1280,13), (857,1),(428,5)*,(856,15)
5×10^{-7}	MIRKN	(5,1)*,(10,1)*,(20,4)*,(40,1)*,(80,2)*,(160,5)*,(320,14), (640,13),(1280,13), (640,1),(320,1),(160,1)
10^{-7}	MIRK	(5,1)*,(10,1)*,(20,1)*,(40,1)*,(80,1)*,(160,3)*,(320,4)*, (640,14),(1280,13), (2560,13),(1597,1),(798,3),(1596,14)
10^{-7}	MIRKN	(5,1)*,(10,1)*,(20,1)*,(40,2)*,(80,1)*,(160,3)*,(320,2)*, (640,14),(1280,13), (2560,13),(1280,1),(640,1)

Table 6: Solution Profiles, Problem Family P2, Sixth Order

	MIRK Methods			MIRKN Methods		
ϵ	$p=2$	$p=4$	$p=6$	$p=2$	$p=4$	$p=6$
1.0	7.76×10^{-3}	1.00×10^{-3}	6.29×10^{-4}	7.32×10^{-3}	8.17×10^{-4}	6.74×10^{-4}
0.1	3.92×10^{-2}	3.06×10^{-3}	2.18×10^{-3}	3.49×10^{-2}	2.76×10^{-3}	1.93×10^{-3}
0.01	FAIL	1.04×10^{-1}	8.75×10^{-2}	FAIL	4.88×10^{-2}	4.37×10^{-2}
0.008	FAIL	6.41×10^{-2}	1.10×10^{-1}	FAIL	6.48×10^{-2}	9.54×10^{-2}
0.005	FAIL	1.71×10^{-1}	3.09×10^{-1}	FAIL	1.64×10^{-1}	1.31×10^{-1}

Table 7: Newton Matrix Timings, Problem Family P1

	MIRK Methods			MIRKN Methods		
ϵ	$p=2$	$p=4$	$p=6$	$p=2$	$p=4$	$p=6$
1.0	3.82×10^{-2}	1.59×10^{-3}	1.23×10^{-3}	1.20×10^{-2}	1.36×10^{-3}	9.68×10^{-4}
0.5	4.56×10^{-2}	1.71×10^{-3}	1.19×10^{-3}	1.36×10^{-2}	1.50×10^{-3}	9.41×10^{-4}
0.1	9.03×10^{-2}	4.93×10^{-3}	2.89×10^{-3}	5.40×10^{-2}	4.06×10^{-3}	2.29×10^{-3}
0.05	1.11×10^{-1}	6.05×10^{-3}	5.88×10^{-3}	6.74×10^{-2}	4.66×10^{-3}	2.38×10^{-3}
0.01	2.25×10^{-1}	1.20×10^{-2}	9.88×10^{-3}	1.62×10^{-1}	9.12×10^{-3}	6.56×10^{-3}
0.005	5.03×10^{-1}	2.11×10^{-2}	1.57×10^{-2}	3.46×10^{-1}	1.52×10^{-2}	6.19×10^{-3}
0.001	FAIL	6.77×10^{-2}	3.73×10^{-2}	FAIL	3.06×10^{-2}	1.85×10^{-2}
0.0005	FAIL	6.82×10^{-2}	7.18×10^{-2}	FAIL	5.72×10^{-2}	5.24×10^{-2}
0.0001	FAIL	1.46×10^{-1}	1.93×10^{-1}	FAIL	1.03×10^{-1}	1.41×10^{-1}
5×10^{-5}	FAIL	1.92×10^{-1}	4.52×10^{-1}	FAIL	1.45×10^{-1}	3.20×10^{-1}
10^{-5}	FAIL	1.72	2.30	FAIL	4.31	9.67×10^{-1}
5×10^{-6}	FAIL	6.68	4.61	FAIL	1.50	1.09
10^{-6}	FAIL	2.99	4.59	FAIL	4.56	2.28
5×10^{-7}	FAIL	4.69	9.15	FAIL	FAIL	4.33
10^{-7}	FAIL	12.1	16.09	FAIL	5.5	8.43

Table 8: Newton Matrix Timings, Problem Family P2

	MIRK Methods			MIRKN Methods		
	$p=2$	$p=4$	$p=6$	$p=2$	$p=4$	$p=6$
P1	1.0×10^{-5}	1.6×10^{-5}	2.7×10^{-5}	9.7×10^{-5}	1.5×10^{-5}	2.6×10^{-5}
P2	7.2×10^{-5}	10.6×10^{-5}	19.2×10^{-5}	5.3×10^{-5}	8.0×10^{-5}	14.0×10^{-5}

Table 9: Newton Block, L_i, R_i, Timings, Problems P1 and P2

ϵ	MIRK Methods			MIRKN Methods		
	$p=2$	$p=4$	$p=6$	$p=2$	$p=4$	$p=6$
1.0	3.13×10^{-2}	4.57×10^{-3}	3.48×10^{-3}	3.72×10^{-2}	5.24×10^{-3}	3.76×10^{-3}
0.1	1.60×10^{-1}	1.41×10^{-2}	8.86×10^{-3}	1.82×10^{-1}	1.38×10^{-2}	8.67×10^{-3}
0.01	FAIL	0.223	0.160	FAIL	0.112	9.35×10^{-2}
0.008	FAIL	0.136	0.193	FAIL	0.148	0.176
0.005	FAIL	0.325	0.508	FAIL	0.3448	0.240

Table 10: Overall Timings, Problem Family P1

ϵ	MIRK Methods			MIRKN Methods		
	$p=2$	$p=4$	$p=6$	$p=2$	$p=4$	$p=6$
1.0	7.63×10^{-2}	4.53×10^{-3}	3.19×10^{-3}	3.17×10^{-2}	4.44×10^{-3}	3.10×10^{-3}
0.5	9.20×10^{-2}	4.84×10^{-3}	3.25×10^{-3}	3.67×10^{-2}	4.78×10^{-3}	3.11×10^{-3}
0.1	0.184	1.06×10^{-2}	6.86×10^{-3}	0.102	1.05×10^{-2}	6.20×10^{-3}
0.05	0.229	1.28×10^{-2}	1.22×10^{-2}	0.175	1.23×10^{-2}	6.73×10^{-3}
0.01	0.487	2.43×10^{-2}	1.95×10^{-2}	0.427	2.48×10^{-2}	1.68×10^{-2}
0.005	1.09	4.47×10^{-2}	3.37×10^{-2}	0.988	4.65×10^{-2}	2.04×10^{-2}
0.001	FAIL	0.112	7.07×10^{-2}	FAIL	8.51×10^{-2}	4.88×10^{-2}
0.0005	FAIL	0.134	0.131	FAIL	0.157	0.134
0.0001	FAIL	0.289	0.337	FAIL	0.277	0.316
5.0×10^{-5}	FAIL	0.443	0.750	FAIL	0.347	0.662
10^{-5}	FAIL	2.79	3.76	FAIL	10.0	1.92
5.0×10^{-6}	FAIL	12.4	7.64	FAIL	3.06	2.12
10^{-6}	FAIL	5.82	7.45	FAIL	10.5	4.42
5.0×10^{-7}	FAIL	9.46	15.10	FAIL	FAIL	8.78
10^{-7}	FAIL	22.9	27.80	FAIL	13.6	17.86

Table 11: Overall Timings, Problem Family P2

	MIRK Methods			MIRKN Methods		
ϵ	$p=2$	$p=4$	$p=6$	$p=2$	$p=4$	$p=6$
0.1	2.91	1.39×10^{-1}	8.11×10^{-2}	9.19×10^{-1}	8.46×10^{-2}	4.36×10^{-2}
0.01	6.96	3.60×10^{-1}	2.89×10^{-1}	3.66	2.14×10^{-1}	1.50×10^{-1}
0.001	FAIL	1.29	1.14	FAIL	7.39×10^{-1}	4.40×10^{-1}
0.0001	FAIL	4.54	5.77	FAIL	2.43	3.33
10^{-5}	FAIL	44.0	72.8	FAIL	102.5	24.0
10^{-6}	FAIL	94.1	143.3	FAIL	110.3	59.5
10^{-7}	FAIL	395.5	515.7	FAIL	132.3	210.8

Table 12: Newton Matrix Timings, Problem Family P3

	MIRK Methods			MIRKN Methods		
	$p=2$	$p=4$	$p=6$	$p=2$	$p=4$	$p=6$
P3	2.2×10^{-3}	3.3×10^{-3}	5.7×10^{-3}	1.3×10^{-3}	1.9×10^{-3}	3.3×10^{-3}

Table 13: Newton Block, L_i, R_i, Timings

	MIRK Methods			MIRKN Methods		
ϵ	$p=2$	$p=4$	$p=6$	$p=2$	$p=4$	$p=6$
0.1	4.04	1.74×10^{-1}	1.01×10^{-1}	1.55	1.24×10^{-1}	5.92×10^{-2}
0.01	10.28	4.56×10^{-1}	3.67×10^{-1}	6.34	3.19×10^{-1}	2.01×10^{-1}
0.001	FAIL	2.11	1.39	FAIL	1.16	6.23×10^{-1}
0.0001	FAIL	6.26	7.35	FAIL	3.94	4.61
10^{-5}	FAIL	63.3	94.2	FAIL	165.3	33.1
10^{-6}	FAIL	133.8	181.7	FAIL	174.5	80.0
10^{-7}	FAIL	515.8	651.1	FAIL	214.4	290.2

Table 14: Overall Timings, Problem Family P3