A Numerical Study of Global Error Estimation Schemes for Defect Control BVODE Codes *

Jason J. Boisvert[†], Paul H. Muir[‡], Raymond J. Spiteri[§]

Abstract

A defect control boundary value ordinary differential equation (BVODE) code computes a numerical solution for which an estimate of the defect, i.e., the amount by which the approximate solution fails to satisfy the BVODE, is less than a user-provided tolerance. Defect control is attractive from a backward error viewpoint and the computation of the defect is inexpensive. A number of defect control codes are now in wide use, including the `MATLAB` codes, `bvp4c` and `bvp5c`, and the `Fortran` codes, `MIRKDC` and `BVP_SOLVER`. However, the global error of a numerical solution, i.e., the difference between the numerical solution and the exact solution, is often more familiar to users, and it can therefore be useful for a defect control code to also return an estimate of the global error for the defect controlled numerical solution it computes. The ratio of the global error to the maximum defect can also provide an indication of the conditioning of the BVODE. `BVP_SOLVER` currently provides an option for the return of a global error estimate based on Richardson extrapolation, and in this paper we consider the practical implementation within `BVP_SOLVER` of three alternative strategies based on the direct use of a higher order discretization formula, an estimate of the BVODE conditioning constant, and a deferred correction approach. We provide numerical results comparing the four estimators that show that the approaches based on the direct use of a higher order discretization formula and deferred correction are superior to the other two approaches.

Subject Classification: 65L06, 65L10.
Keywords: Boundary value ordinary differential equations, conditioning, defect control, deferred correction, global error estimation, Richardson extrapolation, Runge–Kutta methods

*This work was supported by the Mathematics of Information Technology and Complex Systems Network and by the Natural Sciences and Engineering Research Council of Canada.

[†]Department of Computer Science, University of Saskatchewan, Saskatoon SK, Canada, S7N 5C9, jjb701@mail.usask.ca

[‡]Department of Mathematics and Computing Science, Saint Mary's University, Halifax, NS, Canada, B3H 3C3, muir@smu.ca

[§]Department of Computer Science, University of Saskatchewan, Saskatoon SK, Canada, S7N 5C9, spiteri@cs.usask.ca

1 Introduction

Codes for boundary value ordinary differential equations (BVODEs) return an approximate numerical solution to the BVODE. Accordingly, the accuracy of the solution provided by the code must be assessed in some way. There are two common approaches to controlling solution accuracy in BVODE software: global error control and defect control. The global error is the difference between the approximate solution and the exact solution. The defect of a continuous numerical solution is the amount by which that solution fails to satisfy the BVODEs. High quality codes compute an estimate of the accuracy of the solution and attempt to adapt the computation so that the estimate is less than a user-provided tolerance.

Computational approaches for solving BVODEs are commonly divided into two categories: initial-value methods and global methods. We now briefly review these categories.

1.1 Initial-Value Methods

Simple shooting and multiple shooting methods are the primary initial-value methods for solving BVODEs; see, e.g., [2]. These methods rely directly on techniques for numerically solving initial-value problems (IVPs) for ODEs. In simple shooting, guesses are made for the unknown data at one boundary and, using IVP techniques, the resulting IVP is integrated to the other end of the problem domain, where the numerical solution is compared with the known boundary conditions. If these conditions are not satisfied, the initial guesses are then corrected according to an iteration scheme such as Newton's method, and the process repeats.

Unfortunately, simple shooting is known to be unstable [2]; therefore most initial-value BVODE codes implement some form of multiple shooting. Multiple shooting algorithms divide the problem domain into subintervals. Using approximate initial conditions, an IVP solver is then applied to solve a local IVP on each subinterval. Matching conditions are imposed on the numerical solutions from each subinterval, and, together with the boundary conditions, these are solved in an iterative fashion to obtain a continuous numerical solution over the problem domain. The number and locations of the mesh points are also chosen adaptively to improve the reliability and robustness of the computations performed by the IVP solver. For problems that include both exponentially increasing and exponentially decreasing solution components, it is well-known (see, e.g., [2]) that multiple shooting algorithms can experience difficulties due to the failure of the IVP solver or the need for a large number of subintervals. Two well-known multiple shooting codes are the MUSL/MUSN packages of Mattheij and Staarink; see [2] for more information.

Codes that numerically solve IVPs rarely attempt to directly control an estimate of the global error. Usually these codes control an estimate of the local error. Authors of these codes consider global error control to be too computationally costly. Unfortunately, a small local error does not necessarily guarantee

a small global error. However there are a few IVP solvers that use a computationally efficient scheme to compute and control an estimate of the global error. One such code, called `GERK` [26], uses global extrapolation techniques along with a Runge–Kutta–Fehlberg method to efficiently determine an estimate of the global error for each step. A more recent investigation in this area has undertaken by Lang and Verwer (see [20] and references within), where a standard approach based on the integration of the variational equation for the IVP is compared with a new approach by Cao and Petzold [7] based on the use of an adjoint method.

Most multiple shooting codes attempt to solve each of the local IVPs to within the given user tolerance (or fraction thereof) using IVP software that performs some form of local error control. As well, the matching and boundary conditions are solved iteratively to within the user tolerance (or fraction thereof). When this computation is successful, the result is a numerical solution for which it can be shown that the corresponding global error is at least *indirectly* controlled; see [2]. To our knowledge, there does not exist a multiple shooting code that uses an IVP solver that attempts to *directly* control an estimate of the global error when solving the local IVPs.

When an IVP solver employs "error-per-unit-step" local error control and an appropriate interpolant to the discrete numerical solution computed by the IVP solver, then it can be shown that the defect of the numerical solution is also *indirectly* controlled; see Section 2 of [13]. Thus a multiple shooting code that employs such an IVP solver also provides indirect control of the defect. Although there exist IVP solvers that *directly* control the defect, see, e.g., [12], to our knowledge none of these has ever been employed as the IVP solver within a multiple shooting code for BVODEs. And thus, to our knowledge, there does not exist a multiple shooting code that provides *direct* control of the defect of the numerical solution it computes.

1.2 Global Methods

Global methods, based on, e.g., finite difference, Runge–Kutta, or collocation methods for the discretization of the ODEs, generate a system of equations whose solution typically gives approximations to the solution of the BVODE at a set of mesh points that partition the solution domain. Moreover, some measure of the accuracy of the solution (e.g., a global error estimate or an estimate of the maximum defect) for each subinterval is normally computed. If the maximum of these estimates does not satisfy the user-specified tolerance, the error estimates are used to guide mesh refinement. Then based on the resultant mesh, a new approximate solution is computed, and the process repeats; see, e.g., [2], for more details.

1.2.1 Global Error Control Codes

One of the earliest BVODE codes of this type is the (`Fortran`) collocation code `COLSYS` [1]. In the approach implemented in this code, the approximate solution

is assumed to take the form of a piecewise polynomial with unknown coefficients that are determined by requiring the approximate solution to satisfy the ODEs at a set of collocation points over the problem domain. Several modifications of this code have been developed to improve or enhance the capabilities of this solver; the resultant codes include `COLNEW` [5], `COLDAE` [4], and `COLMOD` [10]. These codes estimate the global error by comparing the solution computed on a given mesh with a solution computed on a mesh that is obtained by "doubling" the original mesh; i.e., the second mesh is obtained by halving each subinterval of the original mesh. Although this technique, known as Richardson extrapolation, does provide a good quality error estimate, as we show in this paper, it is relatively computationally expensive.

A special class of multi-step methods, called Top Order Methods, are employed in the `MATLAB` BVODE solver, `TOM` [21], in order to discretize the ODEs. This code employs mesh selection based not only on a global error estimate but also on an estimate of the conditioning of the BVODE.

Mono-implicit Runge–Kutta (MIRK) methods, see, e.g., [6] and references within, and Lobatto collocation methods, see, e.g., [2], employed within a deferred correction framework, provide the basis for the `Fortran` BVODE solvers `TWPBVP` [11] and `ACDC` [10]. A related deferred correction code, `TWPBVPL` [8], also based on Lobatto schemes and deferred corrections, has also recently been developed. These codes control estimates of the global error and base mesh refinement on the global error estimates for each subinterval. Extensions of these codes that consider mesh selection based on global error estimates and an estimate of the conditioning constant of the BVODE have led to the development of new versions of TWPBVP and TWPBVPL, called TWPBVPC and TWPBVPLC [9].

None of the above codes attempt to directly estimate and control the defect of the numerical solution they return.

1.2.2 Defect Control Codes

Most BVODE codes provide a continuous solution approximation through the use of some form of interpolation. In such cases, Enright and Hanson [16] suggest the use of defects rather than global error to assess solution quality and guide mesh refinement because the defect is considerably less computationally expensive to compute than is an estimate of the global error. Codes that control the defect are also interesting from a backward error viewpoint: the numerical solution computed by such a code is the exact solution to a BVODE that is a perturbation of the original BVODE (see Section 3.2).

The BVODE solvers `bvp4c` [18], `bvp5c` [19] (`MATLAB`) and `MIRKDC` [14] and `BVP_SOLVER` [25] (`Fortran`) all use defect control. The two Fortran codes and `bvp4c` are based on MIRK formulas; `bvp5c` is based on a four-point Lobatto scheme. None of these defect control codes attempt to *directly* control the global error. However, `bvp5c` is able to indirectly control the global error; for the Lobatto formula employed by this code, it is shown in [19] that a scaled norm of the defect asymptotically approaches the norm of the global error. As

well, in Section 3.2, we show that any defect control code is able to *indirectly* control the global error provided the conditioning constant of the BVODE is small.

Although a numerical solution for which the estimate of the maximum defect is guaranteed to be less than the user tolerance is interesting, especially from a backward error viewpoint, there can be cases where a defect controlled numerical solution can have a large global error. In extreme cases, a defect control code can return a numerical solution for a problem that in fact does not have a solution. Shampine and Muir [24] refer to such solutions as *pseudo-solutions* and provide examples where `bvp4c` and `MIRKDC` return pseudo-solutions. In such cases, the numerical solution is still the exact solution to a BVODE that is "close" to the original one. However, if the BVODE is ill-conditioned and the tolerance is coarse, the solution of the perturbed problem may not be close to the solution of the original problem. This suggests that it is important for a defect control code to provide an assessment of the conditioning of the BVODE or an estimate of the global error of the numerical solution.

1.3 Estimation of the Global Error in `BVP_SOLVER`

The goal of this paper is explore the use of several standard approaches to global error estimation within the context of a defect control code. We assume that a primary solution with an estimated maximum defect satisfying a user-provided tolerance has been accepted. The idea is then to investigate possible algorithms for the generation of a robust but low cost a posteriori estimate of the global error in the accepted numerical solution. We emphasize that although the solution that is returned by a defect control code has the property that an estimate of the maximum *defect* over the whole problem domain is less than the user-provided tolerance, there is no guarantee that the *global error* is also less than the user tolerance.

Although `BVP_SOLVER` does not attempt to control global error directly, to our knowledge it is the only defect control code to provide the option for the *direct* computation of an a posteriori estimate of the global error; the estimate is based on Richardson extrapolation and has the form

$$\epsilon_{\mathrm{g,R}} = \left|\left| \frac{2^p}{2^p - 1} \left(\mathbf{Y}_h - \mathbf{Y}_{h/2} \right) \right|\right|_\infty , \tag{1}$$

where $\mathbf{Y}_h$ is the original defect controlled numerical solution, $\mathbf{Y}_{h/2}$ is the numerical solution computed on a mesh that is obtained by halving each subinterval of the original mesh, and p is the order of the MIRK method used in the computation of these solutions. Both $\mathbf{Y}_h$ and $\mathbf{Y}_{h/2}$ are evaluated at the points of the *original* mesh. Also, $\epsilon_{\mathrm{g,R}}$ is the global error estimate for the *lower* accuracy solution, $\mathbf{Y}_h$. A derivation of this error estimate can be found in section 5.5.2 of [2].

In this paper, we consider three other approaches for the computation of an a posteriori estimate of the global error of the primary numerical solution.

The first approach employs a higher order discretization formula to generate a more accurate numerical solution that can then be used to give a global error estimate. The second approach computes an estimate of the BVODE conditioning constant that, when multiplied by the estimate of the maximum defect of the primary solution, gives a bound for the global error. The third scheme uses a deferred correction approach to obtain a more accurate numerical solution that allows us to compute an estimate of the global error.

Although the global error estimators presented in this paper can be applied to any BVODE code, `BVP_SOLVER` is the primary focus. Therefore, section 2 describes the algorithms `BVP_SOLVER` uses to solve BVODEs. Section 3 describes the global error estimation techniques and their efficient implementation within `BVP_SOLVER`. Section 4 presents numerical experiments comparing the four global error estimators with respect to accuracy and efficiency. Finally section 5 summarizes our results, gives our conclusions, and provides suggestions for future work.

The estimate of the conditioning constant requires only a few extra back solves at the end of the computation, and it is therefore reasonable to expect that a global error estimate based on this approach would be low cost but potentially not particularly accurate because the underlying theory gives only an upper bound on the global error. On the other hand, the Richardson extrapolation approach is expected to provide an accurate estimate of the global error but at considerable expense as a larger discrete problem must be solved on a doubled mesh. The results of this paper verify these expectations and also show that the other two global error estimators combine the best of the above performances: the approaches based on the direct use of a higher order discretization method and deferred corrections provide the accuracy of the Richardson extrapolation approach but at a significantly lower cost. This suggests an improvement to the global error estimator currently implemented within `BVP_SOLVER`. Furthermore, these results of this paper provide a direct comparison of global error estimation algorithms that are useful for *global error control codes*, suggesting that an estimator based on either a higher order discretization method or deferred correction could replace a Richardson extrapolation approach in such codes.

2 Review of `BVP_SOLVER`

`BVP_SOLVER` is capable of solving a first-order system of n ODEs of the form

$$\mathbf{y}'(x) = \left(\frac{\mathbf{\Lambda}}{x-a}\right)\mathbf{y} + \mathbf{f}(x, \mathbf{y}, \mathbf{p}), \qquad a \le x \le b,$$

subject to nonlinear separated two-point boundary conditions

$$\mathbf{g}_a(\mathbf{y}(a), \mathbf{p}) = \mathbf{0}, \qquad \mathbf{g}_b(\mathbf{y}(b), \mathbf{p}) = \mathbf{0}.$$

Here $\mathbf{y}$ and $\mathbf{f}$ are vectors of length n and $\mathbf{p}$ is an optional vector of length n_p of unknown parameters. The vector $[\mathbf{g}_a, \mathbf{g}_b]^T$ is of length $n + n_p$. The $n \times n$

constant matrix $\mathbf{\Lambda}$ is optional. In this paper we assume the simpler form

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}), \quad a \le x \le b, \qquad \mathbf{g}_a(\mathbf{y}(a)) = \mathbf{0}, \quad \mathbf{g}_b(\mathbf{y}(b)) = \mathbf{0}. \tag{2}$$

In order to solve a BVODE, `BVP_SOLVER` generates a system of nonlinear equations for which the unknowns, $\mathbf{y}_i$, are approximations to the solution values, $\mathbf{y}(x_i)$, at the mesh points $a = x_0 < x_1 < \ldots < x_N = b$. The nonlinear system includes the separated boundary conditions,

$$\mathbf{g}_a(\mathbf{y}_0) = \mathbf{0}, \quad \mathbf{g}_b(\mathbf{y}_N) = \mathbf{0},$$

together with equations that discretize the ODEs on each subinterval using MIRK formulas. Let $h_i = x_i - x_{i-1}$, $i = 1, 2, \ldots, N$. On subinterval $i+1$, $[x_i, x_{i+1}]$, $i = 0, 1, \ldots, N-1$, these equations have the form

$$\boldsymbol{\phi}_{i+1}(\mathbf{y}_i, \mathbf{y}_{i+1}) = \mathbf{y}_{i+1} - \mathbf{y}_i - h_{i+1} \sum_{j=1}^{s} b_j \mathbf{f}(x_i + c_j h_{i+1}, \mathbf{Y}_j) = \mathbf{0}, \tag{3}$$

where

$$\mathbf{Y}_j = (1 - v_j)\mathbf{y}_i + v_j \mathbf{y}_{i+1} + h_{i+1} \sum_{k=1}^{j-1} a_{j,k} \mathbf{f}(x_i + c_k h_{i+1}, \mathbf{Y}_k), \tag{4}$$

for $j = 1, 2, \ldots, s$, are called the stages of the MIRK method.

The coefficients defining the MIRK method are represented by the modified Butcher tableau

$$\begin{array}{c|c|cccc} c_1 & v_1 & 0 & 0 & \ldots & 0 \\ c_2 & v_2 & a_{2,1} & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ c_s & v_s & a_{s,1} & a_{s,2} & \ldots & 0 \\ \hline & & b_1 & b_2 & \ldots & b_s \end{array},$$

where $c_j = v_j + \sum_{k=1}^{j-1} a_{j,k}$ and the $s \times s$ matrix $\mathbf{A}$, with (j,k)th component $a_{j,k}$, is strictly lower triangular.

Equation (3) represents n nonlinear equations involving the unknowns $\mathbf{y}_i$ and $\mathbf{y}_{i+1}$ associated with subinterval $i+1$, $i = 0, 1, \ldots, N-1$. Taking all these equations, for all subintervals, together with the boundary conditions, gives a system of $(N+1)n$ nonlinear equations whose solution gives the discrete solution at the mesh points, $\mathbf{Y} \equiv [\mathbf{y}_0^T, \mathbf{y}_1^T, \ldots, \mathbf{y}_N^T]^T$. This nonlinear system has the form

$$\mathbf{\Phi}(\mathbf{Y}) \equiv \begin{pmatrix} \mathbf{g}_a(\mathbf{y}_0) \\ \boldsymbol{\phi}_1(\mathbf{y}_0, \mathbf{y}_1) \\ \vdots \\ \boldsymbol{\phi}_N(\mathbf{y}_{N-1}, \mathbf{y}_N) \\ \mathbf{g}_b(\mathbf{y}_N) \end{pmatrix} = \mathbf{0}. \tag{5}$$

System (5) is solved using a Newton iteration, which requires the evaluation and factorization of the Jacobian

$$\frac{\partial \mathbf{\Phi}(\mathbf{Y})}{\partial \mathbf{Y}}. \tag{6}$$

When the Newton iteration converges, we obtain the discrete solution, $\{\mathbf{y}_i\}_{i=0}^{N}$, which serves as the basis for a C^1 continuous piecewise polynomial, $\mathbf{S}(x)$, that is generated from a continuous MIRK formula [22]. On subinterval $i+1$, $\mathbf{S}(x)$ takes the form

$$\mathbf{S}(x_i + \theta h_{i+1}) = \mathbf{y}_i + h_{i+1} \sum_{j=1}^{s^*} b_j(\theta) \mathbf{f}(x_i + c_j h_{i+1}, \mathbf{Y}_j),$$

where $0 \leq \theta \leq 1$ and $s^* \geq s$. In the above equation, each $b_j(\theta)$ is a known polynomial in θ, defined by a continuous MIRK method. Because $s^* \geq s$, it follows that $\mathbf{S}(x)$ may need to use extra stages; each such stage has the same general form as in (4). The piecewise polynomial, $\mathbf{S}(x)$, is a C^1 continuous approximation to the exact solution to the BVODE, $\mathbf{y}(x)$.

In order to assess the accuracy of $\mathbf{S}(x)$, `BVP_SOLVER` samples the defect,

$$\boldsymbol{\delta}(x) \equiv \mathbf{S}'(x) - \mathbf{f}(x, \mathbf{S}(x)), \tag{7}$$

of this approximate solution at several points on each subinterval and chooses the maximum of these samples as an estimate of the maximum defect. If the estimated maximum defect is greater than the user-prescribed tolerance on any subinterval, the current solution approximation is rejected. The estimates of the maximum defects on each subinterval are used to guide a process that attempts to construct a new mesh such that (i) the maximum defect estimates are approximately equidistributed over the subintervals of the new mesh, and (ii) the maximum defect estimate on each subinterval of the new mesh is less than the user tolerance. Once this new mesh is obtained, the computation described above is repeated. If the estimated maximum defect for each subinterval is less than the user-prescribed tolerance, the solution is accepted.

We can also associate with $\mathbf{S}(x)$, a different assessment of solution quality, namely, the global error, $\boldsymbol{\epsilon}_{\mathrm{g}}(x)$, given by,

$$\boldsymbol{\epsilon}_{\mathrm{g}}(x) \equiv \mathbf{y}(x) - \mathbf{S}(x). \tag{8}$$

However, because $\mathbf{y}(x)$ is generally not known, the global error must be estimated. As described in section 1.3, `BVP_SOLVER` currently uses a Richardson extrapolation technique, (1), for its a posteriori global error estimator. We now describe and evaluate the practical implementation of several other standard global error estimation techniques within `BVP_SOLVER`.

3 Implementation of New Global Error Estimators in BVP_SOLVER

This section describes three alternative techniques we have implemented within BVP_SOLVER in order to estimate the global error after the defect controlled numerical solution has been accepted by the code. An important point is that we do not estimate the global error until after a candidate final numerical solution has been obtained. It is therefore not appropriate to consider approaches that would add extra work to the computation of *all* the intermediate solutions computed by BVP_SOLVER during its efforts to obtain the primary defect controlled numerical solution.

Subsection 3.1 describes a global error estimation technique based on the direct use of higher order MIRK formulas. Subsection 3.2 examines defect control from a backward error analysis viewpoint and describes a global error estimator based on the norm of the defect and an estimate of the conditioning constant for the BVODE. Subsection 3.3 describes an approach based on the use of higher order formulas within a deferred correction framework.

3.1 Direct Use of a Higher Order MIRK Formula for Global Error Estimation

This section describes a global error estimator that uses a higher order MIRK formula to compute an approximate solution that is of higher order than the primary numerical solution, $\mathbf{S}(x)$, accepted by BVP_SOLVER. Such an approach is commonly used for local error estimation in the numerical solution of IVPs by Runge–Kutta methods, particularly in the form of embedded Runge–Kutta pairs. Suppose that at the end of the ith step, $[x_{i-1}, x_i]$, a numerical solution, $\mathbf{y}_i$, that approximates the true local solution, $\mathbf{y}(x_i)$, is computed and that a higher order numerical solution, $\bar{\mathbf{y}}_i$, is also determined. The difference of these two numerical solutions gives an estimate of the local error in the lower order solution, $\mathbf{y}_i$. The higher order solution, $\bar{\mathbf{y}}_i$, is computed by using a higher order method and is usually more computationally expensive to obtain than $\mathbf{y}_i$. However, embedded Runge–Kutta pairs can be used to improve efficiency because, once $\mathbf{y}_i$ is computed, only a small number of additional stage computations are required to compute $\bar{\mathbf{y}}_i$. In some cases, the higher order solution is propagated to the next step even though the available error estimate is for the lower order solution; this is known as *local extrapolation*. Embedded methods are described in greater detail in, e.g., [3].

In the approach we consider in this paper, we compute a higher order numerical solution by solving an additional nonlinear system obtained by discretizing the BVODE on the same mesh that was used to compute the primary solution but using a Runge–Kutta method that is 2 orders of accuracy higher than that used to compute the primary solution. That is, assuming that the primary solution is obtained using a MIRK method of order p, we obtain a numerical solution of order $p + 2$ by setting up and solving a nonlinear system of the

form (5) for which the underlying MIRK method is of order $p+2$. We choose a method 2 orders higher rather than 1 order higher because it is important to employ symmetric Runge–Kutta methods when solving a BVODE, and the symmetric methods have only even orders [23]. This global error estimate is only for the discrete solution approximation at the mesh points of the final mesh. That is, we do not compute an interpolant for the higher order solution to allow an estimation of the global error of the primary *continuous* numerical solution; however, this feature could be added without much additional cost.

`BVP_SOLVER` can solve BVODEs using a second-, fourth-, or sixth-order MIRK method; see [23] for the tableaus that define these formulas and their associated interpolants. Thus for primary solutions obtained using a second- or fourth-order MIRK formula, there is a natural MIRK formula available for the computation of the error estimate. For the case when the primary solution is obtained by using the sixth-order MIRK formula, we have added an eighth-order MIRK method [15] to `BVP_SOLVER`. Although the formulas included in `BVP_SOLVER` are optimal in a certain sense [23], there is no reason to expect that the eighth-order formula from [15] is also optimal, and further work could be done to develop an optimal eighth-order MIRK formula for this application. However, because the formula is only used on the final mesh, it is not clear that the use of an optimized eighth-order formula rather than the formula of [15] would lead to much improvement overall.

For completeness, we provide here the tableau for the ten-stage, eighth-order MIRK formula from [15] that we have implemented in `BVP_SOLVER`:

0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
$\frac{1}{4}$	$\frac{5}{32}$	$\frac{9}{64}$	$-\frac{3}{64}$	0	0	0	0	0	0	0	0
$\frac{3}{4}$	$\frac{27}{32}$	$\frac{3}{64}$	$\frac{-9}{64}$	0	0	0	0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	$-\frac{5}{24}$	$\frac{5}{24}$	$\frac{2}{3}$	$-\frac{2}{3}$	0	0	0	0	0	0
$\frac{1}{8}$	β	a_1	a_2	a_3	a_4	0	0	0	0	0	0
$\frac{7}{8}$	$1-\beta$	$-a_2$	$-a_1$	$-a_4$	$-a_3$	0	0	0	0	0	0
$\frac{7-\sqrt{21}}{14}$	Θ	a_5	a_6	0	0	0	a_7	a_8	0	0	0
$\frac{7+\sqrt{21}}{14}$	$1-\Theta$	$-a_6$	$-a_5$	0	0	0	$-a_8$	$-a_7$	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	$\frac{29}{896}$	$-\frac{29}{896}$	0	0	0	$\frac{-2}{21}$	$\frac{2}{21}$	$\frac{7\sqrt{21}}{128}$	$-\frac{7\sqrt{21}}{128}$	0
		$\frac{1}{20}$	$\frac{1}{20}$	0	0	0	0	0	$\frac{49}{180}$	$\frac{49}{180}$	$\frac{16}{45}$

where β is a free parameter, and

$$a_1 = \frac{757}{9216} - \frac{\beta}{18}, \qquad a_2 = \frac{43}{9216} - \frac{\beta}{18}, \qquad a_3 = \frac{235}{4608} - \frac{4\beta}{9},$$
$$a_4 = -\frac{59}{4608} - \frac{4\beta}{9}, \qquad a_5 = \frac{3451 + 717\sqrt{21}}{139258}, \qquad a_6 = \frac{-3451 + 717\sqrt{21}}{139258},$$
$$a_7 = \frac{64}{1029} + \frac{1024\sqrt{21}}{69629}, \quad a_8 = \frac{-64}{1029} + \frac{1024\sqrt{21}}{69629}, \quad \Theta = \frac{1}{2} + \frac{2211\sqrt{21}}{19894}.$$

This formula contains embedded formulas of orders 2, 4, and 6, and in [15] it is noted that if the embedded sixth-order formula is not required, stage five can be ignored. We do not make use of the embedded formulas, and thus we implement this formula as a nine-stage method. We also choose $\beta = 0$, which is a reasonable approximation to the value $\beta \approx 0.006970$ that minimizes the 2-norm of the Principal Error Function (see, e.g., [23]) for this eighth order method; the corresponding values for the norms of the ninth and tenth order error coefficients are 6.9950×10^{-6} and 6.9751×10^{-6}, respectively.

Denote the primary solution of order p, evaluated at the mesh points associated with the computation of the primary solution, by $\mathbf{Y}_p$. Denote the solution of order $p+2$, at the same set of points, obtained by using the higher order MIRK formula, by $\mathbf{Y}_{p+2}$. Then the estimate of the norm of the global error for $\mathbf{Y}_p$ is

$$\epsilon_{\text{g,HO}} = ||\mathbf{Y}_p - \mathbf{Y}_{p+2}||_\infty.$$

When implementing this scheme, several observations were exploited in order to obtain substantial savings in computation time. All observations relate to how Newton's method is used to solve the system of nonlinear equations, (5), based on the higher order MIRK formula.

First, from our numerical experiments, we have observed that the primary solution, $\mathbf{Y}_p$, proves to be an effective initial guess for the solution of the system of nonlinear equations based on the higher order MIRK formula. Because $\mathbf{Y}_p$ is saved in the solution structure employed by `BVP_SOLVER`, it is available for use as the initial guess at no additional cost in computation time. This is in contrast to using Richardson extrapolation where evaluations of the continuous numerical solution must first be performed to generate an initial guess at the additional mesh points associated with the doubled mesh.

Second, in order to solve the system of nonlinear equations associated with the higher order MIRK formula using Newton's method, a Jacobian matrix is required and the expensive evaluation and factorization of this matrix can be avoided by employing the Jacobian matrix used by `BVP_SOLVER` when it computes the primary solution. This matrix is also available within one of the arrays used by `BVP_SOLVER` during the computation of the primary solution. Again, this is in contrast to the Richardson extrapolation approach where, because the mesh has been doubled, there is no straightforward way to make use of the factored Jacobian from the primary solution computation. In our numerical experiments, the Jacobian matrix from the primary solution computation has proven to be a good approximation for the Jacobian associated with the nonlinear system based on the higher order MIRK formula.

Third, due to the effectiveness of the initial guess, $\mathbf{Y}_p$, we have observed in our experiments that no further Jacobian evaluations are required to achieve quick convergence of Newton's method. That is, we perform Newton iterations for which the Jacobian is held constant as long as the Newton iterates decrease at a sufficient rate (as is done in the Newton iterations performed during the computation of the primary solution), and in our experiments we have observed that we were able to hold the Jacobian constant for the entire iteration required

to determine $\mathbf{Y}_{p+2}$.

By making use of the (factored) Jacobian from the primary computation, the implementation of this global error estimate involves only a sequence of back solves, involving the evaluation of $\mathbf{\Phi}$ in (5). We have modified `BVP_SOLVER` to provide an option that implements the global error estimation scheme via higher order MIRK formula described above.

3.2 Global Error Estimation based on an Estimate of the Conditioning Constant of the BVODE

The second error estimation approach we consider is based on a backward error analysis for the numerical solution of a BVODE; for completeness we now briefly review the basic ideas; see, e.g., [24] for further details.

Suppose $\mathbf{S}(x)$ is a continuous approximate solution to the BVODE (2). Although $\mathbf{S}(x)$ is an *approximate* solution to (2), it is the *exact* solution to the BVODE

$$\mathbf{z}'(x) = \mathbf{f}(x, \mathbf{z}(x)) + \boldsymbol{\delta}(x), \quad a \le x \le b, \quad \mathbf{g}_a(\mathbf{z}(a)) = \boldsymbol{\sigma}_a, \quad \mathbf{g}_b(\mathbf{z}(b)) = \boldsymbol{\sigma}_b, \quad (9)$$

where the defect $\boldsymbol{\delta}(x)$ is given by (7), and the boundary condition defects are $\boldsymbol{\sigma}_a = \mathbf{g}_a(\mathbf{S}(a))$, and $\boldsymbol{\sigma}_b = \mathbf{g}_b(\mathbf{S}(b))$. When $\mathbf{S}(x)$ is computed by a defect control code, we have

$$||\boldsymbol{\delta}(x)|| \le TOL, \quad ||\boldsymbol{\sigma}_a|| \le TOL, \quad ||\boldsymbol{\sigma}_b|| \le TOL,$$

for an appropriate norm, where TOL is the user tolerance. Thus $\mathbf{S}(x)$ is the exact solution to the BVODE, (9), that differs from the original BVODE (2) by an amount that is bounded by the user tolerance, typically chosen to be small. *Although $\boldsymbol{S}(x)$ is the exact solution to a problem that is close to the original problem, $\boldsymbol{S}(x)$ is not necessarily close to $\boldsymbol{y}(x)$, the exact solution to (2).* The relationship between the defect and the global error depends on the conditioning constant of the BVODE. We now briefly sketch the derivation of this relationship for the case of a linear BVODE with coupled linear boundary conditions; the results can be extended to the nonlinear case in a straightforward manner [2].

Assume that the exact solution, $\mathbf{y}(x)$, satisfies the BVODE,

$$\mathbf{y}'(x) = \mathbf{A}(x)\mathbf{y}(x) + \mathbf{q}(x), \quad \mathbf{B}_a\mathbf{y}(a) + \mathbf{B}_b\mathbf{y}(b) = \boldsymbol{\beta}. \quad (10)$$

In (10), $\mathbf{A}(x), \mathbf{B}_a, \mathbf{B}_b \in R^{n\times n}$ and $\mathbf{q}(x), \mathbf{y}(x), \boldsymbol{\beta} \in R^n$. A *fundamental solution* for (10), $\mathbf{Y}(x) \in R^{n\times n}$, is defined as the solution of the IVP system

$$\mathbf{Y}'(x) = \mathbf{A}(x)\mathbf{Y}(x), \quad a < x < b, \quad \mathbf{Y}(a) = \mathbf{I}.$$

Then the solution to (10) can then be expressed as

$$\mathbf{y}(x) = \mathbf{Y}(x)\mathbf{Q}^{-1}\boldsymbol{\beta} + \int_a^b \mathbf{G}(x,t)\mathbf{q}(t)dt, \quad (11)$$

where $\mathbf{G}(x,t)$ is the associated Green's function and $\mathbf{Q}$ is the non-singular matrix

$$\mathbf{Q} = \mathbf{B}_a\mathbf{Y}(a) + \mathbf{B}_b\mathbf{Y}(b).$$

In this case, the approximate solution, $\mathbf{S}(x)$, exactly satisfies the perturbed BVODE

$$\mathbf{S}'(x) = \mathbf{A}(x)\mathbf{S}(x) + \mathbf{q}(x) + \boldsymbol{\delta}(x), \tag{12}$$

and the perturbed boundary conditions

$$\mathbf{B}_a\mathbf{S}(a) + \mathbf{B}_b\mathbf{S}(b) = \boldsymbol{\beta} + \boldsymbol{\sigma},$$

where

$$\boldsymbol{\delta}(x) = \mathbf{S}'(x) - \mathbf{A}(x)\mathbf{S}(x) - \mathbf{q}(x),$$

and

$$\boldsymbol{\sigma} = \mathbf{B}_a\mathbf{S}(a) + \mathbf{B}_b\mathbf{S}(b) - \boldsymbol{\beta},$$

are the defects associated with the ODE and the boundary conditions, respectively. Because $\mathbf{S}(x)$ satisfies (12), we can use the associated Green's function to obtain

$$\mathbf{S}(x) = \mathbf{Y}(x)\mathbf{Q}^{-1}\boldsymbol{\beta} + \mathbf{Y}(x)\mathbf{Q}^{-1}\boldsymbol{\sigma} + \int_a^b \mathbf{G}(x,t)\mathbf{q}(t)dt + \int_a^b \mathbf{G}(x,t)\boldsymbol{\delta}(t)dt. \tag{13}$$

Subtracting (11) from (13) gives an expression for the global error of $\mathbf{S}(x)$:

$$\mathbf{S}(x) - \mathbf{y}(x) = \mathbf{Y}(x)\mathbf{Q}^{-1}\boldsymbol{\sigma} + \int_b^a \mathbf{G}(x,t)\boldsymbol{\delta}(t)dt. \tag{14}$$

We next introduce weight functions to take into account the weighted norm used in `BVP_SOLVER` to scale the defect; see [25] for further details. Let $\mathbf{w}_1(x)$, $\mathbf{w}_2$, and $\mathbf{w}_3(x)$ be $n \times n$ diagonal matrices with positive entries. The weighting functions $\mathbf{w}_2$ and $\mathbf{w}_3(x)$ are associated with the scaling of the defect; $\mathbf{w}_1(x)$ is associated with a corresponding scaling for the global error. Then weighted norms are defined as follows:

$$||\boldsymbol{\delta}(x)||_{\mathbf{w}_1} = \max_{a\le x\le b} ||\mathbf{w}_1^{-1}(x)\boldsymbol{\delta}(x)||_\infty, \quad ||\boldsymbol{\sigma}||_{\mathbf{w}_2} = ||\mathbf{w}_2^{-1}\boldsymbol{\sigma}||_\infty,$$

$$||\mathbf{S}(x) - \mathbf{y}(x)||_{\mathbf{w}_3} = \max_{a\le x\le b} ||\mathbf{w}_3^{-1}(x)\left(\mathbf{S}(x) - \mathbf{y}(x)\right)||_\infty.$$

Rewriting (14) as

$$\mathbf{w}_3^{-1}(x)(\mathbf{S}(x) - \mathbf{y}(x)) = (\mathbf{w}_3^{-1}(x)\mathbf{Y}(x)\mathbf{Q}^{-1}\mathbf{w}_2)(\mathbf{w}_2^{-1}\boldsymbol{\sigma})$$

$$+ \int_a^b (\mathbf{w}_3^{-1}(x)\mathbf{G}(x,t)\mathbf{w}_1(t))(\mathbf{w}_1^{-1}(t)\boldsymbol{\delta}(t))dt.$$

and taking norms, we get

$$||\mathbf{y}(x) - \mathbf{S}(x)||_{\mathbf{w}_3} \le \kappa \max\left(||\boldsymbol{\delta}(x)||_{\mathbf{w}_1}, ||\boldsymbol{\sigma}||_{\mathbf{w}_2}\right), \tag{15}$$

where the conditioning constant, κ, for the BVODE is given by

$$\kappa = \max_{a \le x \le b} \int_a^b ||\mathbf{w}_3^{-1}(x)\mathbf{G}(x,t))\mathbf{w}_1(t)||_\infty dt + ||\mathbf{w}_3^{-1}(x)\mathbf{Y}(x)\mathbf{Q}^{-1}\mathbf{w}_2||_\infty.$$

Inequality (15) says that the global error is bounded by the product of the conditioning constant for the BVODE and the defect, and therefore a bound on the global error can be obtained from

$$\epsilon_{\mathrm{g,CO}} \equiv \kappa \|\boldsymbol{\delta}\|_\infty.$$

Because a defect control code must compute an estimate of the maximum defect, the remaining work involves the computation of an estimate of the conditioning constant.

The paper [24] also explains how to compute an estimate of κ. Let $\mathbf{W}_{12} = \mathrm{diag}\{\mathbf{w}_1(x_2), \ldots, \mathbf{w}_1(x_{N+1}), \mathbf{w}_2\}$ and $\mathbf{W}_3 = \mathrm{diag}\{\mathbf{w}_3(x_1), \ldots, \mathbf{w}_3(x_{N+1})\}$. Then in [24], it is shown that, for a sufficiently fine mesh,

$$\kappa \approx ||\mathbf{W}_3^{-1} \frac{\partial \boldsymbol{\Phi}(\mathbf{Y})}{\partial \mathbf{Y}}^{-1} \mathbf{W}_{12}^{-1}||_\infty,$$

where $\frac{\partial \boldsymbol{\Phi}(\mathbf{Y})}{\partial \mathbf{Y}}$ is the Newton matrix, (6), which is computed and factored during the computation to determine the primary numerical solution.

Because it is impractical to compute $||\mathbf{W}_3^{-1} \frac{\partial \boldsymbol{\Phi}(\mathbf{Y})}{\partial \mathbf{Y}}^{-1} \mathbf{W}_{12}^{-1}||_\infty$ directly (due to the presence of $\frac{\partial \boldsymbol{\Phi}(\mathbf{Y})}{\partial \mathbf{Y}}^{-1}$), [24] suggests the use of the Higham–Tisseur algorithm [17] for the efficient estimation of the norm of $||\mathbf{W}_3^{-1} \frac{\partial \boldsymbol{\Phi}(\mathbf{Y})}{\partial \mathbf{Y}}^{-1} \mathbf{W}_{12}^{-1}||_\infty$. Once the primary solution is accepted, and because the factored Newton matrix that was used by `BVP_SOLVER` to obtain the primary solution is still available, the Higham–Tisseur algorithm can be used to obtain the estimate for κ using only a few additional back solves, involving the matrix $\mathbf{W}_{12} \frac{\partial \boldsymbol{\Phi}(\mathbf{Y})}{\partial \mathbf{Y}} \mathbf{W}_3$, where the matrix $\frac{\partial \boldsymbol{\Phi}(\mathbf{Y})}{\partial \mathbf{Y}}$ is already in factored form.

We have modified `BVP_SOLVER` to use the above approach to efficiently estimate κ after the primary solution is accepted. The product of κ and the estimate of the maximum norm of the defect then gives a bound on the global error, as indicated above. However, it is worth noting that, especially for a defect control code, it may be useful to estimate and return κ itself because this quantity gives a measure of the sensitivity of the solution to small changes in the problem definition. In this case the bound on the global error is obtained at the cost of only one multiplication.

3.3 Global Error Estimation based on Deferred Corrections

We begin the discussion of this approach by considering the system of nonlinear equations, (5), whose solution gives a discrete approximation to the exact solution evaluated at the mesh points. When the MIRK method upon which (5) is

based is of order p, let us refer to this system as

$$\mathbf{\Phi}_p(\mathbf{Y}_p) = \mathbf{0},$$

where the pth order discrete solution, obtained by solving this system, is $\mathbf{Y}_p$.

In [11], the authors describe a *deferred correction method* based on MIRK formulas. They demonstrate how solutions of orders four, six, and eight can be computed by solving the systems

$$\mathbf{\Phi}_4(\mathbf{Y}_4) = \mathbf{0}, \quad \mathbf{\Phi}_4(\mathbf{Y}_6) = -\mathbf{\Phi}_6(\mathbf{Y}_4), \quad \mathbf{\Phi}_4(\mathbf{Y}_8) = \mathbf{\Phi}_4(\mathbf{Y}_6) - \mathbf{\Phi}_8(\mathbf{Y}_6). \tag{16}$$

In [11], the authors first compute a solution of order four and then use two steps of deferred correction to efficiently compute the solutions of orders six and eight. That is, the fourth-order solution $\mathbf{Y}_4$ is computed using the first equation from (16) and the sixth- and eighth-order solutions, $\mathbf{Y}_6$ and $\mathbf{Y}_8$, are obtained by solving the second and third equations from (16).

Because `BVP_SOLVER` uses MIRK formulas, we can use an approach similar to that of [11]; however, we use only one step of deferred correction to obtain a higher order solution that can give us a global error estimate. When `BVP_SOLVER` solves a BVODE it returns one solution of order two, four, or six, depending on what order of method the user has selected. This gives what we have referred to as the primary solution. Assume the primary solution is of order p; i.e., we have $\mathbf{Y}_p$ available from the primary defect controlled computation. Then the deferred correction equation that allows us to obtain the higher order solution, $\mathbf{Y}_{p+2}$, is

$$\mathbf{\Phi}_p(\mathbf{Y}_{p+2}) = -\mathbf{\Phi}_{p+2}(\mathbf{Y}_p).$$

That is, we need to solve the nonlinear system

$$\mathbf{\Phi}_p(\mathbf{z}) + \mathbf{\Phi}_{p+2}(\mathbf{Y}_p) = \mathbf{0}, \tag{17}$$

for the unknown $\mathbf{z}$. A Newton iteration is employed to obtain $\mathbf{z}$, and upon convergence, we set $\mathbf{Y}_{p+2} = \mathbf{z}$. The primary expense is the setup and factorization of the Jacobian matrix of this nonlinear system. However, the system

$$\mathbf{\Phi}_p(\mathbf{z}) = \mathbf{0}, \tag{18}$$

is the one that was just solved during the primary computation to get $\mathbf{Y}_p$. That computation was based on a Newton iteration and the corresponding Jacobian (evaluated at $\mathbf{Y}_p$ or an approximation to it) was computed and factored for use in that iteration. A significant advantage of employing (17) to determine $\mathbf{Y}_{p+2}$ is that that system has the same Jacobian matrix as (18). Furthermore, a natural initial guess for $\mathbf{Y}_{p+2}$ to start the Newton iteration for (17) is to use $\mathbf{Y}_{p+2}^{(0)} = \mathbf{Y}_p$. Thus in the Newton iteration used to compute the solution, $\mathbf{Y}_{p+2}$, to (17), we can employ the already computed and factored Jacobian from the Newton iteration used in the primary computation to get $\mathbf{Y}_p$.

As in the approach described in Section 3.1, once the higher order solution $\mathbf{Y}_{p+2}$ is available, the estimate of the norm of the global error for $\mathbf{Y}_p$ is

$$\epsilon_{\mathrm{g,DC}} = ||\mathbf{Y}_p - \mathbf{Y}_{p+2}||_\infty.$$

The computational costs incurred in this approach involve only a sequence of evaluations of $\mathbf{\Phi}_p(\mathbf{z})$ and corresponding back solves associated with applying Newton's method with a fixed Jacobian to (17) and the computation of the correction term, $\mathbf{\Phi}_{p+2}(\mathbf{Y}_p)$, once at the beginning of the iteration. We have modified `BVP_SOLVER` to allow it to perform this additional computation after the primary solution has been accepted.

3.4 Summary

Previous subsections have described the various global error estimations schemes for which we present numerical results in the next section. Based on the descriptions of these estimators, we can make an assessment of their relative costs. The approach based on Richardson extrapolation requires the setup and factorization of a Newton matrix that is twice the size of that employed to compute the primary accepted solution on the final mesh. This obviously represents a significant additional cost, and we therefore expect to see that the costs for this approach are significantly larger than the other estimators. The deferred correction approach makes use of the already factored Newton matrix during the computation of the pth order primary solution, and as can be seen from (17) and (18), this is exactly the required Newton matrix. This is in contrast to the approach that directly employs a higher order method, as described in Section 3.1, where the system to be solved should employ a Newton matrix based on the MIRK method of order $p + 2$, but we instead use the already factored Newton matrix associated with the computation of the pth order primary solution. Thus the Newton matrix we employ in this case is only an approximation to the matrix that we should employ, and we might expect that more iterations to obtain convergence are required than for the approach based on deferred corrections in some cases. Both of these methods require each Newton iteration to compute a fairly complicated right hand side vector based on a MIRK method; on the other hand the approach based on the estimation of a conditioning constant for the BVODE uses the already factored Newton matrix from the primary computation and only simple right hand sides computed by the conditioning constant estimation software. We therefore expect its costs to be less than the previous two approaches.

4 Numerical Results

With the implementation of the three approaches for global error estimation discussed in Sections 3.1, 3.2, and 3.3 and the Richardson extrapolation global error estimate that was already available, `BVP_SOLVER` now has four possible methods for estimating the global error. This section presents a comparison of accuracy and computational efficiency for these four estimators. We consider five test problems, and for each test problem we examine the performance of `BVP_SOLVER` and the global error estimators for the three MIRK formula order options: 2, 4, and 6. All test problems were first converted to first order sys-

tems as required by `BVP_SOLVER`. In order to assess the quality of the global error estimates, we need to compare them with the true global error. For test problems with a known exact solution, we provide this solution in the problem description and use it to compute the exact global error. For the other problems, we compute a high precision solution using the `BVP_SOLVER` package with a stringent tolerance setting and use this numerical solution as the reference solution.

The computations were performed using an Intel Xeon w3520 quad core processor running at 2.667 GHz. The RAM consists of 6GB of DDR3 memory running at 1.333 GHz. The operating system is Ubuntu 8.04 with kernel 2.6.24-25-generic and the `Fortran` compiler is gfortran with gcc 4.2.4-lubuntu4.

4.1 Test Problems

The first problem is the linear problem

$$y'' - \frac{y'}{x^2} + 100y = 1000x - \frac{10}{x^2} + \frac{10\cos(10x)}{x^2}, \tag{19}$$

with boundary conditions,

$$y'(0) = 0, \quad y(1) = 10 - \sin(10).$$

It can be found in [18]. The problem has the exact solution,

$$y(x) = 10x - \sin(10x).$$

Because the problem has a singularity at the left endpoint $a = 0$, we solved the problem with $a = 0.0038$. To achieve stable, measurable timings, the problem was solved 15 times in a loop - timing results are accumulated over all iterations. An initial guess $y(x) \equiv y'(x) \equiv 0$ was used for each iteration.

The second problem is the nonlinear problem (see [9]),

$$\epsilon y'' + (y')^2 = 1, \tag{20}$$

with boundary conditions

$$y(0) = 1 + \epsilon \ln\cosh\frac{-0.745}{\epsilon}, \quad y(1) = 1 + \epsilon \ln\cosh\frac{0.255}{\epsilon}.$$

The problem has the exact solution,

$$y(x) = 1 + \epsilon \ln\cosh\left(\frac{x - 0.745}{\epsilon}\right).$$

To achieve stable, measurable timings, the problem was solved in a loop of 80 iterations. For each iteration, we use $\epsilon = 0.028$ and the initial guess $y(x) \equiv \frac{1}{2}$, $y'(x) \equiv 0$. Timing results are the accumulated totals over all iterations.

The third problem is the nonlinear Problem 21 from the BVP test set at `www.ma.ic.ac.uk/~jcash/BVP_software`; it is

$$\epsilon y'' = y + y^2 - \exp\left(\frac{-2x}{\sqrt{x}}\right), \tag{21}$$

with boundary conditions,

$$y(0) = 1, \quad y(1) = \exp\left(\frac{-1}{\sqrt{\epsilon}}\right).$$

The problem has the exact solution,

$$y(x) = \exp\left(\frac{-x}{\sqrt{\epsilon}}\right).$$

We use $\epsilon = 10^{-7}$ and the guess $y(x) \equiv \frac{1}{2}$, $y'(x) \equiv 0$. To achieve measurable timings, the problem was solved in a loop of 10 iterations. Timing results are cumulative over all iterations.

The fourth problem is the nonlinear Example 1.19 from [2]; it has the form

$$\begin{aligned} f''' + \frac{1}{2}(3-n)ff'' + n(f')^2 + g^2 - sf' &= \gamma^2, \\ g'' + \frac{1}{2}(3-n)fg' + (n-1)gf' - s(g-1) &= 0, \end{aligned} \tag{22}$$

with boundary conditions,

$$f(0) = f'(0) = g(0) = 0, \quad f'(\infty) = 0, \quad g(\infty) = \gamma,$$

where $S = 0.05$, $n = 0.3$, and $\gamma = 1$. Because the problem is posed on a semi-infinite interval, x is transformed to

$$t = \frac{1}{x+1}, \quad 0 < t \leq 1.$$

The transformed problem has a singularity at the left endpoint. Accordingly, we solve this transformed problem with a left end point $a = 0.0025$; we use initial solution guesses $f(t) \equiv g(t) \equiv 0.8$, $f'(t) \equiv f''(t) \equiv g'(t) \equiv 0$. To achieve stable, measurable timings, the problem was solved repeatedly in a loop of 45 iterations. Timing results are for the cumulative run time over all iterations. Because no exact solution for this problem is known, a reference solution was computed by `BVP_SOLVER` using the sixth-order MIRK method with a tolerance of 10^{-11}.

The fifth problem is Example 1.20 of [2] given by

$$\epsilon f''' + ff''' + gg' = 0, \quad \epsilon g'' + fg' - f'g = 0, \tag{23}$$

with boundary conditions,

$$f(0) = f(1) = f'(0) = f'(1) = 0, g(0) = \Omega_0, \quad g(1) = \Omega_1,$$

where $\Omega_0 = -1$ and $\Omega_1 = -1$ and $\epsilon = 9 \times 10^{-5}$. The initial solution guesses are $g(x) = 2x - 1$, $g'(x) = 2$, and $f(x) \equiv f'(x) \equiv f''(x) \equiv f'''(x) \equiv 0$. To achieve stable, measurable timings, the problem was solved repeatedly in a loop of 30 iterations. Timing results are for the cumulative run time over all iterations. No exact solution is known and a numerical reference solution, obtained using the same approach as for the fourth problem, is employed.

4.2 Results for Second Order

`BVP_SOLVER` was used, with *method* = 2 in this case, to solve the five test problems as described above, over a range of tolerance values, typically from 10^{-4} to 10^{-8}. In essentially all results there is excellent agreement between the exact global error and the estimated global error from Richardson Extrapolation (RE), the use of a higher order method (HO), and deferred correction (DC). The results from the use of the conditioning constant estimate on the other hand give a substantial overestimate of the global error, typically overestimating the global error by several orders of magnitude. In the Appendix we provide tables of results for the second order case for each test problem. See Tables 1, 2, 3, 4, and 5 for results on problems (19), (20), (21), (22), and (23), respectively.

In order to illustrate the main points of our investigation, we present several graphs based on some of the data from the tables; these allow us to investigate the relative efficiency of the estimators by considering plots of tolerance vs. execution time of each global error estimator expressed as a percentage of the time required to compute the primary solution. Typical results are obtained for test problem (19); see Figure 1. Similar results were obtained for problems (21), (22), and (23) - see Figure 2, Figure 3 and Figure 4, respectively. The execution time for RE is a much higher percentage of the primary solution computation time than the HO or DC estimators whereas the execution for the CO estimator is essentially negligible.

For the test problem (20), we see slightly different results for the coarsest tolerance; see Figure 5. The HO estimator cost in this case is comparable to that of the RE estimator.

In summary, over a large majority of the tests involving the second order MIRK method, the RE estimator costs are generally a large fraction of the primary solution computation costs, whereas the HO and DC costs are usually comparable in cost and represent a significantly smaller fraction of the overall cost. The costs associated with the CO method are negligible for all tolerances and problems. However, the global error estimate from the CO method is such a substantial overestimate of the global error that it is not of practical interest. The other three estimators are in excellent agreement, in general, with the exact global error. The Newton iterations associated with all of the estimation schemes converge sufficiently well that it is possible to iterate to convergence without the need for an update of the Newton matrix; typically the RE and DC methods require only a small number of Newton iterations to obtain convergence to the desired tolerance; the HO method sometimes needs a few extra iterations. The CO method typically requires 5 iterations but the evaluation of the right

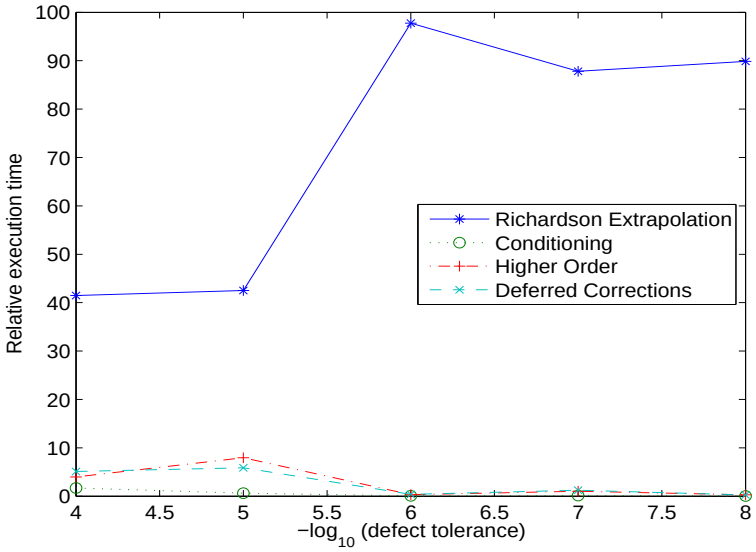

Figure 1: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for second order for test problem (19).

hand side vectors has a relatively negligible cost. (See the Appendix Tables 1, 2, 3, 4, and 5 for further details.)

4.3 Results for Fourth Order

We applied `BVP_SOLVER` with *method* = 4 to solve the five problems over the same range of tolerance values considered in the previous subsection. As before, there is generally excellent agreement between the exact global error and the estimated global error from RE, HO, and DC approaches, and the CO approach gives a substantial overestimate of the global error. Details are provided in the Appendix. See Tables 6, 7, 8, 9, and 10 for results on problems (19), (20), (21), (22), and (23), respectively.

We present several graphs based on some of the data from the tables in order to investigate the relative efficiency of the estimators for the fourth order case. Figure 6 and Figure 7 show typical results for the test problems (21) and (23): the RE approach has a cost that is significantly greater than that of the DC and HO methods, and the CO approach has a relatively negligible cost. Slightly different results are obtained for the test problems (19) and (20) - see Figure 8 and Figure 9. The RE cost is somewhat smaller and the HO cost is somewhat greater for these problems.

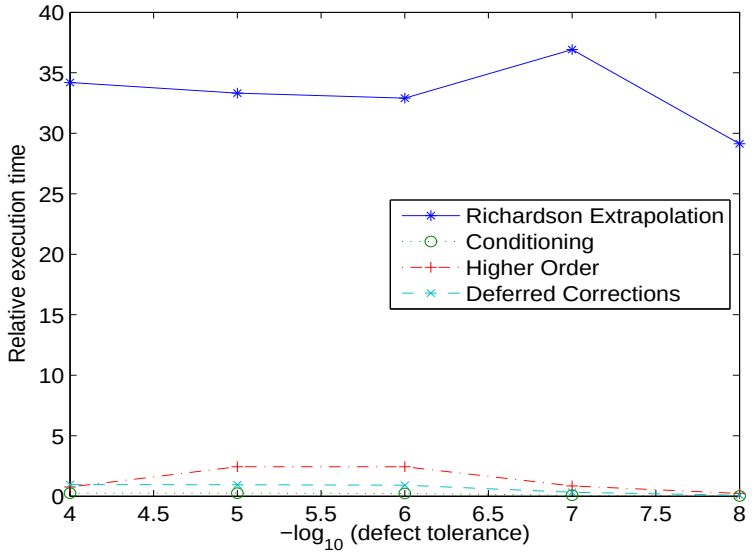

Figure 2: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for second order for test problem (21).

For test problem (22) all of the estimators have relatively small costs when measured as a percentage of primary solution execution time - see Figure 10. This is attributable to the large amount of time required to obtain the primary solution for this problem.

In summary, it is generally the case that the RE approach has a substantially higher cost than that of either the HO and DC approaches, which are themselves generally comparable in cost, and, as in the second order case, the CO approach has a negligible cost. From the tables given in the Appendix - see Tables 6, 7, 8, 9, and 10 - it can also be seen that the RE, HO, and DC estimates are generally accurate and in good agreement while the CO method gives substantial overestimates of the error.

4.4 Results for Sixth Order

In this subsection we discuss results for `BVP_SOLVER` with *method* = 6 applied to the five problems over the same tolerance values as in the previous two cases. The accuracy results are consistent with the previous two cases. The efficiency results are generally consistent with those reported in the previous subsection. We illustrate this with the graphs given below. Further details are provided in the tables in the Appendix. See Tables 11, 12, 13, 14, and 15 for results on

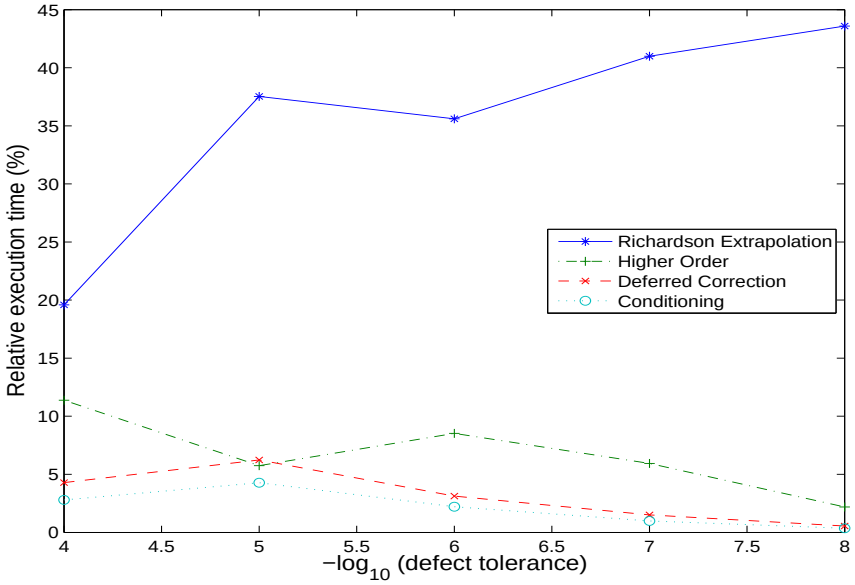

Figure 3: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for second order for test problem (22).

problems (19), (20), (21), (22), and (23), respectively.

Figure 11 shows typical results; these were obtained for test problem (21). Test problems (19) and (23) show similar results - see Figure 12 and Figure 13. The other problems, for the most part, yield similar results, with a few exceptions. For test problem (20), the cost of the DC estimator is comparable to that of the RE estimator for the coarser tolerances; see Figure 14. As in the fourth order case, test problem (22) shows all of the estimators having relatively small costs compared to the cost of the primary solution computation - see Figure 15.

In summary, the results for the sixth order case are generally comparable to the results from the other two cases: the RE approach is usually more expensive than the HO and DC approaches, which have similar costs, and the CO approach has a relatively negligible cost. And except for the CO method, all estimators give good approximations to the global error.

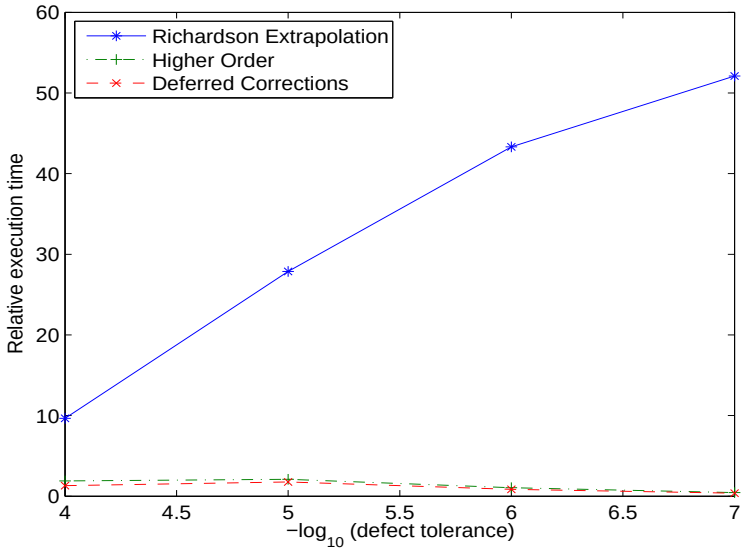

Figure 4: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for second order for test problem (23). Costs for the CO approach are negligible and are not included.

5 Summary, Conclusions, and Future Work

5.1 Summary

Although several well-known BVODE codes control an estimate of the maximum defect of the approximate solution they return, and this kind of control is interesting from a backward error perspective, users of these codes are often also interested in obtaining an estimate of the global error of the approximate solution. One recently developed defect control code, `BVP_SOLVER`, provides an option for an a posteriori estimate (based on Richardson extrapolation) of the global error of the defect controlled solution it returns. This paper considers three other global error estimators for defect control BVODE codes. These are a global error estimator that makes direct use of higher order methods, one that makes use of a conditioning constant, and one that is based on deferred corrections. Efficient implementations of these estimators have been added to the `BVP_SOLVER` package.

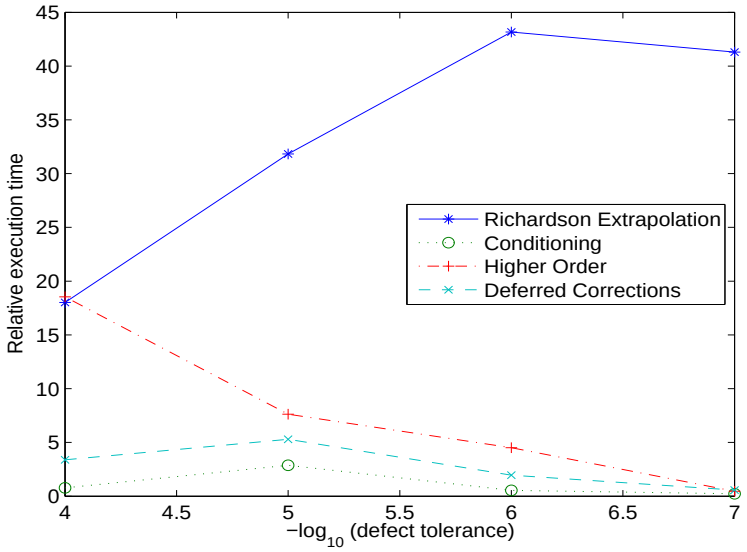

Figure 5: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for second order for test problem (20).

5.2 Conclusions

After comparing the three global error estimators, along with one already available in `BVP_SOLVER`, we can make several observations. First, both the approach based on the direct use of a higher order MIRK method and the use of a higher order MIRK method within a deferred correction approach are generally faster than the Richardson extrapolation estimator while achieving a global error estimate with the same accuracy. The Richardson extrapolation estimate, although of good quality, can represent a significant additional cost. Second, the estimator that uses an estimate of the conditioning constant has a relatively negligible cost but does not have comparable accuracy. However, a user may wish to compute an estimate of the conditioning constant in order to be aware of possible ill-conditioning for a given BVODE [24]. In that case, the conditioning constant based global error estimate can be computed essentially for free by making use of the last defect norm the code has computed for the solution.

We can draw several conclusions from the results presented in this paper:

(i) The results suggest that the a posteriori global error estimation employed by `BVP_SOLVER` should be based on the direct use of a higher order MIRK method or the use of a higher order MIRK method with a deferred correction framework rather than Richardson extrapolation. Furthermore, as

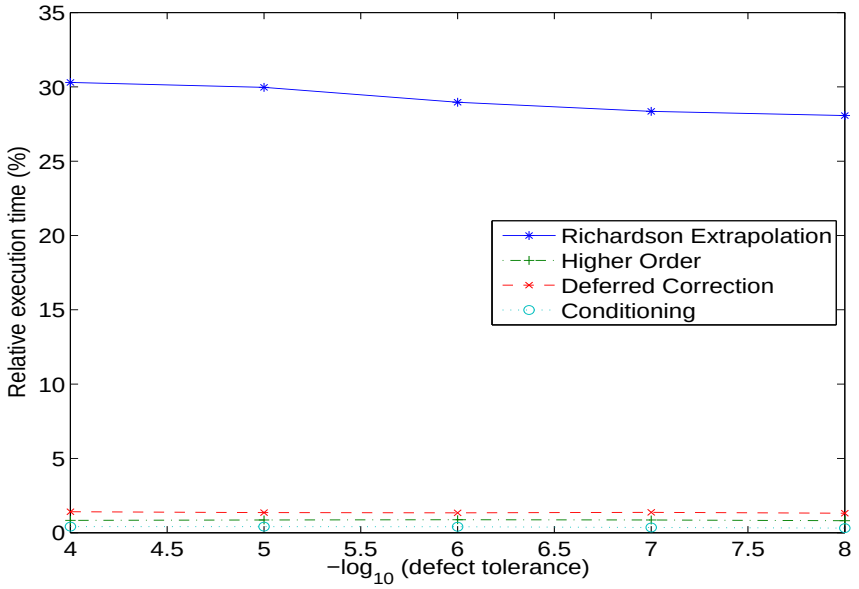

Figure 6: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (21).

long as the factored Jacobian matrix from the computation of the primary solution is available, this conclusion is also relevant for other defect control codes.

(ii) Prior to conducting this investigation, we felt that the estimation of the conditioning constant for the BVODE would cost less than any of the global error estimation schemes (although we were aware that the accuracy of the corresponding estimate might not be as good as that of the other approaches since it provides a upper bound on the global error rather than an estimate of the global error.) The results presented in this paper show that the more efficient of the global error estimators cost somewhat more than the estimation of the conditioning constant and yet provide much more accurate results. This suggests that a better way of estimating the conditioning constant for a BVODE may be through the direct estimation of the global error: rewriting (15), we get

$$\frac{||\mathbf{y}(x) - \mathbf{S}(x)||_{\mathbf{w}_3}}{\max(||\boldsymbol{\delta}(x)||_{\mathbf{w}_1}, ||\boldsymbol{\sigma}||_{\mathbf{w}_2})} \leq \kappa,$$

which gives a lower bound on κ.

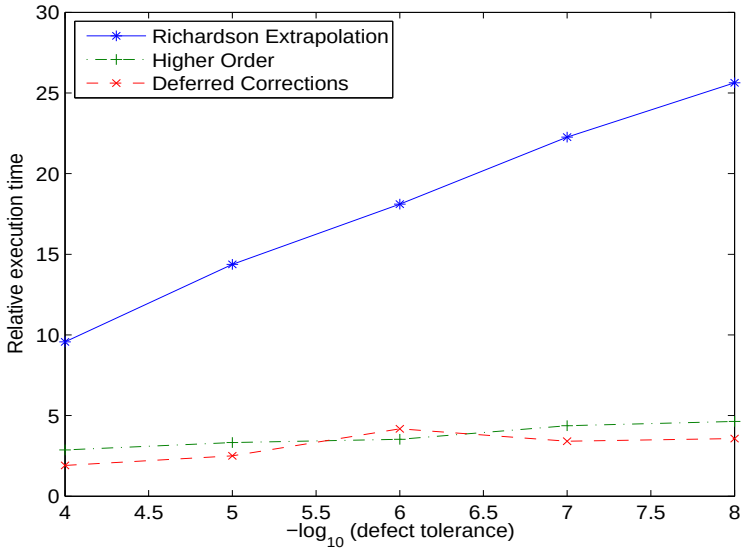

Figure 7: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (23). Costs for the CO approach are negligible and are not included.

(iii) As mentioned above, the global error estimators considered in this paper are quite general and thus the results presented in this paper are also relevant for *global error control codes*. In particular, they suggest that an improvement to the performance of codes, such as `COLSYS`, that employ Richardson extrapolation for global error estimation can be obtained instead by the direct use of a higher order discretization method or by the use of a higher order discretization method within a deferred correction approach.

5.3 Future Work

One possible direction for future work would be to implement continuous extensions of the higher order solutions provided by the Richardson extrapolation, high order MIRK, and deferred corrections schemes. This would allow an assessment of the global error of the *continuous* approximate solution obtained from the primary computation. This continuous solution is in fact what is provided to the user, and thus an assessment of the global error for the continuous approximate solution would be more stringent than the current one, which is based only on the solution values at the mesh points.

Another possible area of investigation would be to explore the feasibility of

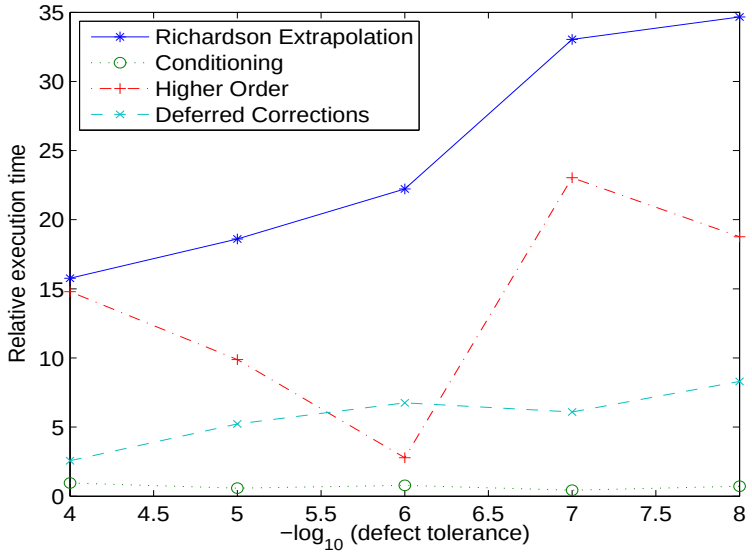

Figure 8: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (19).

employing the factored Newton matrix from the computation of the primary solution within the Newton iteration performed by the Richardson extrapolation based global error estimator. Because the two matrices are of different dimensions, the direct use of the already factored matrix is not possible but an efficient algorithm that makes use of the factored matrix indirectly might be possible.

A modification of COLSYS to replace the current global error estimator with one based on the use of a higher order collocation method or deferred correction approach using a higher order collocation method might lead to significant improvements in the efficiency of the code.

Since both the defect and the global error provide interesting measures of solution quality, the work presented in this paper also suggests that one might implement a practical BVODE code that employs a hybrid defect/global error control strategy. This would mean that, in addition to the low cost computation of the maximum defect estimate, a low cost global error estimator, such as one of the ones we recommend in this paper, would also be computed after every intermediate solution and then a combination of the maximum defect estimate and the global error estimate would be controlled and used to guide the mesh refinement process.

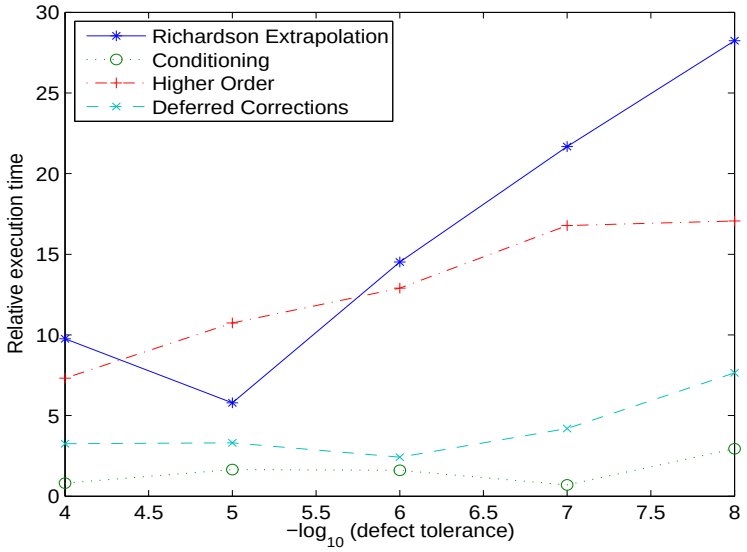

Figure 9: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (20).

References

[1] U. M. Ascher, J. Christiansen, and R. D. Russell. COLSYS - - a collocation code for boundary - value problems. In *Proceedings of a Working Conference on Codes for Boundary-Value Problems in Ordinary Differential Equations*, pages 164–185, London, UK, 1979. Springer-Verlag.

[2] U. M. Ascher, R. M. M. Mattheij, and R. D. Russell. *Numerical solution of boundary value problems for ordinary differential equations*, volume 13 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995. Corrected reprint of the 1988 original.

[3] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1998.

[4] U. M. Ascher and R. J. Spiteri. Collocation software for boundary value differential-algebraic equations. *SIAM J. Sci. Comput.*, 15(4):938–952, 1994.

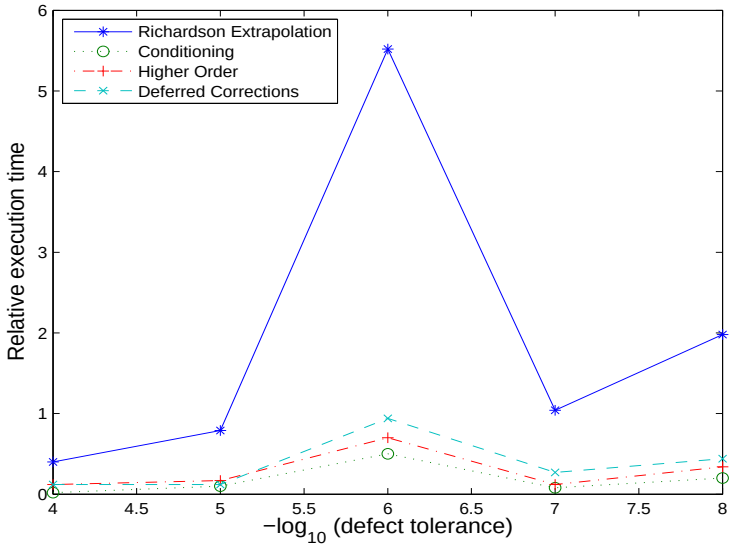

Figure 10: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (22).

[5] G. Bader and U. Ascher. A new basis implementation for a mixed order boundary value ODE solver. *SIAM J. Sci. Statist. Comput.*, 8(4):483–500, 1987.

[6] K. Burrage, F. H. Chipman, and P. H. Muir. Order results for mono-implicit Runge-Kutta methods. *SIAM J. Numer. Anal.*, 31(3):876–891, 1994.

[7] Y. Cao and L. Petzold. A posteriori error estimation and global error control for ordinary differential equations by the adjoint method. *SIAM J. Sci. Comput.*, 26(2):359–374 (electronic), 2004.

[8] S. Capper, J. Cash, and F. Mazzia. On the development of effective algorithms for the numerical solution of singularly perturbed two-point boundary value problems. *Int. J. Comput. Sci. Math.*, 1(1):42–57, 2007.

[9] J. R. Cash and F. Mazzia. Hybrid mesh selection algorithms based on conditioning for two-point boundary value problems. *JNAIAM J. Numer. Anal. Ind. Appl. Math.*, 1(1):81–90, 2006.

[10] J. R. Cash, G. Moore, and R. W. Wright. An automatic continuation strategy for the solution of singularly perturbed linear two-point boundary value problems. *J. Comput. Phys.*, 122(2):266–279, 1995.

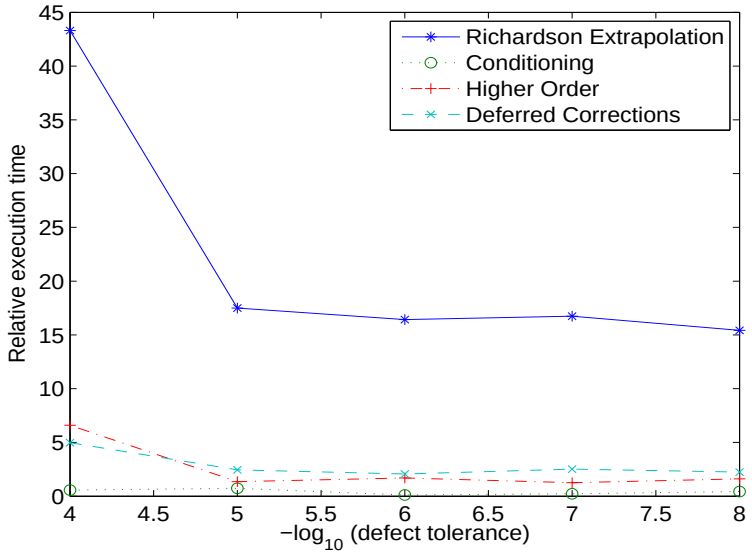

Figure 11: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (21).

[11] 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. Statist. Comput.*, 12(4):971–989, 1991.

[12] W. Enright and L. Yan. The quality/cost trade-off for a class of ode solvers. *Numer. Algorithms*, 2009.

[13] W. H. Enright. Continuous numerical methods for ODEs with defect control. *J. Comput. Appl. Math.*, 125(1-2):159–170, 2000. Numerical analysis 2000, Vol. VI, Ordinary differential equations and integral equations.

[14] W. H. Enright and P. H. Muir. Runge-Kutta software with defect control for boundary value ODEs. *SIAM J. Sci. Comput.*, 17(2):479–497, 1996.

[15] S. Gupta. An adaptive boundary value Runge-Kutta solver for first order boundary value problems. *SIAM J. Numer. Anal.*, 22(1):114–126, 1985.

[16] P. M. Hanson and W. H. Enright. Controlling the defect in existing variable-order Adams codes for initial value problems. *ACM Trans. Math. Software*, 9(1):71–97, 1983.

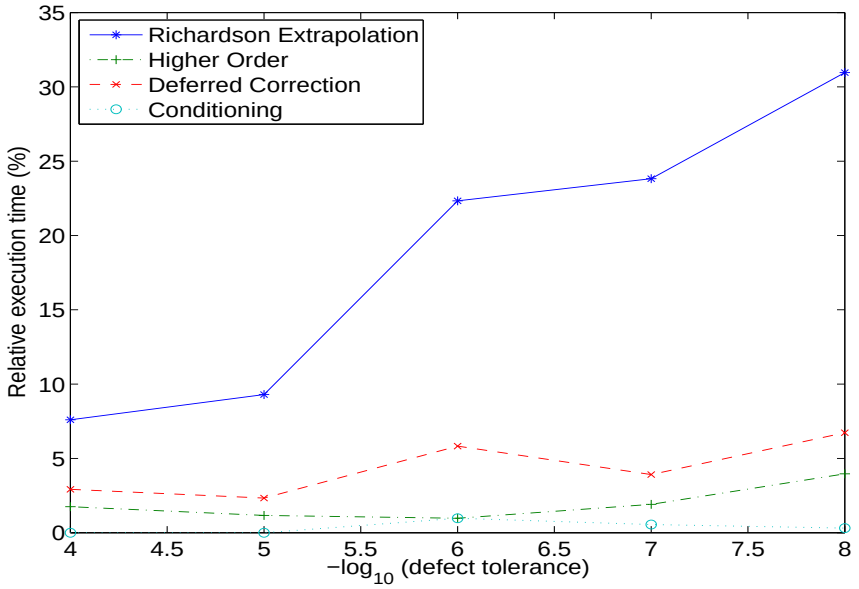

Figure 12: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (19).

[17] N. J. Higham. FORTRAN codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation. *ACM Trans. Math. Software*, 14(4):381–396 (1989), 1988.

[18] J. Kierzenka and L. F. Shampine. A BVP solver based on residual control and the MATLAB PSE. *ACM Trans. Math. Software*, 27(3):299–316, 2001.

[19] J. Kierzenka and L. F. Shampine. A BVP solver that controls residual and error. *JNAIAM J. Numer. Anal. Ind. Appl. Math.*, 3(1-2):27–41, 2008.

[20] J. Lang and J. G. Verwer. On global error estimation and control for initial value problems. *SIAM J. Sci. Comput.*, 29(4):1460–1475 (electronic), 2007.

[21] F. Mazzia and D. Trigiante. A hybrid mesh selection strategy based on conditioning for boundary value ODE problems. *Numer. Algorithms*, 36(2):169–187, 2004.

[22] P. Muir and B. Owren. Order barriers and characterizations for continuous mono-implicit Runge-Kutta schemes. *Math. Comp.*, 61(204):675–699, 1993.

[23] P. H. Muir. Optimal discrete and continuous mono-implicit Runge-Kutta schemes for BVODEs. *Adv. Comput. Math.*, 10(2):135–167, 1999.

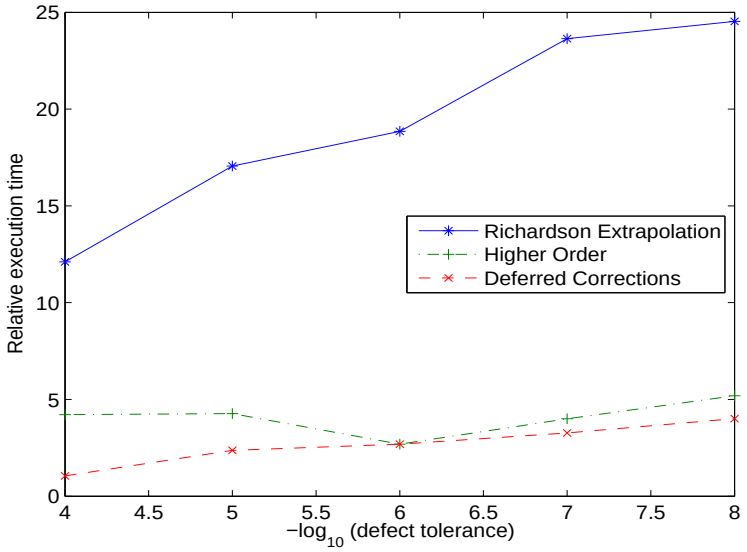

Figure 13: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (23). Costs for the CO approach are negligible and are not included.

[24] L. F. Shampine and P. H. Muir. Estimating conditioning of BVPs for ODEs. *Math. Comput. Modelling*, 40(11-12):1309–1321, 2004.

[25] L. F. Shampine, P. H. Muir, and H. Xu. A user-friendly Fortran BVP solver. *JNAIAM J. Numer. Anal. Ind. Appl. Math.*, 1(2):201–217, 2006.

[26] L. F. Shampine and H. A. Watts. Global error estimation for ordinary differential equations. *ACM Trans. Math. Software*, 2(2):172–186, 1976.

A Appendix: Tables for Orders 2, 4, and 6

See Tables 1, 2, 3, 4, and 5 for results on second order methods applied to problems (19), (20), (21), (22), and (23), respectively. See Tables 6, 7, 8, 9, and 10 for results on fourth order methods applied to problems (19), (20), (21), (22), and (23), respectively. See Tables 11, 12, 13, 14, and 15 for results on sixth order methods applied to problems (19), (20), (21), (22), and (23), respectively.

The entries in each table are organized by columns. The first column is the method used to estimate the global error. The second column (Tol) is the

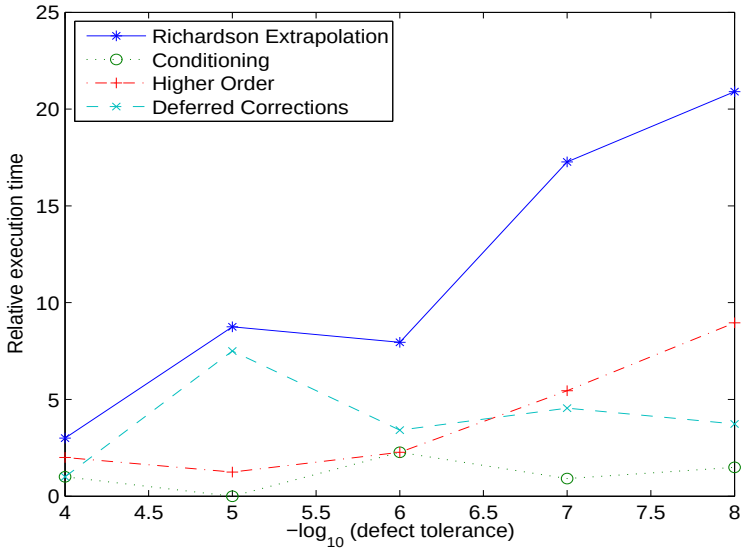

Figure 14: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (20).

tolerance for the defect. The third column (Time) is the run time in seconds for the estimator. The forth column (% Total) is the percent of the total solution time used to determine the global error with that estimation method. The fifth column (Actual) is the actual global error. The sixth column (Estimated) is estimated global error as determined by that estimation method. The seventh column is $\tau = |E - G|$, where E is the estimated global error and G is the actual global error. The last column (BS) is the number of backsolves, over all runs, used to determine the error estimate.

For the CO method the backsolves are associated with estimating the norm of the inverse of the Newton matrix, (6); the right hand sides used in the backsolves are simple vectors generated by the Higham–Tisseur algorithm [17] at relatively little cost. For the RE, HO, and DC methods, the right hand side vectors used in the backsolves are much more expensive evaluations of the residual vector, appearing in (5). From some simple profiling of the computations, we have observed that the cost of a backsolve is only about one sixth of the cost of the evaluation of the residual. Thus even though the CO method employs more backsolves than do the other methods, in some cases, the costs for the CO method are much lower because the costs for generating the right hand sides it uses for the backsolves are negligible.

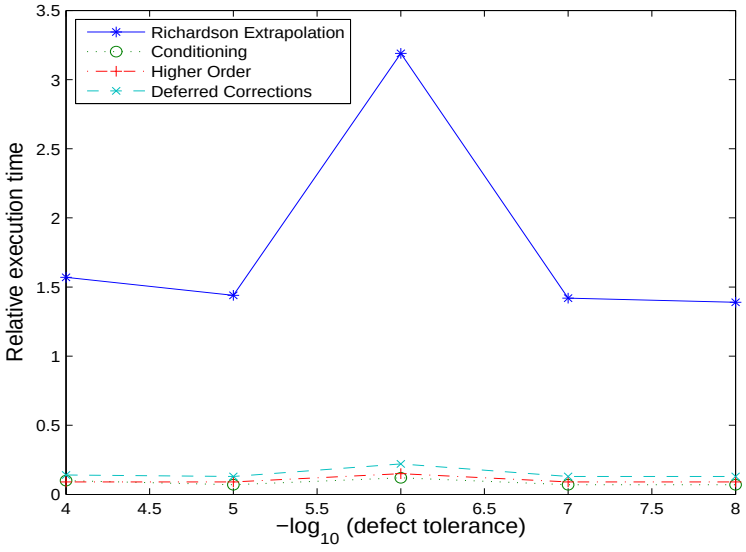

Figure 15: Plot of $-\log_{10}$ of defect tolerance vs. execution time for global error estimator as percentage of primary solution execution time for fourth order for test problem (22).

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	0.285	41.48 %	2.524×10^{-3}	2.524×10^{-3}	1.996×10^{-7}	7
	1×10^{-5}	2.402	42.50 %	1.135×10^{-4}	2.548×10^{-5}	8.800×10^{-5}	8
	1×10^{-6}	1184.113	97.76 %	1.422×10^{-7}	1.380×10^{-7}	4.184×10^{-9}	2
	1×10^{-7}	347.359	87.81 %	4.442×10^{-7}	1.952×10^{-8}	4.247×10^{-7}	4
	1×10^{-8}	2469.672	89.87 %	6.683×10^{-8}	6.684×10^{-8}	1.309×10^{-11}	2
CO	1×10^{-4}	0.012	1.70 %	2.524×10^{-3}	1.441	1.438	5
	1×10^{-5}	0.035	0.62 %	1.135×10^{-4}	3.381×10^{-1}	3.380×10^{-1}	5
	1×10^{-6}	1.133	0.09 %	1.422×10^{-7}	8.990×10^{-2}	8.990×10^{-2}	5
	1×10^{-7}	0.605	0.15 %	4.442×10^{-7}	1.591×10^{-2}	1.591×10^{-2}	5
	1×10^{-8}	1.602	0.06 %	6.683×10^{-8}	5.404×10^{-3}	5.403×10^{-3}	5
HO	1×10^{-4}	0.027	3.98 %	2.524×10^{-3}	2.515×10^{-3}	9.327×10^{-6}	2
	1×10^{-5}	0.449	7.95 %	1.135×10^{-4}	1.129×10^{-4}	5.492×10^{-7}	7
	1×10^{-6}	3.883	0.32 %	1.422×10^{-7}	1.422×10^{-7}	2.149×10^{-11}	2
	1×10^{-7}	4.063	1.03 %	4.442×10^{-7}	4.434×10^{-7}	8.325×10^{-10}	4
	1×10^{-8}	8.156	0.30 %	6.683×10^{-8}	6.685×10^{-8}	1.410×10^{-11}	3
DC	1×10^{-4}	0.035	5.11 %	2.524×10^{-3}	2.514×10^{-3}	1.010×10^{-5}	2
	1×10^{-5}	0.332	5.87 %	1.135×10^{-4}	1.134×10^{-4}	1.028×10^{-7}	5
	1×10^{-6}	4.715	0.39 %	1.422×10^{-7}	1.422×10^{-7}	1.801×10^{-12}	2
	1×10^{-7}	4.910	1.24 %	4.442×10^{-7}	4.451×10^{-7}	8.846×10^{-10}	5
	1×10^{-8}	6.773	0.25 %	6.683×10^{-8}	6.678×10^{-8}	4.782×10^{-11}	2

Table 1: Results for problem (19) for order 2.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	0.539	18.02 %	9.458×10^{-6}	9.466×10^{-6}	8.425×10^{-9}	2
	1×10^{-5}	3.195	31.83 %	1.027×10^{-6}	1.027×10^{-6}	4.388×10^{-10}	2
	1×10^{-6}	42.938	43.17 %	1.116×10^{-7}	1.116×10^{-7}	1.385×10^{-11}	2
	1×10^{-7}	463.250	41.30 %	9.037×10^{-9}	9.037×10^{-9}	6.034×10^{-14}	2
CO	1×10^{-4}	0.023	0.78 %	9.458×10^{-6}	7.088×10^{-1}	7.088×10^{-1}	5
	1×10^{-5}	0.289	2.88 %	1.027×10^{-6}	2.125×10^{-1}	2.125×10^{-1}	5
	1×10^{-6}	0.539	0.54 %	1.116×10^{-7}	7.037×10^{-2}	7.037×10^{-2}	5
	1×10^{-7}	2.453	0.22 %	9.037×10^{-9}	2.249×10^{-2}	2.249×10^{-2}	5
HO	1×10^{-4}	0.555	18.54 %	9.458×10^{-6}	9.460×10^{-6}	2.041×10^{-9}	13
	1×10^{-5}	0.766	7.63 %	1.027×10^{-6}	1.023×10^{-6}	3.586×10^{-9}	7
	1×10^{-6}	4.484	4.51 %	1.116×10^{-7}	1.110×10^{-7}	5.370×10^{-10}	2
	1×10^{-7}	5.133	0.46 %	9.037×10^{-9}	9.050×10^{-9}	1.279×10^{-11}	2
DC	1×10^{-4}	0.102	3.39 %	9.458×10^{-6}	9.512×10^{-6}	5.386×10^{-8}	2
	1×10^{-5}	0.531	5.29 %	1.027×10^{-6}	1.029×10^{-6}	2.571×10^{-9}	2
	1×10^{-6}	1.953	1.96 %	1.116×10^{-7}	1.117×10^{-7}	1.787×10^{-10}	2
	1×10^{-7}	6.430	0.57 %	9.037×10^{-9}	9.034×10^{-9}	3.630×10^{-12}	2

Table 2: Results for problem (20) for order 2.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	10.527	34.20 %	1.560×10^{-6}	1.560×10^{-6}	8.021×10^{-10}	2
	1×10^{-5}	10.555	33.32 %	1.501×10^{-6}	1.501×10^{-6}	5.323×10^{-11}	2
	1×10^{-6}	10.648	32.91 %	1.470×10^{-6}	1.470×10^{-6}	1.601×10^{-12}	2
	1×10^{-7}	110.277	36.92 %	1.350×10^{-7}	1.350×10^{-7}	1.314×10^{-15}	2
	1×10^{-8}	1073.238	29.13 %	1.442×10^{-8}	1.442×10^{-8}	3.262×10^{-17}	2
CO	1×10^{-4}	0.078	0.25 %	1.560×10^{-6}	1.460	1.460	5
	1×10^{-5}	0.078	0.25 %	1.501×10^{-6}	6.632×10^{-1}	6.632×10^{-1}	5
	1×10^{-6}	0.074	0.23 %	1.470×10^{-6}	6.356×10^{-1}	6.356×10^{-1}	5
	1×10^{-7}	0.238	0.08 %	1.350×10^{-7}	3.312×10^{-1}	3.312×10^{-1}	5
	1×10^{-8}	0.785	0.02 %	1.442×10^{-8}	6.485×10^{-2}	6.485×10^{-2}	5
HO	1×10^{-4}	0.234	0.76 %	1.560×10^{-6}	1.560×10^{-6}	8.332×10^{-10}	2
	1×10^{-5}	0.773	2.44 %	1.501×10^{-6}	1.501×10^{-6}	1.044×10^{-11}	7
	1×10^{-6}	0.785	2.43 %	1.470×10^{-6}	1.470×10^{-6}	8.115×10^{-12}	7
	1×10^{-7}	2.551	0.85 %	1.350×10^{-7}	1.350×10^{-7}	7.580×10^{-15}	7
	1×10^{-8}	8.078	0.22 %	1.442×10^{-8}	1.442×10^{-8}	1.935×10^{-16}	7
DC	1×10^{-4}	0.297	0.96 %	1.560×10^{-6}	1.560×10^{-6}	7.403×10^{-10}	2
	1×10^{-5}	0.297	0.94 %	1.501×10^{-6}	1.501×10^{-6}	5.304×10^{-11}	2
	1×10^{-6}	0.293	0.91 %	1.470×10^{-6}	1.470×10^{-6}	2.424×10^{-12}	2
	1×10^{-7}	0.945	0.32 %	1.350×10^{-7}	1.350×10^{-7}	8.035×10^{-15}	2
	1×10^{-8}	2.992	0.08 %	1.442×10^{-8}	1.442×10^{-8}	1.935×10^{-16}	2

Table 3: Results for problem (21) for order 2.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	0.410	19.59 %	1.942×10^{-5}	1.941×10^{-5}	1.564×10^{-8}	2
	1×10^{-5}	1.199	37.53 %	2.868×10^{-6}	2.865×10^{-6}	2.885×10^{-9}	2
	1×10^{-6}	9.613	35.60 %	2.096×10^{-7}	2.096×10^{-7}	6.654×10^{-11}	2
	1×10^{-7}	44.945	40.99 %	2.046×10^{-8}	2.047×10^{-8}	1.011×10^{-11}	2
	1×10^{-8}	416.844	43.59 %	1.992×10^{-9}	1.993×10^{-9}	1.005×10^{-12}	2
CO	1×10^{-4}	0.059	2.80 %	1.942×10^{-5}	7.650×10^{-2}	7.648×10^{-2}	5
	1×10^{-5}	0.137	4.28 %	2.868×10^{-6}	2.762×10^{-2}	2.761×10^{-2}	5
	1×10^{-6}	0.598	2.21 %	2.096×10^{-7}	6.335×10^{-3}	6.334×10^{-3}	5
	1×10^{-7}	1.082	0.99 %	2.046×10^{-8}	1.543×10^{-3}	1.543×10^{-3}	5
	1×10^{-8}	3.480	0.36 %	1.992×10^{-9}	4.767×10^{-4}	4.767×10^{-4}	5
HO	1×10^{-4}	0.238	11.38 %	1.942×10^{-5}	1.941×10^{-5}	1.558×10^{-8}	7
	1×10^{-5}	0.184	5.75 %	2.868×10^{-6}	2.865×10^{-6}	2.883×10^{-9}	2
	1×10^{-6}	2.301	8.52 %	2.096×10^{-7}	2.096×10^{-7}	6.927×10^{-11}	7
	1×10^{-7}	6.516	5.94 %	2.046×10^{-8}	2.047×10^{-8}	1.046×10^{-11}	11
	1×10^{-8}	20.922	2.19 %	1.992×10^{-9}	1.994×10^{-9}	1.058×10^{-12}	11
DC	1×10^{-4}	0.090	4.29 %	1.942×10^{-5}	1.941×10^{-5}	1.569×10^{-8}	2
	1×10^{-5}	0.199	6.23 %	2.868×10^{-6}	2.865×10^{-6}	2.893×10^{-9}	2
	1×10^{-6}	0.844	3.12 %	2.096×10^{-7}	2.096×10^{-7}	7.007×10^{-11}	2
	1×10^{-7}	1.656	1.51 %	2.046×10^{-8}	2.047×10^{-8}	1.002×10^{-11}	2
	1×10^{-8}	5.242	0.55 %	1.992×10^{-9}	1.993×10^{-9}	9.596×10^{-13}	2

Table 4: Results for problem (22) for order 2.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	3.992	9.66 %	4.684×10^{-3}	8.493×10^{-3}	3.809×10^{-3}	3
	1×10^{-5}	25.977	27.87 %	2.860×10^{-4}	8.795×10^{-4}	5.935×10^{-4}	3
	1×10^{-6}	190.117	43.31 %	3.561×10^{-5}	6.968×10^{-5}	3.407×10^{-5}	2
	1×10^{-7}	1639.477	52.11 %	3.954×10^{-6}	7.758×10^{-6}	3.804×10^{-6}	2
CO	1×10^{-4}	0.586	1.42 %	4.684×10^{-3}	2.323×10^{3}	2.323×10^{3}	9
	1×10^{-5}	1.805	1.94 %	2.860×10^{-4}	8.459×10^{2}	8.459×10^{2}	9
	1×10^{-6}	5.430	1.24 %	3.561×10^{-5}	2.220×10^{2}	2.220×10^{2}	9
	1×10^{-7}	16.422	0.52 %	3.954×10^{-6}	6.854×10	6.854×10	9
HO	1×10^{-4}	0.785	1.90 %	4.684×10^{-3}	8.432×10^{-3}	3.748×10^{-3}	5
	1×10^{-5}	1.957	2.10 %	2.860×10^{-4}	8.765×10^{-4}	5.905×10^{-4}	4
	1×10^{-6}	4.594	1.05 %	3.561×10^{-5}	6.964×10^{-5}	3.403×10^{-5}	3
	1×10^{-7}	13.914	0.44 %	3.954×10^{-6}	7.758×10^{-6}	3.803×10^{-6}	3
DC	1×10^{-4}	0.535	1.30 %	4.684×10^{-3}	8.680×10^{-3}	3.995×10^{-3}	3
	1×10^{-5}	1.652	1.77 %	2.860×10^{-4}	8.887×10^{-4}	6.027×10^{-4}	3
	1×10^{-6}	3.746	0.85 %	3.561×10^{-5}	6.980×10^{-5}	3.419×10^{-5}	2
	1×10^{-7}	11.516	0.37 %	3.954×10^{-6}	7.760×10^{-6}	3.805×10^{-6}	2

Table 5: Results for problem (23) for order 2.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	0.191	15.76 %	2.719×10^{-6}	2.589×10^{-5}	2.317×10^{-5}	4
	1×10^{-5}	0.125	18.60 %	1.927×10^{-7}	5.394×10^{-6}	5.202×10^{-6}	4
	1×10^{-6}	0.219	22.22 %	2.244×10^{-7}	3.871×10^{-9}	2.205×10^{-7}	4
	1×10^{-7}	0.297	33.04 %	5.556×10^{-7}	6.217×10^{-7}	6.608×10^{-8}	7
	1×10^{-8}	0.375	34.66 %	8.846×10^{-8}	1.473×10^{-9}	8.698×10^{-8}	6
CO	1×10^{-4}	0.012	0.96 %	2.719×10^{-6}	8.867×10^{-1}	8.867×10^{-1}	5
	1×10^{-5}	0.004	0.58 %	1.927×10^{-7}	5.198×10^{-2}	5.198×10^{-2}	5
	1×10^{-6}	0.008	0.79 %	2.244×10^{-7}	4.890×10^{-3}	4.890×10^{-3}	5
	1×10^{-7}	0.004	0.43 %	5.556×10^{-7}	3.129×10^{-3}	3.128×10^{-3}	5
	1×10^{-8}	0.008	0.72 %	8.846×10^{-8}	6.542×10^{-4}	6.541×10^{-4}	5
HO	1×10^{-4}	0.180	14.79 %	2.719×10^{-6}	5.667×10^{-8}	2.662×10^{-6}	11
	1×10^{-5}	0.066	9.88 %	1.927×10^{-7}	1.827×10^{-7}	9.947×10^{-9}	7
	1×10^{-6}	0.027	2.78 %	2.244×10^{-7}	2.349×10^{-7}	1.051×10^{-8}	2
	1×10^{-7}	0.207	23.04 %	5.556×10^{-7}	5.352×10^{-7}	2.036×10^{-8}	15
	1×10^{-8}	0.203	18.77 %	8.846×10^{-8}	8.885×10^{-8}	3.961×10^{-10}	8
DC	1×10^{-4}	0.031	2.57 %	2.719×10^{-6}	1.586×10^{-7}	2.560×10^{-6}	1
	1×10^{-5}	0.035	5.23 %	1.927×10^{-7}	1.761×10^{-7}	1.657×10^{-8}	2
	1×10^{-6}	0.066	6.75 %	2.244×10^{-7}	2.429×10^{-7}	1.851×10^{-8}	3
	1×10^{-7}	0.055	6.09 %	5.556×10^{-7}	4.907×10^{-7}	6.488×10^{-8}	4
	1×10^{-8}	0.090	8.30 %	8.846×10^{-8}	8.300×10^{-8}	5.454×10^{-9}	4

Table 6: Results for problem (19) for order 4.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	0.094	9.76 %	4.778×10^{-6}	4.784×10^{-6}	6.182×10^{-9}	2
	1×10^{-5}	0.055	5.79 %	5.472×10^{-7}	5.473×10^{-7}	1.380×10^{-10}	2
	1×10^{-6}	0.141	14.52 %	5.968×10^{-8}	5.968×10^{-8}	6.768×10^{-12}	2
	1×10^{-7}	0.242	21.68 %	6.637×10^{-9}	6.637×10^{-9}	4.497×10^{-13}	2
	1×10^{-8}	0.375	28.24 %	1.549×10^{-9}	1.549×10^{-9}	5.487×10^{-14}	2
CO	1×10^{-4}	0.008	0.81 %	4.778×10^{-6}	3.378×10^{-2}	3.378×10^{-2}	5
	1×10^{-5}	0.016	1.65 %	5.472×10^{-7}	6.870×10^{-3}	6.870×10^{-3}	5
	1×10^{-6}	0.016	1.61 %	5.968×10^{-8}	1.144×10^{-3}	1.144×10^{-3}	5
	1×10^{-7}	0.008	0.70 %	6.637×10^{-9}	1.993×10^{-4}	1.993×10^{-4}	5
	1×10^{-8}	0.039	2.94 %	1.549×10^{-9}	4.648×10^{-5}	4.648×10^{-5}	5
HO	1×10^{-4}	0.070	7.32 %	4.778×10^{-6}	4.746×10^{-6}	3.166×10^{-8}	7
	1×10^{-5}	0.102	10.74 %	5.472×10^{-7}	5.422×10^{-7}	5.018×10^{-9}	7
	1×10^{-6}	0.125	12.90 %	5.968×10^{-8}	5.925×10^{-8}	4.268×10^{-10}	7
	1×10^{-7}	0.188	16.78 %	6.637×10^{-9}	6.597×10^{-9}	3.967×10^{-11}	7
	1×10^{-8}	0.227	17.06 %	1.549×10^{-9}	1.545×10^{-9}	3.654×10^{-12}	7
DC	1×10^{-4}	0.031	3.25 %	4.778×10^{-6}	4.807×10^{-6}	2.943×10^{-8}	2
	1×10^{-5}	0.031	3.31 %	5.472×10^{-7}	5.499×10^{-7}	2.697×10^{-9}	2
	1×10^{-6}	0.023	2.42 %	5.968×10^{-8}	5.995×10^{-8}	2.698×10^{-10}	2
	1×10^{-7}	0.047	4.20 %	6.637×10^{-9}	6.666×10^{-9}	2.925×10^{-11}	2
	1×10^{-8}	0.102	7.65 %	1.549×10^{-9}	1.552×10^{-9}	3.327×10^{-12}	2

Table 7: Results for problem (20) for order 4.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	2.863	30.30 %	9.860×10^{-12}	6.933×10^{-12}	2.928×10^{-12}	1
	1×10^{-5}	2.863	29.97 %	4.308×10^{-12}	4.791×10^{-12}	4.830×10^{-13}	1
	1×10^{-6}	2.863	28.97 %	2.459×10^{-12}	2.463×10^{-12}	4.319×10^{-15}	1
	1×10^{-7}	2.840	28.35 %	2.391×10^{-12}	2.390×10^{-12}	5.230×10^{-16}	1
	1×10^{-8}	2.852	28.07 %	2.345×10^{-12}	2.344×10^{-12}	7.846×10^{-16}	1
CO	1×10^{-4}	0.039	0.41 %	9.860×10^{-12}	6.465×10^{-3}	6.465×10^{-3}	5
	1×10^{-5}	0.039	0.41 %	4.308×10^{-12}	2.599×10^{-4}	2.599×10^{-4}	5
	1×10^{-6}	0.039	0.40 %	2.459×10^{-12}	3.296×10^{-6}	3.296×10^{-6}	5
	1×10^{-7}	0.035	0.35 %	2.391×10^{-12}	1.480×10^{-6}	1.480×10^{-6}	5
	1×10^{-8}	0.031	0.31 %	2.345×10^{-12}	1.715×10^{-6}	1.715×10^{-6}	5
HO	1×10^{-4}	0.078	0.83 %	9.860×10^{-12}	6.930×10^{-12}	2.931×10^{-12}	1
	1×10^{-5}	0.082	0.86 %	4.308×10^{-12}	4.791×10^{-12}	4.831×10^{-13}	1
	1×10^{-6}	0.086	0.87 %	2.459×10^{-12}	2.464×10^{-12}	4.430×10^{-15}	1
	1×10^{-7}	0.086	0.86 %	2.391×10^{-12}	2.390×10^{-12}	4.022×10^{-16}	1
	1×10^{-8}	0.082	0.81 %	2.345×10^{-12}	2.344×10^{-12}	6.690×10^{-16}	1
DC	1×10^{-4}	0.133	1.41 %	9.860×10^{-12}	6.973×10^{-12}	2.887×10^{-12}	1
	1×10^{-5}	0.129	1.35 %	4.308×10^{-12}	4.791×10^{-12}	4.834×10^{-13}	1
	1×10^{-6}	0.133	1.34 %	2.459×10^{-12}	2.464×10^{-12}	4.430×10^{-15}	1
	1×10^{-7}	0.137	1.37 %	2.391×10^{-12}	2.390×10^{-12}	4.022×10^{-16}	1
	1×10^{-8}	0.133	1.31 %	2.345×10^{-12}	2.344×10^{-12}	6.690×10^{-16}	1

Table 8: Results for problem (21) for order 4.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	0.082	0.40 %	1.418×10^{-6}	1.368×10^{-6}	5.012×10^{-8}	2
	1×10^{-5}	0.160	0.79 %	3.380×10^{-7}	3.362×10^{-7}	1.865×10^{-9}	2
	1×10^{-6}	1.168	5.52 %	9.527×10^{-8}	9.399×10^{-8}	1.288×10^{-9}	2
	1×10^{-7}	0.195	1.04 %	1.349×10^{-8}	1.344×10^{-8}	5.072×10^{-11}	2
	1×10^{-8}	0.391	1.98 %	7.716×10^{-10}	7.703×10^{-10}	1.280×10^{-12}	2
CO	1×10^{-4}	0.004	0.02 %	1.418×10^{-6}	3.542×10^{-3}	3.540×10^{-3}	5
	1×10^{-5}	0.020	0.10 %	3.380×10^{-7}	5.986×10^{-4}	5.982×10^{-4}	5
	1×10^{-6}	0.105	0.50 %	9.527×10^{-8}	2.171×10^{-3}	2.171×10^{-3}	5
	1×10^{-7}	0.016	0.08 %	1.349×10^{-8}	4.874×10^{-5}	4.873×10^{-5}	5
	1×10^{-8}	0.039	0.20 %	7.716×10^{-10}	9.675×10^{-6}	9.674×10^{-6}	5
HO	1×10^{-4}	0.023	0.12 %	1.418×10^{-6}	1.367×10^{-6}	5.188×10^{-8}	2
	1×10^{-5}	0.035	0.17 %	3.380×10^{-7}	3.360×10^{-7}	2.034×10^{-9}	2
	1×10^{-6}	0.148	0.70 %	9.527×10^{-8}	9.395×10^{-8}	1.319×10^{-9}	2
	1×10^{-7}	0.023	0.12 %	1.349×10^{-8}	1.344×10^{-8}	5.236×10^{-11}	2
	1×10^{-8}	0.066	0.34 %	7.716×10^{-10}	7.703×10^{-10}	1.302×10^{-12}	2
DC	1×10^{-4}	0.023	0.12 %	1.418×10^{-6}	1.368×10^{-6}	5.058×10^{-8}	2
	1×10^{-5}	0.023	0.12 %	3.380×10^{-7}	3.361×10^{-7}	1.943×10^{-9}	2
	1×10^{-6}	0.199	0.94 %	9.527×10^{-8}	9.398×10^{-8}	1.296×10^{-9}	2
	1×10^{-7}	0.051	0.27 %	1.349×10^{-8}	1.344×10^{-8}	5.116×10^{-11}	2
	1×10^{-8}	0.086	0.44 %	7.716×10^{-10}	7.703×10^{-10}	1.286×10^{-12}	2

Table 9: Results for problem (22) for order 4.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	0.156	9.57 %	6.927×10^{-4}	8.084×10^{-4}	1.157×10^{-4}	3
	1×10^{-5}	0.270	14.37 %	3.696×10^{-5}	3.212×10^{-5}	4.840×10^{-6}	2
	1×10^{-6}	0.762	18.11 %	3.642×10^{-5}	8.977×10^{-5}	5.335×10^{-5}	3
	1×10^{-7}	0.816	22.26 %	3.394×10^{-6}	1.070×10^{-5}	7.306×10^{-6}	2
	1×10^{-8}	1.598	25.63 %	4.756×10^{-7}	7.341×10^{-7}	2.585×10^{-7}	2
CO	1×10^{-4}	0.023	1.44 %	6.927×10^{-4}	1.116×10^{2}	1.116×10^{2}	5
	1×10^{-5}	0.059	3.13 %	3.696×10^{-5}	2.717×10	2.717×10	7
	1×10^{-6}	0.086	2.04 %	3.642×10^{-5}	5.674	5.674	5
	1×10^{-7}	0.063	1.70 %	3.394×10^{-6}	9.280×10^{-1}	9.280×10^{-1}	5
	1×10^{-8}	0.215	3.45 %	4.756×10^{-7}	9.158×10^{-2}	9.158×10^{-2}	7
HO	1×10^{-4}	0.047	2.87 %	6.927×10^{-4}	7.423×10^{-4}	4.967×10^{-5}	4
	1×10^{-5}	0.063	3.33 %	3.696×10^{-5}	3.282×10^{-5}	4.137×10^{-6}	3
	1×10^{-6}	0.148	3.53 %	3.642×10^{-5}	9.214×10^{-5}	5.573×10^{-5}	3
	1×10^{-7}	0.160	4.37 %	3.394×10^{-6}	1.077×10^{-5}	7.377×10^{-6}	3
	1×10^{-8}	0.289	4.64 %	4.756×10^{-7}	7.360×10^{-7}	2.604×10^{-7}	3
DC	1×10^{-4}	0.031	1.91 %	6.927×10^{-4}	7.252×10^{-4}	3.256×10^{-5}	3
	1×10^{-5}	0.047	2.50 %	3.696×10^{-5}	3.264×10^{-5}	4.323×10^{-6}	2
	1×10^{-6}	0.176	4.18 %	3.642×10^{-5}	9.160×10^{-5}	5.518×10^{-5}	3
	1×10^{-7}	0.125	3.41 %	3.394×10^{-6}	1.075×10^{-5}	7.357×10^{-6}	2
	1×10^{-8}	0.223	3.57 %	4.756×10^{-7}	7.353×10^{-7}	2.598×10^{-7}	2

Table 10: Results for problem (23) for order 4.

Method	Tol	Time	% Total Time	Actual Error	Estimated Error	τ	BS
RE	1×10^{-4}	0.051	7.60 %	2.524×10^{-7}	4.595×10^{-7}	2.072×10^{-7}	1
	1×10^{-5}	0.031	9.30 %	1.231×10^{-6}	5.396×10^{-7}	6.910×10^{-7}	4
	1×10^{-6}	0.090	22.33 %	2.544×10^{-8}	1.077×10^{-8}	1.466×10^{-8}	8
	1×10^{-7}	1.023	23.82 %	3.259×10^{-11}	2.193×10^{-11}	1.066×10^{-11}	1
	1×10^{-8}	2.715	30.96 %	1.567×10^{-10}	1.317×10^{-10}	2.493×10^{-11}	2
CO	1×10^{-4}	0.000	0.00 %	2.524×10^{-7}	2.131×10^{-1}	2.131×10^{-1}	5
	1×10^{-5}	0.000	0.00 %	1.231×10^{-6}	4.306×10^{-2}	4.306×10^{-2}	5
	1×10^{-6}	0.004	0.97 %	2.544×10^{-8}	3.785×10^{-3}	3.785×10^{-3}	5
	1×10^{-7}	0.023	0.55 %	3.259×10^{-11}	8.765×10^{-4}	8.765×10^{-4}	5
	1×10^{-8}	0.027	0.31 %	1.567×10^{-10}	1.227×10^{-4}	1.227×10^{-4}	5
HO	1×10^{-4}	0.012	1.75 %	2.524×10^{-7}	1.798×10^{-7}	7.260×10^{-8}	1
	1×10^{-5}	0.004	1.16 %	1.231×10^{-6}	1.288×10^{-6}	5.782×10^{-8}	2
	1×10^{-6}	0.004	0.97 %	2.544×10^{-8}	2.620×10^{-8}	7.599×10^{-10}	2
	1×10^{-7}	0.082	1.91 %	3.259×10^{-11}	1.825×10^{-11}	1.434×10^{-11}	1
	1×10^{-8}	0.348	3.96 %	1.567×10^{-10}	8.405×10^{-11}	7.263×10^{-11}	2
DC	1×10^{-4}	0.020	2.92 %	2.524×10^{-7}	3.500×10^{-7}	9.763×10^{-8}	1
	1×10^{-5}	0.008	2.33 %	1.231×10^{-6}	1.266×10^{-6}	3.581×10^{-8}	2
	1×10^{-6}	0.023	5.83 %	2.544×10^{-8}	4.716×10^{-8}	2.172×10^{-8}	4
	1×10^{-7}	0.168	3.91 %	3.259×10^{-11}	3.324×10^{-11}	6.450×10^{-13}	1
	1×10^{-8}	0.590	6.73 %	1.567×10^{-10}	2.044×10^{-10}	4.775×10^{-11}	3

Table 11: Results for problem (19) for order 6.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	0.023	3.00 %	1.540×10^{-6}	1.542×10^{-6}	2.045×10^{-9}	2
	1×10^{-5}	0.055	8.75 %	2.481×10^{-7}	2.482×10^{-7}	8.826×10^{-11}	2
	1×10^{-6}	0.055	7.95 %	2.306×10^{-8}	2.306×10^{-8}	4.643×10^{-12}	2
	1×10^{-7}	0.148	17.27 %	2.497×10^{-9}	2.497×10^{-9}	1.841×10^{-13}	2
	1×10^{-8}	0.219	20.90 %	2.732×10^{-10}	2.732×10^{-10}	1.452×10^{-14}	2
CO	1×10^{-4}	0.008	1.00 %	1.540×10^{-6}	1.521×10^{-2}	1.521×10^{-2}	5
	1×10^{-5}	0.000	0.00 %	2.481×10^{-7}	4.641×10^{-3}	4.641×10^{-3}	5
	1×10^{-6}	0.016	2.27 %	2.306×10^{-8}	1.990×10^{-4}	1.990×10^{-4}	5
	1×10^{-7}	0.008	0.91 %	2.497×10^{-9}	3.234×10^{-5}	3.234×10^{-5}	5
	1×10^{-8}	0.016	1.49 %	2.732×10^{-10}	4.365×10^{-6}	4.365×10^{-6}	5
HO	1×10^{-4}	0.016	2.00 %	1.540×10^{-6}	1.566×10^{-6}	2.647×10^{-8}	2
	1×10^{-5}	0.008	1.25 %	2.481×10^{-7}	2.406×10^{-7}	7.555×10^{-9}	2
	1×10^{-6}	0.016	2.27 %	2.306×10^{-8}	2.292×10^{-8}	1.400×10^{-10}	2
	1×10^{-7}	0.047	5.45 %	2.497×10^{-9}	2.488×10^{-9}	9.209×10^{-12}	2
	1×10^{-8}	0.094	8.96 %	2.732×10^{-10}	2.732×10^{-10}	3.438×10^{-14}	2
DC	1×10^{-4}	0.008	1.00 %	1.540×10^{-6}	1.577×10^{-6}	3.709×10^{-8}	2
	1×10^{-5}	0.047	7.50 %	2.481×10^{-7}	2.476×10^{-7}	5.587×10^{-10}	2
	1×10^{-6}	0.023	3.41 %	2.306×10^{-8}	2.305×10^{-8}	5.570×10^{-13}	2
	1×10^{-7}	0.039	4.55 %	2.497×10^{-9}	2.502×10^{-9}	5.637×10^{-12}	2
	1×10^{-8}	0.039	3.73 %	2.732×10^{-10}	2.732×10^{-10}	1.382×10^{-15}	2

Table 12: Results for problem (20) for order 6.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	1.770	43.31 %	7.134×10^{-5}	7.155×10^{-5}	2.090×10^{-7}	2
	1×10^{-5}	0.559	17.50 %	2.726×10^{-14}	5.370×10^{-16}	2.672×10^{-14}	1
	1×10^{-6}	0.527	16.42 %	6.336×10^{-16}	5.900×10^{-16}	4.363×10^{-17}	1
	1×10^{-7}	0.570	16.74 %	1.279×10^{-15}	4.404×10^{-16}	8.389×10^{-16}	1
	1×10^{-8}	0.563	15.42 %	1.335×10^{-15}	3.929×10^{-16}	9.422×10^{-16}	1
CO	1×10^{-4}	0.023	0.57 %	7.134×10^{-5}	5.899×10^{-1}	5.898×10^{-1}	5
	1×10^{-5}	0.023	0.73 %	2.726×10^{-14}	6.771×10^{-7}	6.771×10^{-7}	5
	1×10^{-6}	0.004	0.12 %	6.336×10^{-16}	1.409×10^{-8}	1.409×10^{-8}	5
	1×10^{-7}	0.008	0.23 %	1.279×10^{-15}	5.513×10^{-10}	5.513×10^{-10}	5
	1×10^{-8}	0.016	0.43 %	1.335×10^{-15}	1.440×10^{-6}	1.440×10^{-6}	5
HO	1×10^{-4}	0.270	6.60 %	7.134×10^{-5}	7.325×10^{-5}	1.912×10^{-6}	3
	1×10^{-5}	0.043	1.35 %	2.726×10^{-14}	5.286×10^{-16}	2.673×10^{-14}	1
	1×10^{-6}	0.055	1.70 %	6.336×10^{-16}	5.808×10^{-16}	5.285×10^{-17}	1
	1×10^{-7}	0.043	1.26 %	1.279×10^{-15}	4.387×10^{-16}	8.407×10^{-16}	1
	1×10^{-8}	0.059	1.61 %	1.335×10^{-15}	4.026×10^{-16}	9.325×10^{-16}	1
DC	1×10^{-4}	0.203	4.97 %	7.134×10^{-5}	7.458×10^{-5}	3.240×10^{-6}	2
	1×10^{-5}	0.078	2.45 %	2.726×10^{-14}	6.288×10^{-16}	2.663×10^{-14}	1
	1×10^{-6}	0.066	2.07 %	6.336×10^{-16}	6.856×10^{-16}	5.196×10^{-17}	1
	1×10^{-7}	0.086	2.52 %	1.279×10^{-15}	5.211×10^{-16}	7.583×10^{-16}	1
	1×10^{-8}	0.082	2.25 %	1.335×10^{-15}	5.135×10^{-16}	8.216×10^{-16}	1

Table 13: Results for problem (21) for order 6.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	5.383	1.57 %	2.347×10^{-9}	1.757×10^{-9}	5.899×10^{-10}	1
	1×10^{-5}	4.973	1.44 %	2.661×10^{-9}	2.170×10^{-9}	4.912×10^{-10}	1
	1×10^{-6}	11.699	3.19 %	4.065×10^{-9}	4.075×10^{-9}	1.044×10^{-11}	1
	1×10^{-7}	4.988	1.42 %	7.920×10^{-13}	8.189×10^{-13}	2.694×10^{-14}	1
	1×10^{-8}	4.953	1.39 %	6.255×10^{-15}	3.019×10^{-15}	3.236×10^{-15}	1
CO	1×10^{-4}	0.359	0.10 %	2.347×10^{-9}	9.758×10^{-2}	9.758×10^{-2}	7
	1×10^{-5}	0.246	0.07 %	2.661×10^{-9}	1.538×10^{-2}	1.538×10^{-2}	5
	1×10^{-6}	0.449	0.12 %	4.065×10^{-9}	5.676×10^{-3}	5.676×10^{-3}	5
	1×10^{-7}	0.238	0.07 %	7.920×10^{-13}	1.240×10^{-4}	1.240×10^{-4}	5
	1×10^{-8}	0.234	0.07 %	6.255×10^{-15}	1.188×10^{-5}	1.188×10^{-5}	5
HO	1×10^{-4}	0.316	0.09 %	2.347×10^{-9}	1.270×10^{-9}	1.077×10^{-9}	1
	1×10^{-5}	0.305	0.09 %	2.661×10^{-9}	1.865×10^{-9}	7.958×10^{-10}	1
	1×10^{-6}	0.551	0.15 %	4.065×10^{-9}	4.052×10^{-9}	1.286×10^{-11}	1
	1×10^{-7}	0.313	0.09 %	7.920×10^{-13}	8.017×10^{-13}	9.674×10^{-15}	1
	1×10^{-8}	0.309	0.09 %	6.255×10^{-15}	3.012×10^{-15}	3.244×10^{-15}	1
DC	1×10^{-4}	0.480	0.14 %	2.347×10^{-9}	1.269×10^{-9}	1.078×10^{-9}	1
	1×10^{-5}	0.457	0.13 %	2.661×10^{-9}	1.865×10^{-9}	7.958×10^{-10}	1
	1×10^{-6}	0.813	0.22 %	4.065×10^{-9}	4.052×10^{-9}	1.280×10^{-11}	1
	1×10^{-7}	0.461	0.13 %	7.920×10^{-13}	8.018×10^{-13}	9.851×10^{-15}	1
	1×10^{-8}	0.453	0.13 %	6.255×10^{-15}	3.029×10^{-15}	3.226×10^{-15}	1

Table 14: Results for problem (22) for order 6.

	Tol	Time	% Total	Actual	Estimated	τ	BS
RE	1×10^{-4}	0.090	12.11 %	1.602×10^{-3}	6.227×10^{-4}	9.790×10^{-4}	2
	1×10^{-5}	0.141	17.06 %	1.300×10^{-4}	1.530×10^{-5}	1.147×10^{-4}	2
	1×10^{-6}	0.191	18.85 %	1.689×10^{-5}	9.061×10^{-7}	1.598×10^{-5}	2
	1×10^{-7}	0.254	23.64 %	3.316×10^{-6}	3.393×10^{-7}	2.977×10^{-6}	2
	1×10^{-8}	0.406	24.53 %	2.588×10^{-7}	9.734×10^{-8}	1.614×10^{-7}	2
CO	1×10^{-4}	0.020	2.63 %	1.602×10^{-3}	2.260×10	2.260×10	5
	1×10^{-5}	0.020	2.37 %	1.300×10^{-4}	2.368	2.368	7
	1×10^{-6}	0.016	1.54 %	1.689×10^{-5}	1.071	1.071	5
	1×10^{-7}	0.020	1.82 %	3.316×10^{-6}	2.079×10^{-1}	2.079×10^{-1}	5
	1×10^{-8}	0.031	1.89 %	2.588×10^{-7}	4.197×10^{-2}	4.197×10^{-2}	5
HO	1×10^{-4}	0.031	4.21 %	1.602×10^{-3}	7.201×10^{-4}	8.816×10^{-4}	4
	1×10^{-5}	0.035	4.27 %	1.300×10^{-4}	1.389×10^{-5}	1.161×10^{-4}	3
	1×10^{-6}	0.027	2.69 %	1.689×10^{-5}	9.121×10^{-7}	1.597×10^{-5}	2
	1×10^{-7}	0.043	4.00 %	3.316×10^{-6}	3.388×10^{-7}	2.977×10^{-6}	2
	1×10^{-8}	0.086	5.19 %	2.588×10^{-7}	9.642×10^{-8}	1.624×10^{-7}	3
DC	1×10^{-4}	0.008	1.05 %	1.602×10^{-3}	7.060×10^{-4}	8.957×10^{-4}	2
	1×10^{-5}	0.020	2.37 %	1.300×10^{-4}	1.379×10^{-5}	1.162×10^{-4}	2
	1×10^{-6}	0.027	2.69 %	1.689×10^{-5}	9.090×10^{-7}	1.598×10^{-5}	2
	1×10^{-7}	0.035	3.27 %	3.316×10^{-6}	3.384×10^{-7}	2.977×10^{-6}	2
	1×10^{-8}	0.066	4.01 %	2.588×10^{-7}	9.627×10^{-8}	1.625×10^{-7}	2

Table 15: Results for problem (23) for order 6.