A PREDICTED SEQUENTIAL REGULARIZATION METHOD FOR INDEX-2 HESSENBERG DAES

PING LIN* AND RAYMOND J. SPITERI†

Abstract. The sequential regularization method (SRM) is a dynamic iterative method for the numerical solution of higher-index differential-algebraic equations (DAEs). The SRM has the advantage that it is based on a regularized problem that is less stiff than those produced by standard regularization methods. Consequently, non-stiff integrators may be used, making the SRM a competitive alternative to popular integrators. In past work, the number of SRM iterations has been taken to be roughly equal to the order of the numerical method used in each dynamic iteration. In this paper, we propose a *predicted* SRM (PSRM) that reduces the number of iterations in each dynamic iteration to one. We give a new error analysis for explicit Runge-Kutta methods applied to linear index-2 Hessenberg DAEs with or without singularities. We also give numerical examples to confirm the predicted convergence rates. For the PSRM, extrapolation formulas and methods based on the differential part of the differential-algebraic equations serve as a predictor, and the SRM iteration serves as a corrector. Implementation of higher-order schemes for the PSRM makes use of continuous extensions of Runge-Kutta methods. In particular, we give a prediction scheme for the algebraic variable at intermediate stage points that suppresses order reduction in the differential variable near a singularity. Moreover, the SRM/PSRM provides new insight on operator splitting and fast convergence rates for waveform relaxation.

Keywords. Differential-algebraic equations, singular perturbation, regularization, waveform relaxation, Runge-Kutta methods, higher index.
AMS(MOS) subject classifications. 65L10, 65L20

1. Introduction. Differential-algebraic equations (DAEs) arise in a variety of applications. They can be difficult to solve when they have an index greater than one; cf. [13]. Higher-index DAEs are ill-posed in a certain sense, especially when the index is greater than two, and a direct discretization generally does not work well. A popular solution strategy is the use of index-reduction methods, which involve the repeated differentiation of the constraint equations until a well-posed problem (index-1 DAE or ordinary differential equation (ODE)) is obtained. However, it is well known that repeated index reduction by direct differentiation leads to instability for numerical integrations (i.e., the drift problem, where the error in satisfying the constraint accumulates with time). A stabilization formulation was first introduced by Baumgarte [10] for the simulation of constrained multi-body systems. This technique was subsequently improved by many others (cf. [13, 19, 8] and references therein). An alternative approach is the *regularization* of DAEs, where a perturbation term (measured by a small positive regularization parameter ϵ) is added to the original DAE (see, e.g., [20, 24, 23, 29, 33]). The regularized problem is usually a singular perturbation problem, and the DAE becomes the reduced singular perturbation problem. In this case a stiff solver is typically needed to solve the regularized problem, even if the original DAE is not stiff. Recently the *sequential regularization method* (SRM) was proposed in [4, 5] for both initial-value and boundary-value problems of linear and nonlinear higher-index DAEs. It combines Baumgarte's stabilization (or modified version thereof) with a modified penalty method. It also relates to and improves an augmented Lagrangian formulation in the context of optimization used in [11] for the simulation of constrained mechanical systems. However, unlike usual regularization, the regularization parameter ϵ need not be very small. Therefore, no extra stiffness is introduced, and more computationally efficient explicit methods can be used to solve the regularized problem. Further results on the application of the SRM to partial differential equations with constraints such as Navier-Stokes equations and miscible displacement in porous media can be found in [27, 28]; an SRM algorithm with linear computational complexity for large mechanical chains with many closed loops is described in [6].

*Department of Mathematics, The National University of Singapore, Singapore 119260 (e-mail: matlinp@math.nus.edu.sg, tel.: 65-8742488, fax: 65-7795452). The work of this author was partially supported under Singapore academic research grant RP 3991619.

†Department of Computer Science, Dalhousie University, Halifax, NS, Canada B3H 1W5 (e-mail: spiteri@cs.dal.ca, tel.: 902-585-1556, fax: 902-492-1517). The work of this author was partially supported under a research grant from Canada's National Science and Engineering Council.

In [4, 5], it has been pointed out that the SRM can be more suitable for problems with a certain class of constraint singularities[1] than stabilization formulations because SRM requires few if any constraint differentiations. Other theoretical and practical advantages of the SRM can be found in [6, 27, 28]. However, the SRM can be somewhat less efficient than usual stabilization formulations because of the required iterations. This is the issue addressed in this paper. We show how one formulation of the SRM avoids solving a linear system at each time step, making the expense of the SRM iteration no worse (and indeed often better) than usual stabilization formulations. For a more robust version of the SRM that requires the solution of a linear system, we propose a predicted SRM (PSRM) that generally requires only one iteration. Hence the computational expense of the PSRM is competitive with other stabilization formulations.

The PSRM starts by producing an initial guess for the algebraic component of the solution. The more accurate the initial guess is, the fewer the number of SRM iterations required to achieve a given overall accuracy. Hence the design of good prediction algorithms is important to reduce the number of iterations. Extrapolation formulas and methods based on the differential part of the DAEs can be used as effective predictors. Extrapolation formulas work only for a modified SRM formulation with a different computing scheme for algebraic components, whereas those based on differential part of the DAE work for both SRM and modified SRM formulations.

The role of higher-order explicit schemes for PSRM is another important issue. We mainly consider Runge-Kutta methods in this paper; however, multi-step methods can be considered similarly. Although the SRM formulation is not as stiff as that for typical regularization methods, the well-known order reduction of Runge-Kutta methods is still present. We study the Dormand-Prince 5(4) Runge-Kutta pair and find only a small order reduction because the regularization parameter can be chosen to be relatively large. Its numerical performance is therefore quite satisfactory. The issue of prediction of the algebraic variable at intermediate stages is considered for the first time; in previous work, no intermediate stage calculations were required. In this paper we propose a high-order prediction scheme for the algebraic variable at an intermediate stage that leads to no order reduction for the differential variable near a singularity.

The SRM is a dynamic iterative method. We will show that it can be interpreted as a special type of waveform relaxation (WR) algorithm. Thus, implementational techniques for WR algorithms, such as the continuous extension of Runge-Kutta methods and adaptive step-size control, can be applied directly to the SRM. The author of [30] has studied WR algorithms for linear, constant-coefficient DAEs and provides some guidance on how to do waveform splitting for DAEs. The SRM is an example of a WR algorithm that accelerates the convergence rate and widens the choices of waveform splitting for DAEs or other differential equations. This may also imply more profound results about the DAE waveform splitting than the theorem given in [30].

In practice DAEs often take the form

$$x^{(\nu)} = f(x, x', \ldots, x^{(\nu-1)}, t) - B(x,t)y, \tag{1.1a}$$

$$0 = g(x,t), \tag{1.1b}$$

which has index $\nu + 1$ if GB is nonsingular for all t, $0 \le t \le t_f$, where $G = g_x$. For example, the Euler-Lagrange equations for mechanical systems with holonomic constraints are of this form with $\nu = 2$, and many other applied problems exist with $\nu = 1$. In this paper we concentrate on the index-2 case. For higher-index problems, we can proceed in a similar fashion to [5], or we can assume that a stable reduction to index 2 has been performed (cf. [8, 13, 19]). In §2, we briefly review the SRM and its convergence properties. In §3, we give a detailed error analysis of the PSRM for linear problems with or without singularities, and use examples to lend support to the applicability of our result to nonlinear problems. In §4, we discuss the implementation of the PSRM. Finally, in §5, we describe the implementation of some high-order explicit Runge-Kutta methods for the PSRM formulation and elucidate its relationship to the WR algorithm.

[1] Namely, problems with a highly ill-conditioned constraint Jacobian.

2. The Sequential Regularization Method (SRM). We consider (1.1) with $\nu = 1$:

$$x' = f(x,t) - B(x,t)y, \tag{2.1a}$$
$$0 = g(x,t), \tag{2.1b}$$

where f, B, and g are sufficiently smooth functions of $(x,t) \in \mathbf{R}^{n_x} \times [0, t_f]$ and $y \in \mathbf{R}^{n_y}$, subject to consistent initial conditions

$$x(0) = \beta. \tag{2.2}$$

For convenience, we recall the SRM formulation given in [5]. Application of the modified penalty regularization gives

$$g(x,t) = \epsilon T^{-1}(y - y^0),$$

where y^0 is an initial guess for the solution y and the matrix $T(x,t)$ is chosen such that the resulting system for x is stable. Some choices for T will be mentioned later. Iterating with this, we obtain the following SRM formulation for (2.1): for $\sigma = 1, 2, \ldots,$

$$(x^\sigma)' = f(x^\sigma, t) - B(x^\sigma, t)y^\sigma, \tag{2.3}$$

where

$$y^\sigma = y^{\sigma-1} + \frac{1}{\epsilon}T(x^\sigma, t)g(x^\sigma, t), \tag{2.4}$$

subject to the initial conditions (2.2). We first quote the convergence result for this method from [5]. It can be shown that (cf. perturbation index in [19]):

$$\|\hat{x}(t) - x(t)\| \le C(\|\delta(t)\| + \|\delta'(t)\|), \tag{2.5a}$$
$$\|\hat{y}(t) - y(t)\| \le C(\|\delta(t)\| + \|\delta'(t)\|), \tag{2.5b}$$

where $\|\cdot\|$ is a suitable norm (e.g., maximum norm), C is a generic constant, and $\hat{x}$ and $\hat{y}$ satisfy the following perturbed version of (2.1):

$$\hat{x}' = f(\hat{x}, t) - B(\hat{x}, t)\hat{y}, \tag{2.6a}$$
$$0 = g(\hat{x}, t) + \delta(t), \tag{2.6b}$$

with the same initial conditions as (2.2). For the linear case, assuming that the essential underlying ODE (EUODE)[2] is stable, (2.5a) can be replaced by (cf. [4]):

$$\|\hat{x}(t) - x(t)\| \le C\|\delta(t)\|. \tag{2.7}$$

Using (2.5), we have the following theorem [5]:

THEOREM 2.1. *Let all functions in the DAE (2.1) be sufficiently smooth and GB be nonsingular, and let*

$$y^0(0) = y(0),\ (y^0)'(0) = y'(0), \ldots, (y^0)^{(m)}(0) = y^{(m)}(0). \tag{2.8}$$

In addition, suppose that the matrix function $T(x,t)$ has been chosen so that GBT is positive definite with eigenvalues uniformly bounded away from 0 for $0 \le t \le t_f$. Then, for the iteration (2.3)-(2.4) we have the following error estimates:

$$x^\sigma(t) - x(t) = O(\epsilon^\sigma), \tag{2.9a}$$
$$y^\sigma(t) - y(t) = O(\epsilon^\sigma), \tag{2.9b}$$

[2] defined in the next section

for $1 \le \sigma \le m+1$ and $0 \le t \le t_f$. For $\sigma > m+1$, the error estimate (2.9) is valid away from an initial layer region near $t=0$ of size $O(\epsilon)$. We can choose $T = (GB)^{-1}$, $T = (GB)^T$, or if $G = B^T$ (as is true in many applications), we can even choose $T = I$, the identity matrix. Similar results hold for linear DAEs even when the system has certain type of singularities (cf. [4]).

If we apply a p-th order discretization to the regularized problem (2.3)-(2.4), then the error of the approximate solution of the regularized problem with respect to the true solution of (2.1) is expected to be $O_\epsilon(h^p) + O(\epsilon^\sigma)$. Here $O_\epsilon(h^p) = C_\epsilon h^p$, where C_ϵ is a constant that may depend on ϵ. So if $\epsilon = O(h)$, roughly p SRM iterations are needed in order for the error to reach the numerical accuracy $O_\epsilon(h^p)$. We note that if we take $T = (GB)^T$ (or $T = I$ when $G = B^T$), the SRM (2.3)-(2.4) is explicit, in direct contrast to stabilization formulations and implicit methods. Moreover, stabilization formulations generally require an iterative solution of linear systems, and the number of iterations required would generally be larger than p. Thus, in these cases, the cost of the SRM may even be less than that of stabilization formulations. However, the choice $T = (GB)^{-1}$ is generally more robust because in this case the eigenvalues of GBT are all unity, independently of x and t. With this choice of T, a linear system must be solved at each iteration. The main goal of the paper is to propose, for $T = (GB)^{-1}$, a *predicted* SRM to reduce the number of iterations required from p to one. Hence, the PSRM should be competitive with stabilization formulations for $T = (GB)^{-1}$. We thus let $T = (GB)^{-1}$ in what follows and give a corollary to Theorem 2.1.

COROLLARY 2.1. *Let the conditions of Theorem 2.1 hold. If*

$$y^0(t) - y(t) = \delta e(t),$$

where $e(t)$ is a smooth function, then

$$x^\sigma(t) - x(t) = O(\delta\epsilon^\sigma), \tag{2.10a}$$

$$y^\sigma(t) - y(t) = O(\delta\epsilon^\sigma). \tag{2.10b}$$

The proof follows that of Theorem 2.1.

This corollary implies that if we can choose a better initial guess y^0, e.g., $y^0 - y(t) = O(h^{p_0})$ where $p_0 < p$ (the order of the discretization), then the number of SRM iterations required is reduced to about $p - p_0$. This motivates our predicted sequential regularization method.

3. Analysis of the Predicted SRM. Suppose that we discretize the regularized problem (2.3)-(2.4) by an explicit method of order p. If we can predict the initial guess y^0 such that

$$y^0(t_j) - y(t_j) = O(h^{p_0}), \tag{3.1}$$

where t_j is either a mesh point or an internal stage point, then we expect one SRM iteration will achieve an accuracy $O(\epsilon h^{p_0})$.

In this section, we provide a proof of this expectation. The regularized problem is a singular singularly perturbed problem, and not much analysis is available. In [2], implicit Runge-Kutta schemes are considered that are equivalent to collocation based on Gauss or Lobatto points for linear, singular singularly perturbed problems on a special exponential mesh. We are interested in explicit schemes and give a detailed analysis below on a general mesh. We will also allow certain singularities (as discussed in [4]) to be defined shortly. The linear case of (2.1) reads:

$$x' = \bar{A}(t)x - \bar{B}(t)y + \bar{q}(t), \tag{3.2a}$$

$$0 = \bar{G}(t)x + \bar{r}(t), \tag{3.2b}$$

where we assume matrices $\bar{A}(t)$, $\bar{B}(t)$, and $\bar{G}(t)$ are bounded functions of t. In this paper, we consider only one-step discretizations, although multi-step methods can be treated similarly. Moreover, numerical examples given in the next section indicate that the result obtained also applies to nonlinear problems.

As in [4], we allow $(\bar{G}\bar{B})(t)$ to be singular at some point, say t_*, with the following two conditions:

ASSUMPTION 3.1.

(i) *The matrix function* $P(t) = (\bar{B}(\bar{G}\bar{B})^{-1}\bar{G})(t)$ *is smooth, or more precisely,* $P(t)$ *is continuous and* $P'(t)$ *is bounded near any singular point* t_**, where we define*

$$P(t_*) = \lim_{t \to t_*} (\bar{B}(\bar{G}\bar{B})^{-1}\bar{G})(t).$$

(ii) *The inhomogeneity* $\bar{r}(t)$ *satisfies* $\bar{r} \in \mathcal{S}$*, where*

$$\mathcal{S} = \{w(t) \in \mathbf{R}^{n_y} : \textit{ there exists a smooth function } z(t) \textit{ such that } Gz = w\}.$$

This assumption is motivated from certain singular configurations of mechanical multi-body systems [4, 1]. We are interested in the case where the solution $x(t)$ is smooth. The two parts of the assumption are sufficient to guarantee this (see Lemma 2.1 in [4]). Also, the boundedness of $P'(t)$ is generally needed for the stability of the regularized problem (see Lemma 3.1 in [4]). Although x is assumed to be smooth, the variable y may still be unbounded near the singular point t_*. Nevertheless, the quantity $(\bar{B}y)(t)$ should be smooth from the equation (3.2a). We also note that Assumption 3.1 is automatically satisfied if $\bar{G}\bar{B}$ is nonsingular for all t.

One SRM iteration applied to (3.2) yields

$$(x^1)' = \bar{f}(t, x^1, y^0, \bar{r}, \bar{q}), \tag{3.3a}$$

$$x^1(0) = \beta, \tag{3.3b}$$

where

$$\bar{f}(t, x^1, y^0, \bar{r}, \bar{q}) = -\frac{1}{\epsilon}\bar{B}T(\bar{G}x^1 + \bar{r}) + \bar{A}x^1 - \bar{B}y^0 + \bar{q}. \tag{3.4}$$

As mentioned in the previous section, we take $T = (\bar{G}\bar{B})^{-1}$. Even if $\bar{G}\bar{B}$ is singular, then from Assumption 3.1 we have that $\bar{B}T\bar{G}$ and $\bar{B}T\bar{r}$ are well defined for all t and so we would never actually form or use $(\bar{G}\bar{B})^{-1}$ directly. We note that f and $\bar{f}$ are linear and so will satisfy a Lipschitz condition because of the assumptions of boundedness. It is not difficult to see that the solution of (3.3) is the same as that of (3.2) if $y^0 \equiv y$, where y is the solution of (3.2). Hence for the solution $\{x, y\}$ of (3.2), we have

$$x' = \bar{f}(t, x, y, \bar{r}, \bar{q}), \tag{3.5}$$

where $\bar{f}$ is defined in (3.4) and the initial conditions are (2.2).

According to [4], there exists a smooth matrix function $R(t) \in \mathbf{R}^{(n_x - n_y) \times n_x}$ that has full row rank and satisfies

$$(R\bar{B})(t) \equiv 0, \ \ t \in [0, t_f],$$

where R can be taken to have orthonormal rows. Let $P(t) = \bar{B}(\bar{G}\bar{B})^{-1}\bar{G}$ and define new variables

$$u = Rx, \ w = Px, \ 0 \le t \le t_f. \tag{3.6}$$

The inverse transformation exists and is given by

$$x = Su + w, \tag{3.7}$$

where S is a $n_x \times (n_x - n_y)$ matrix. Differentiating (3.6) and using (3.5) and (3.7), we obtain

$$u' = (R\bar{A} + R')Su + (R\bar{A} + R')w + R\bar{q}, \tag{3.8a}$$

$$\epsilon w' = -w + \epsilon(P\bar{A} + P')(Su + w) + \epsilon P\bar{q} - \bar{B}T\bar{r} - \epsilon\bar{B}y, \tag{3.8b}$$

where (3.8a) corresponds to the so-called essential underlying ODE (EUODE):

$$u' = (R\bar{A} + R')Su + \text{non-homogeneous terms}, \tag{3.9}$$

which is assumed to be stable. By stability, here we mean that the solution is bounded by a suitable stability constant times the sum of the initial condition of u and the non-homogeneous terms under the maximum norm.

We consider an s-stage explicit Runge-Kutta discretization of (3.3) on a mesh

$$\pi : 0 = t_0 < t_1 < \ldots < t_N = t_f,$$
$$h_n = t_n - t_{n-1}, \quad h = \max_{1 \le n \le N} h_n,$$

and denote by x_n^1, y_n^1 the corresponding approximations of $x^1(t_n)$, $y^1(t_n)$, respectively. In standard notation, the Runge-Kutta scheme reads

$$x_n^1 = x_{n-1}^1 + h_n \sum_{i=1}^{s} b_i \bar{f}(t_{ni}, X_{ni}^1, y^0, \bar{r}, \bar{q}), \tag{3.10}$$

$$X_{ni}^1 = x_{n-1}^1 + h_n \sum_{j=1}^{i-1} a_{ij} \bar{f}(t_{nj}, X_{nj}^1, y^0, \bar{r}, \bar{q}), \tag{3.11}$$

where $t_{ni} = t_{n-1} + h_n c_i$, $c_1 = 0$, and we assume $c_i \in [0,1]$ for $i = 1, 2, \ldots, s$. Applying the same scheme to (3.5), we have

$$x(t_n) = x(t_{n-1}) + h_n \sum_{i=1}^{s} b_i \bar{f}(t_{ni}, X_{ni}, y, \bar{r}, \bar{q}) + h_n \tau_n, \tag{3.12}$$

$$X_{ni} = x(t_{n-1}) + h_n \sum_{j=1}^{i-1} a_{ij} \bar{f}(t_{nj}, X_{nj}, y, \bar{r}, \bar{q}), \tag{3.13}$$

where $h_n \tau_n$ is the local truncation error. Let

$$\sum_{i=1}^{s} b_i = 1, \qquad \sum_{j=1}^{i-1} a_{ij} = c_i. \tag{3.14}$$

We assume the Runge-Kutta method is of order p, so $\tau_n = O_\epsilon(h_n^p)$, where we note a potential ϵ dependence. There has been much discussion on order reduction of implicit Runge-Kutta methods for stiff problems (cf. [13, 19, 17, 8]). The situation is generally worse for explicit methods. However, because the solution (x, y) of the DAE and its derivatives are independent of ϵ and because the solution satisfies $\bar{G}x + \bar{r} \equiv 0$, we can have high-order explicit Runge-Kutta methods for our SRM formulation with only a slight order reduction (see §5). The specific truncation errors τ_n of some explicit Runge-Kutta methods are given after the derivation and statement of the next theorem.

First we recall the usual stability restriction of explicit difference schemes. We assume

$$h/\epsilon = \max_n h_n/\epsilon < 1. \tag{3.15}$$

Then, subtracting (3.13) from (3.11) and using (3.15), we can easily obtain

$$|X_{ni}^1 - X_{ni}| \le K(|x_{n-1}^1 - x(t_{n-1})| + h_n \max_{1 \le i \le s} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))|). \tag{3.16}$$

Here and in what follows, K represents a generic constant independent of ϵ, h, and n, and $|\cdot|$ represents the vector maximum norm. If the problem is not singular, the dependence on the term $\bar{B}(t_{ni})$ may essentially be absorbed in K.

Let $e_n = x(t_n) - x_n^1$ and $E_{ni} = X_{ni} - X_{ni}^1$. Subtracting (3.10) from (3.12), and (3.11) from (3.13), and noting that $\bar{f}(t, x, y, \bar{r}, \bar{q})$ is linear in the variables x, y, $\bar{r}$, and $\bar{q}$, we have

$$e_n = e_{n-1} + h_n \sum_{i=1}^{s} b_i \bar{f}(t_{ni}, E_{ni}, y - y^0, 0, 0) + h_n \tau_n, \tag{3.17}$$

$$E_{ni} = e_{n-1} + h_n \sum_{j=1}^{i-1} a_{ij} \bar{f}(t_{nj}, E_{nj}, y - y^0, 0, 0). \tag{3.18}$$

Let $e_n^u = R(t_n)e_n$, $e_n^w = P(t_n)e_n$, $e_{ni}^u = R(t_{ni})E_{ni}$, and $e_{ni}^w = P(t_{ni})E_{ni}$. Then, $e_n = S(t_n)e_n^u + e_n^w$ and $E_{ni} = S(t_{ni})e_{ni}^u + e_{ni}^w$, where R, P, and S have been defined previously.

Now multiplying (3.17) by $P(t_n)$, and (3.18) by $P(t_{ni})$, and noting that $P(t_n)\bar{B}(t_n) = \bar{B}(t_n)$, $P(t_{ni})\bar{B}(t_{ni}) = \bar{B}(t_{ni})$, $P(t_n) - P(t_{n-1}) = O(h_n)$, and $P(t_{ni}) - P(t_{nj}) = O(h_n)$, we have

$$\begin{aligned} e_n^w = e_{n-1}^w - \frac{h_n}{\epsilon} \sum_{i=1}^{s} b_i e_{ni}^w + h_n \sum_{i=1}^{s} b_i [O\left(\frac{h_n}{\epsilon}\right) e_{ni}^w + P(t_n)\bar{A}(t_{ni})(S(t_{ni})e_{ni}^u + e_{ni}^w) + (1 + \\ O(h_n))\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))] + O(h_n)(S(t_{n-1})e_{n-1}^u + e_{n-1}^w) + h_n P(t_n)\tau_n, \end{aligned} \tag{3.19}$$

$$\begin{aligned} e_{ni}^w = e_{n-1}^w - \frac{h_n}{\epsilon} \sum_{j=1}^{i-1} a_{ij} e_{nj}^w + h_n \sum_{j=1}^{i-1} a_{ij} [O\left(\frac{h_n}{\epsilon}\right) e_{nj}^w + P(t_{ni})\bar{A}(t_{nj})(S(t_{nj})e_{nj}^u + e_{nj}^w) + \\ (1 + O(h_n))\bar{B}(t_{nj})(y(t_{nj}) - y^0(t_{nj}))] + O(h_n)(S(t_{n-1})e_{n-1}^u + e_{n-1}^w), \end{aligned} \tag{3.20}$$

Using (3.20) successively, we can obtain

$$\begin{aligned} e_{n2}^w = \left(1 - c_2 \frac{h_n}{\epsilon}\right) e_{n-1}^w + h_n [O\left(1 + \frac{h_n}{\epsilon}\right) e_{n-1}^w + \\ O(1)e_{n-1}^u + (1 + O(h_n))\bar{B}(t_{n1})(y(t_{n1}) - y^0(t_{n1}))], \end{aligned} \tag{3.21}$$

$$\begin{aligned} e_{n3}^w = \left(1 - c_3 \frac{h_n}{\epsilon} + a_{32} c_2 \frac{h_n^2}{\epsilon^2}\right) e_{n-1}^w + \\ h_n \sum_{j=1}^{2} \left[O\left(1 + \frac{h_n}{\epsilon}\right) e_{nj}^w + O(1)e_{nj}^u + (1 + O(h_n))\bar{B}(t_{nj})(y(t_{nj}) - y^0(t_{nj}))\right], \end{aligned} \tag{3.22}$$

$$\begin{aligned} e_{n4}^w = \left(1 - c_4 \frac{h_n}{\epsilon} + (a_{42}c_2 + a_{43}c_3)\frac{h_n^2}{\epsilon^2} - a_{43}a_{32}c_2 \frac{h_n^3}{\epsilon^3}\right) e_{n-1}^w + \\ h_n \sum_{j=1}^{3} \left[O(1 + \frac{h_n}{\epsilon}) e_{nj}^w + O(1)e_{nj}^u + (1 + O(h_n))\bar{B}(t_{nj})(y(t_{nj}) - y^0(t_{nj}))\right], \end{aligned} \tag{3.23}$$

etc.

Now from (3.16) we can deduce

$$|e_{ni}^w| \le K(|e_{n-1}^w| + |e_{n-1}^u| + h_n \max_{1 \le i \le s} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni})|)), \tag{3.24}$$

$$|e_{ni}^u| \le K(|e_{n-1}^w| + |e_{n-1}^u| + h_n \max_{1 \le i \le s} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))|). \tag{3.25}$$

Substituting the expressions for e_{n2}^w, e_{n3}^w, etc. into (3.19) and using (3.24) and (3.25), we obtain

$$\begin{aligned} |e_n^w| \le \alpha |e_{n-1}^w| + K h_n [\max_{1 \le i \le s} (|\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))|) + \\ \|e^w\|_{n-1} + \|e^u\|_{n-1} + |P(t_n)\tau_n|], \end{aligned} \tag{3.26}$$

where $\|\cdot\|_n = \max_{1\le j\le n} |\cdot_j|$ and

$$\alpha = 1 - \left(\sum_{i=1}^{s} b_i\right)\frac{h_n}{\epsilon} + \left(\sum_{i=2}^{s} b_i c_i\right)\frac{h_n^2}{\epsilon^2} - \left(\sum_{i=3}^{s} b_i \left(\sum_{j=2}^{i-1} a_{ij} c_j\right)\right)\frac{h_n^3}{\epsilon^3} + \cdots$$
$$= 1 - \frac{h_n}{\epsilon} + \frac{1}{2}\frac{h_n^2}{\epsilon^2} - \frac{1}{6}\frac{h_n^3}{\epsilon^3} + \cdots \le 1 - \frac{h_n}{\epsilon} + k\frac{h_n^2}{\epsilon^2},$$

where the order conditions of Runge-Kutta methods (cf. II.1-II.3 in [18]) have been used and $k = \frac{1}{2} < 1$. We thus have

$$\begin{aligned} |e_n^w| \le K & \Bigg[\left(1 - \frac{h_n}{\epsilon} + \frac{h_n^2}{2\epsilon^2}\right)\cdots\left(1 - \frac{h_1}{\epsilon} + \frac{h_1^2}{2\epsilon^2}\right)|e_0^w| + (\max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| \\ & + \|e^w\|_{n-1} + \|e^u\|_{n-1} + \|P\tau\|_n)\sum_{j=1}^{n} h_j \left(1 - \frac{h_{j+1}}{\epsilon} + \frac{h_{j+1}^2}{2\epsilon^2}\right)\cdots\left(1 - \frac{h_n}{\epsilon} + \frac{h_n^2}{2\epsilon^2}\right)\Bigg] \\ \le K & \left[e^{-\frac{1}{2\epsilon}t_n}|e_0^w| + \epsilon(\max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + \|e^w\|_{n-1} + \|e^u\|_{n-1} + \|P\tau\|_n)\right], \end{aligned} \tag{3.27}$$

where the last inequality is obtained by noting that (cf. [8])

$$1 - \frac{h}{\epsilon} + \frac{h^2}{2\epsilon^2} \le \exp\left(-\frac{h}{\epsilon}(1 - \frac{h}{2\epsilon})\right) \le \exp\left(-\frac{h}{2\epsilon}\right)$$

(using $h/\epsilon < 1$), which implies

$$\begin{aligned} \left(1 - \frac{h_{j+1}}{\epsilon} + \frac{h_{j+1}^2}{2\epsilon^2}\right)\cdots\left(1 - \frac{h_n}{\epsilon} + \frac{h_n^2}{2\epsilon^2}\right) &\le \exp\left(-\frac{h_{j+1}}{2\epsilon} - \cdots - \frac{h_n}{2\epsilon}\right) \\ &= \exp\left(-\frac{1}{2\epsilon}(t_n - t_j)\right), \end{aligned}$$

and then

$$\begin{aligned} \sum_{j=1}^{n} h_j \exp\left(-\frac{1}{2\epsilon}(t_n - t_j)\right) &\le \sum_{j=1}^{n} \int_{t_{j-1}}^{t_j} \exp\left(-\frac{1}{2\epsilon}(t_n - t)\right) dt \\ &= 2\epsilon\left(1 - \exp\left(-\frac{t_n}{2\epsilon}\right)\right). \end{aligned}$$

Hence, noting $e_0^w = 0$, we have

$$\|e^w\|_n \le K\,\epsilon\,\left[\max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + \|e^u\|_{n-1} + \|P\tau\|_n\right], \tag{3.28}$$

and from (3.24)

$$|e_{ni}^w| \le K\epsilon\left[\max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + \|e^u\|_{n-1} + \|P\tau\|_{n-1}\right]. \tag{3.29}$$

Similarly, multiplying (3.17) by $R(t_n)$, and (3.18) by $R(t_{ni})$, respectively, and noting that $R(t_n)\bar{B}(t_n) = 0$, $R(t_{ni})\bar{B}(t_{ni}) = 0$, $R(t_{ni}) - R(t_{n-1}) = c_i R'(t_{n-1})h_n + O(h_n^2)$, and $R(t_{ni}) - R(t_{nj}) =$

$O(h_n)$, we have

$$
\begin{aligned}
e_n^u &= e_{n-1}^u + h_n R'(t_{n-1}) S(t_{n-1}) e_{n-1}^u + h_n \sum_{i=1}^{s} b_i R(t_{ni}) \bar{A}(t_{ni}) S(t_{ni}) e_{ni}^u \\
&\quad + h_n [R'(t_{n-1}) e_{n-1}^w + O(h_n)(S(t_{n-1}) e_{n-1}^u + e_{n-1}^w) + \sum_{i=1}^{s} b_i (R(t_{ni}) \bar{A}(t_{ni}) e_{ni}^w \\
&\quad + O\left(\frac{h_n}{\epsilon}\right) e_{ni}^w + O(h_n) \bar{A}(t_{ni})(S(t_{ni}) e_{ni}^u + e_{ni}^w) + O(h_n) \bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni})))] \\
&\quad + h_n R(t_n) \tau_n, \qquad (3.30)
\end{aligned}
$$

$$
\begin{aligned}
e_{ni}^u &= e_{n-1}^u + h_n \sum_{j=1}^{i-1} a_{ij} R(t_{nj}) \bar{A}(t_{nj}) S(t_{nj}) e_{nj}^u + h_n [(c_i R'(t_{n-1}) \\
&\quad + O(h_n))(S(t_{n-1}) e_{n-1}^u + e_{n-1}^w) + \sum_{j=1}^{i-1} a_{ij} (R(t_{nj}) \bar{A}(t_{nj}) e_{nj}^w + O\left(\frac{h_n}{\epsilon}\right) e_{nj}^w \\
&\quad + O(h_n) \bar{A}(t_{nj})(S(t_{nj}) e_{nj}^u + e_{nj}^w) + O(h_n) \bar{B}(t_{nj})(y(t_{nj}) - y^0(t_{nj})))]. \qquad (3.31)
\end{aligned}
$$

Now, the scheme

$$
\begin{aligned}
e_n^u &= e_{n-1}^u + h_n R'(t_{n-1}) S(t_{n-1}) e_{n-1}^u + h_n \sum_{i=1}^{s} b_i R(t_{ni}) \bar{A}(t_{ni}) S(t_{ni}) e_{ni}^u \\
&\quad + h_n (\text{non-homogeneous terms}) \\
e_{ni}^u &= e_{n-1}^u + h_n \sum_{j=1}^{i-1} a_{ij} R(t_{nj}) \bar{A}(t_{nj}) S(t_{nj}) e_{nj}^u
\end{aligned}
$$

is a one-step difference scheme of the EUODE (3.9). Because this difference operator is independent of ϵ and consistent with the EUODE, and because the EUODE is assumed to be stable, there exists h_0 independent of ϵ such that when $h = \max_n \; h_n \le h_0$, the difference scheme is stable (cf. Theorem 5.38 of [3]); in other words, substituting the expressions for e_{ni}^u into the expression for e_n^u leads to a solution e_n^u that is bounded in terms of e_0^u and the non-homogeneous terms in the sense of the maximum norm. Hence substituting (3.31) into (3.30) we have

$$
|e_n^u| \le K \left[|e_0^u| + \max_{n,i} |e_{ni}^w| + \|e^w\|_{n-1} + h \max_{n,i} |\bar{B}(t_{ni})(y(t_{ni} - y^0(t_{ni}))| + \|R\tau\|_n \right]. \qquad (3.32)
$$

Noting that $e_0^u = 0$ and applying (3.29) and (3.28), we then have

$$
\|e^u\|_n \le K \left[\|e^w\|_{n-1} + \epsilon \max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + \|R\tau\|_n \right]. \qquad (3.33)
$$

From (3.33), (3.28), and $x(t_n) - x_n^1 = S(t_n) e_n^u + e_n^w$, we immediately have the following theorem:

THEOREM 3.1. *Let $x(t)$ be the solution of the problem (3.2) and let x_n^1 be the solution of the s-stage explicit Runge-Kutta method (3.10)-(3.11). If (3.14) and (3.15) hold and the EUODE (3.9) is stable, then under Assumption 3.1 we have*

$$
\max_{1 \le j \le n} |x(t_j) - x_j^1| \le K \left(\epsilon \max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + \epsilon \|P\tau\|_n + \|R\tau\|_n \right), \qquad (3.34)
$$

where matrices P and R are the same as those in (3.6). We note that if the DAE is not singular then Assumption 3.1 is automatically satisfied.

REMARK 3.1. *We now give the truncation errors of a few Runge-Kutta methods. We assume that matrices $\bar{A}(t)$, $\bar{B}(t)$, $P(t)$, etc. are sufficiently smooth so that $x(t)$ is sufficiently smooth. In general, if these matrices are not smooth enough, then besides the order reduction caused by the stiffness (see below), we also expect order reduction due to the non-smoothness. For simplicity of exposition, we denote $\bar{f}(t,x,y,\bar{r},\bar{q})$ from (3.5) as $\bar{f}(t,x)$.*

1. A second-order explicit Runge-Kutta scheme:

$$A = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix},\ b = (1/2, 1/2),\ \textit{and}\ c = (0,1). \tag{3.35}$$

For this scheme,

$$\tau_n = h_n^2 \left(-\frac{1}{12} \bar{f}^{(2)}(t_n, x(t_n)) + \frac{1}{4} \bar{f}_x \bar{f}^{(1)}(t_n, x(t_n)) \right),$$

where $\bar{f}^{(i)}(t,x(t)) = \frac{d^i}{dt^i}\bar{f}(t,x(t))$. Note that $\bar{G}x + \bar{r} \equiv 0$; thus, $\bar{f}^{(i)}(t,x(t))$ is uniformly bounded in ϵ. Note also that $\bar{f}_x = \frac{1}{\epsilon}P(t) + \bar{A}(t)$. Thus, $R(t)\bar{f}_x = R(t)\bar{A}(t)$ is also uniformly bounded in ϵ. Hence, for this scheme

$$\begin{aligned} &\epsilon P(t_n)\tau_n = O(h_n^2), \\ &R(t_n)\tau_n = h_n^2 \left[-\frac{1}{12} R(t_n) \bar{f}^{(2)}(t_n, x(t_n)) + \frac{1}{4} R(t_n)\bar{A}(t_n)\bar{f}^{(1)}(t_n) \right] = O(h_n^2). \end{aligned}$$

Both terms in the truncation error are independent of ϵ. Therefore,

$$\max_{1\le j\le n} |x(t_j) - x_j^1| \le K(\epsilon \max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + h^2). \tag{3.36}$$

2. The classical four-stage, fourth-order explicit Runge-Kutta scheme:

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, b = (1/6, 1/3, 1/3, 1/6),\ \textit{and}\ \ c = (0, 1/2, 1/2, 1). \tag{3.37}$$

This is a higher-order scheme. The terms $\bar{f}_{tx}$, $\bar{f}_{ttx}$, $\cdots$, are involved in the truncation error τ_n and thus $R(t)\bar{f}_{tx} = \frac{1}{\epsilon}R(t)P'(t)$, $R(t)\bar{f}_{ttx} = \frac{1}{\epsilon}R(t)P''(t)$, $\cdots$, are not generally zero. Furthermore, because of the stiffness (although less than for other formulations), we generally have order reduction. The error estimate becomes

$$\max_{1\le j\le n} |x(t_j) - x_j^1| \le K(\epsilon \max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + h^4/\epsilon^2). \tag{3.38}$$

If we can predict $y^0(t)$ or $\bar{B}(t)y^0(t)$ such that $\max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| = O(h^2)$ *and take $\epsilon = O(h^{2/3})$, then we have*

$$\max_{1\le j\le n} |x(t_j) - x_j^1| \le K h^{8/3}. \tag{3.39}$$

Such an order reduction might also appear in stabilization methods where recommended choices of stabilization parameters often depend on the step size h (cf. [8]).

3. The embedded Dormand-Prince 5(4) Runge-Kutta formula [15, 18]
The corresponding Runge-Kutta coefficient matrix $A = (a_{ij})$ is very cumbersome and is omitted here. This is a 6-stage method with global order $O(h^5)$ or local order $O(h^6)$. For non-stiff problems it can be verified that at internal stages, the local truncation error is $O(h^2)$ when $i = 2$, and at least $O(h^3)$ at the internal stages $i = 3,4,5,6$. We can then derive an

ϵ-dependent expression for the local truncation error of the method as follows[3]*. Without loss of generality, let $h_n \equiv h$. Using the above results on the truncation error at internal stages, noting that $\bar f(t_{ni}, x(t_{ni}))$ is independent of ϵ since $\bar G x + \bar r \equiv 0$ and that $X_{n1} = x(t_{n1}) = x(t_{n-1})$, and writing $\bar f(t_{nj}, X_{nj}) = \bar f(t_{nj}, x(t_{nj})) + \bar f(t_{nj}, X_{nj}) - \bar f(t_{nj}, x(t_{nj}))$ for $j \geq 2$, we can obtain*

$$
\begin{aligned}
x(t_{n2}) - X_{n2} &= x(t_{n-1} + c_2 h) - x(t_{n-1}) - a_{21} h \bar f(t_{n-1}, x(t_{n-1})) \\
&\equiv \alpha_n h^2 + O(h^3), \\
x(t_{n3}) - X_{n3} &= x(t_{n-1} + c_3 h) - x(t_{n-1}) - h \sum_{j=1}^{2} a_{3j} \bar f(t_{nj}, x(t_{nj})) \\
&\quad - a_{32} h \left(\frac{1}{\epsilon} P + \bar A \right) (X_{n2} - x(t_{n2})) \\
&= a_{32} P \alpha_n \frac{h^3}{\epsilon} + a_{32} (\bar A P) \alpha_n \frac{h^4}{\epsilon} + P \left[O\left(\frac{h^4}{\epsilon} \right) \right] + O\left(\frac{h^5}{\epsilon} \right) + O(h^3), \\
x(t_{ni}) - X_{ni} &= x(t_{n-1} + c_i h) - x(t_{n-1}) - h \sum_{j=1}^{i-1} a_{ij} \bar f(t_{nj}, x(t_{nj})) \\
&\quad - h \sum_{j=2}^{i-1} a_{ij} \left(\frac{1}{\epsilon} P + \bar A \right) (X_{nj} - x(t_{nj})) \\
&= (P\alpha_n) O\left(\sum_{j=3}^{i} \frac{h^j}{\epsilon^{j-2}} \right) + (\bar A P)(t_{nj}) \alpha_n O\left(\sum_{j=3}^{i} \frac{h^{j+1}}{\epsilon^{j-2}} \right) + P \left[O\left(\sum_{j=3}^{i} \frac{h^{j+1}}{\epsilon^{j-2}} \right) \right] \\
&\quad + O\left(\sum_{j=3}^{i} \frac{h^{j+2}}{\epsilon^{j-2}} \right) + O(h^3), \; i = 4, 5, 6,
\end{aligned}
$$

where α_n is a constant vector independent of ϵ and h and the order notation "O" is also independent of ϵ and h. Writing

$$
\begin{aligned}
\bar f(t_{ni}, X_{ni}) &= \bar f(t_{ni}, x(t_{ni})) + \bar f(t_{ni}, X_{ni}) - \bar f(t_{ni}, x(t_{ni})) \\
&= \bar f(t_{ni}, x(t_{ni})) + \left(\frac{1}{\epsilon} P + A \right) (X_{ni} - x(t_{ni}))
\end{aligned}
$$

in (3.12) and using above expressions of X_{ni} we obtain an expansion of local truncation error $h\tau_n$, where $(P\alpha_n)\frac{h^4}{\epsilon^2}$ is the only ϵ-dependent term associated with the power h^4. Note that the local truncation error of the Runge-Kutta method is of power h^6. Hence, the coefficient of the power h^4 should be zero and so $P\alpha_n = 0$. Therefore, the local truncation error $h\tau_n = P[O(\frac{h^6}{\epsilon^3})] + O(\frac{h^6}{\epsilon^2}) + O(h^6)$ since we assume $h < \epsilon$ and since all terms associated with powers of h^i, $i = 1, \cdots, 5$ must vanish to achieve the local truncation error $O(h^6)$ of the method. From Theorem 3.1, we thus have

$$
\max_{1 \leq j \leq n} |x(t_j) - x_j^1| \leq K(\epsilon \max_{n,i} |\bar B(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + h^5/\epsilon^2). \tag{3.40}
$$

If we have an accurate prediction for y then we can take ϵ relatively large so that the order reduction caused by the regularization parameter ϵ is small (see also numerical experiments in §5).

[3]The derivation is based on the known non-stiff order of the method to avoid a lengthy Taylor expansion.

We can also estimate $x(t_{ni}) - X^1_{ni}$. First, from (3.16) and (3.34) we have

$$|X_{ni} - X^1_{ni}| \le K(\epsilon \max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + \epsilon \|P\tau\|_n + \|R\tau\|_n). \tag{3.41}$$

Now let τ^I_{ni} stand for the local truncation error at the internal stage of the Runge-Kutta method (3.13). Then we have

$$x(t_{ni}) = x(t_{n-1}) + h_n \sum_{j=1}^{i-1} a_{ij} \bar{f}(t_{nj}, x(t_{nj}), y, \bar{r}, \bar{q}) + \tau^I_{ni}. \tag{3.42}$$

Subtracting (3.13) from (3.42) yields

$$x(t_{ni}) - X_{ni} = h_n \sum_{j=1}^{i-1} a_{ij} \bar{f}(t_{nj}, x(t_{nj}) - X_{nj}, 0, 0, 0) + \tau^I_{ni}. \tag{3.43}$$

Hence, using (3.43) successively in i, we have

$$|x(t_{ni}) - X_{ni}| \le K \max_{1 \le i \le s} |\tau^I_{ni}|. \tag{3.44}$$

Thus, (3.41) and (3.44) give the following:

THEOREM 3.2. *Suppose that the conditions of Theorem 3.1 hold. Then X^1_{ni} is an approximation of $x(t_{ni})$ at the internal stage point t_{ni} with the following error estimate:*

$$|x(t_{ni}) - X^1_{ni}| \le K(\epsilon \max_{n,i} |\bar{B}(t_{ni})(y(t_{ni}) - y^0(t_{ni}))| + \epsilon \|P\tau\|_n + \|R\tau\|_n + \max_{1 \le i \le s} |\tau^I_{ni}|), \tag{3.45}$$

where τ_i and τ^I_{ni} are the truncation error and local internal truncation error, respectively.

The approximate solution y^1_n can be obtained by (2.4), i.e.,

$$y^1_n = y^0_n + \frac{1}{\epsilon}(G(x^1_n, t_n)B(x^1_n, t_n))^{-1} g(x^1_n, t_n), \tag{3.46}$$

or an equivalent version multiplied by $B(x^1_n, t_n)$ in the case for singular DAEs (cf. [4, 5]). However, for higher-order schemes, we do not expect the accuracy for y^1_n to be as high as for x^1_n with this y-updating scheme (3.46) even if there is no singularity in the DAE. Some examples showing order reduction for y^1_n are given in [4] and in §4 of this paper. A better alternative is based on an exact expression of y from the original differential-algebraic system (2.1):

$$y = (GB)^{-1}(Gf + g_t). \tag{3.47}$$

Hence, we can calculate an approximate solution of y from

$$\bar{y}^1_n = (G(x^1_n, t_n)B(x^1_n, t_n))^{-1}[G(x^1_n, t_n)f(x^1_n, t_n) + g_t(x^1_n, t_n)]. \tag{3.48}$$

So if the function $(GB)^{-1}(Gf + g_t)|_{(x,t)}$ satisfies a Lipschitz condition with respect to x, then we have

$$|\bar{y}^1_n - y(t_n)| \le K|x^1_n - x(t_n)| \tag{3.49}$$

with no order reduction if the DAE is not singular. Similarly,

$$|\bar{y}^1_{ni} - y(t_{ni})| \le K|X^1_{ni} - x(t_{ni})|, \tag{3.50}$$

where

$$\bar{y}^1_{ni} = (G(X^1_{ni}, t_{ni})B(X^1_{ni}, t_{ni}))^{-1}[G(X^1_{ni}, t_{ni})f(X^1_{ni}, t_{ni}) + g_t(X^1_{ni}, t_{ni})]. \tag{3.51}$$

We point out that in the singular case, even if Assumption 3.1 is satisfied, we need not have a Lipschitz condition for $(B(GB)^{-1}(Gf+g_t))(x,t)$ with respect to x since $\frac{\partial}{\partial x}(B(GB)^{-1}g_t)$ may become unbounded at the singular point. In this case, order reduction in the quantity By is expected near the singularity and this order reduction will affect the accuracy of x in the computation. We will use an alternative formulation to mitigate this order reduction, namely,

$$By = f(x,t) - x'. \tag{3.52}$$

This formulation is better for the singular problem with which we are concerned because x' is assumed to be smooth. Although there may be an $O(h)$ reduction in approximating x', the quantity By in the SRM formulation is multiplied by a factor ϵ that will essentially allow us to recover the expected accuracy in the overall computation. These points will be demonstrated in the next section by means of an example.

The newly obtained y^1_n and y^1_{ni} from the y-updating schemes (2.4), (3.48), or (3.52) will be used to produce the initial guesses y^0_n and y^0_{ni} in (3.11) for computing X^1_{ni} recursively and then for computing x^1_{n+1}. Because the local truncation error τ^I_{ni} is of lower order at the internal stages of Runge-Kutta methods, we need to develop higher-order prediction schemes. One such scheme will be given in §5 based on the continuous extension of the Runge-Kutta method.

4. The prediction algorithm for the PSRM. We now consider the implementation of the SRM with a better predicted initial guess. For simplicity of notation, we describe the method for nonsingular DAEs. For singular DAEs, y can be replaced by By unless otherwise indicated.

Before implementing the prediction, we need to rearrange the order of the implementation of the numerical method applied to the regularized problem (2.3)-(2.4). As presented, when using the SRM to find $x^\sigma(t)$ and $y^\sigma(t)$, we require knowledge of the function $y^{\sigma-1}(t)$; or numerically, we require $y^{\sigma-1}(t_n)$, $n = 1, 2, \ldots$. Thus in principle we would have to store all these values in order to compute $x^\sigma(t_n)$ and $y^\sigma(t_n)$. This storage requirement is unnecessarily large in the context of an initial-value problem. In [4], a rearrangement of computing order is briefly mentioned; namely, at the time t_n, we compute $x^\sigma(t_n)$ and $y^\sigma(t_n)$ for all $\sigma = 0, 1, \ldots, m$, where m is the total number of SRM iterations. This corresponds to other dynamic iterative methods such as waveform relaxation methods, where the computation proceeds with windows. Our case corresponds to taking the window size equal to the step size. We note that the rearrangement is only algorithmic: This SRM formulation does not change the discrete SRM formulation. See Figure 4.1 for an illustration of the rearrangment for the forward Euler method. Other one-step and multi-step methods can be similarly rearranged. In the figure, solid arrows show values at previous steps or iterations needed in computing x^σ_n and dashed arrows show the values at previous steps or iterations needed in computing y^σ_n. For example, in computing x^2_3 we only need to know y^1_2 and x^2_2, and in computing y^2_3 we only need to know x^2_3 and y^1_3. From the figure, we see that only values in the dotted rectangle at the previous time step need to be stored for computing values in the dotted rectangle at the current time step. With this rearrangement, the storage is reduced to O(number of iterations), where the number of iterations m is roughly $O(p)$, and p is the order of the numerical method used. In addition, this rearrangement makes the prediction of a better initial guess possible since before starting to find $y^\sigma(t_{n+1})$ $(\sigma = 0, 1, \cdots, m)$ at t_{n+1}, we have already obtained good approximations of x and y at all previous time steps, and they can be used to predict the initial guess $y^0_{n+1} = y^0(t_{n+1})$ at t_{n+1}.

Now the question is how to perform the prediction step using the approximate solution obtained at previous steps. The simplest way would be to let $y^0(t_{n+1}) = y^m(t_n)$. But the predicted solution in this case is only a first-order approximation to the exact solution. From the corollary of the previous section, at most one iteration is reduced by this prediction scheme. Alternatively, we can use higher-order extrapolation formulas, e.g., the second-order extrapolation $y^0(t_{n+1}) = 2y^m(t_n) - y^m(t_{n-1})$.

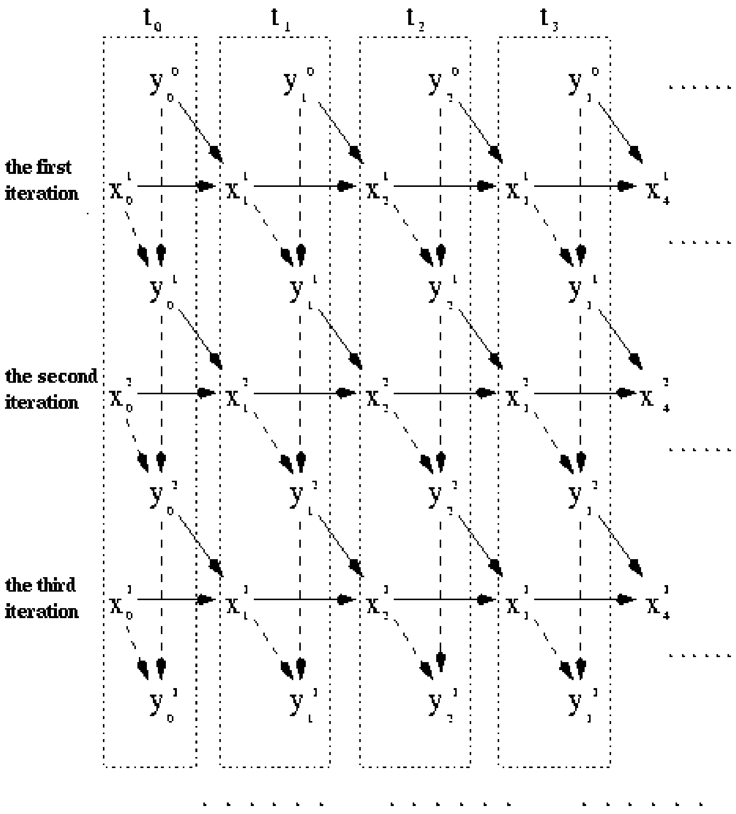

FIG. 4.1. *Illustration of the rearrangement for a one-step method where the initial guess* $y_n^0 = y^0(t_n)$ *and the initial value* $x_0^\sigma = \beta$.

Unfortunately, this kind of extrapolation formula is not stable for the y-updating scheme (3.46). The instability is due to coefficients in the extrapolation formulas having magnitude greater than 1. We illustrate this for the linear problem (3.2a)-(3.2b). Using $m = 1$ and the y-updating scheme (3.46),

$$\begin{aligned} y_n^1 &= 2y_{n-1}^1 - y_{n-2}^1 + \frac{1}{\epsilon}(GB)^{-1}(Gx_n^1 + r(t_n)) \\ &= (2^2 - 1)y_{n-2}^1 - 2y_{n-3}^1 + \frac{1}{\epsilon}[(GB)^{-1}(Gx_n^1 + r(t_n)) + (GB)^{-1}(Gx_{n-1}^1 + r(t_{n-1}))] \\ &= \cdots = O(2^n) + \frac{1}{\epsilon}\sum_{j=1}^{n} (GB)^{-1}(Gx_j^1 + r(t_j)). \end{aligned}$$

On the other hand, from equations (3.28) and (3.33) to estimate the error of difference schemes, we see that the part involving y is multiplied by a factor ϵ. Therefore, in computing x_n^1 the exponential increase of y_n^1 is controlled at earlier time steps. A similar control does not appear in the y-updating scheme (3.46). Therefore, y_n^1 increases dramatically as n increases, eventually destabilizing the scheme. On the other hand, the y-updating scheme (3.48) involves only x_n^1. No such exponential increase appears in updating y_n^1. Thus, the method with this y-updating scheme is stable. The following example illustrates the above analysis.

EXAMPLE 4.1. *Consider the DAE (2.1) with*

$$f = \begin{pmatrix} 1-e^{-t} \\ \cos t + e^t \sin t \end{pmatrix}, \; B = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$
$$g = \frac{1}{2}(x_1^2 + x_2^2 - e^{-2t} - \sin^2 t),$$

subject to $x_1(0) = 1,\ x_2(0) = 0$.

The exact solution is $x = (\,e^{-t}, \quad \sin t\,)$, $y = e^t$. *Using the explicit second-order Runge-Kutta method (3.35) with* $h = 0.001$ *and* $\epsilon = 0.005$, *we list the computational results adopting the simple prediction scheme* $y_{n+1}^0 = y_n^m$ *in Table 4.1. At a given time t, we use 'ex' to denote the maximum over all components of the error in x, while 'ey' denotes the maximum over all components of the error in y. Similarly, 'drift' denotes the maximum residual in the algebraic equations. We note that this is the most computationally inexpensive variant for a second-order accurate scheme.*

y updating	error at →	t=.1	t=.3	t=.5	t=.7	t=1.0
(3.46)	ex	5.035e-6	5.619e-6	6.697e-6	7.712e-6	1.003e-5
	ey	5.119e-5	2.439e-5	1.463e-5	5.294e-5	9.631e-5
	drift	4.558e-6	4.291e-6	4.924e-6	6.658e-6	1.146e-5
(3.48)	ex	5.324e-6	5.789e-6	6.749e-6	7.661e-6	9.779e-6
	ey	1.218e-5	1.546e-5	1.788e-5	2.077e-5	3.334e-5
	drift	4.822e-6	4.420e-6	4.937e-6	6.555e-6	1.120e-5

TABLE 4.1

Errors for Example 4.1 using the explicit second-order Runge-Kutta scheme (3.35) with the simple prediction formula $y_{n+1}^0 = y_n^m$ *and one SRM iteration.*

We observe that for the second-order method, the number of iterations is reduced to one by the simple prediction scheme. If we use the second-order extrapolation scheme

$$y_{n+1}^0 = 2y_n^m - y_{n-1}^m \tag{4.1}$$

and y-updating scheme (3.46), the error increases very rapidly after a short time and overflow occurs at $t = 0.2$. *On the other hand, if we use (4.1) and y-updating scheme (3.48), the computation is stable and the results are improved. The results are shown in Table 4.2.*

y updating	error at →	t=.1	t=.3	t=.5	t=.7	t=1.0
(3.48)	ex	2.590e-7	4.224e-7	3.332e-7	2.753e-7	2.546e-7
	ey	6.115e-7	1.331e-6	9.432e-7	4.032e-7	2.387e-7
	drift	2.328e-7	1.354e-7	4.538e-8	3.901e-8	1.487e-7

TABLE 4.2

Errors for Example 4.1 using the explicit second-order Runge-Kutta scheme (3.35) with the second-order extrapolation prediction formula (4.1) and one SRM iteration.

Next we consider the use of simple, explicit, and relatively low-order discretizations of the original differential-algebraic equations (2.1) to predict a better initial guess at the next time step of the SRM. Because the approximations to x and y at previous steps are both available, we can apply an explicit one-step or multi-step method directly to the differential equations (2.1a) to compute the approximation of $x(t_{n+1})$ at the current time step. Then we can obtain the initial guess $y^0(t_{n+1})$ by the formula (3.47). Unlike the extrapolation prediction formula, which generally causes an exponential increase in y, this prediction formula is based on difference schemes of (2.1a) and should work well for both y-updating schemes (3.46) and (3.48), as long as the difference schemes themselves are stable. The prediction proceeds as follows:

Algorithm 4.1 *Prediction based on the differential equation (2.1a):*

1. *Record values at previous time steps:* $x_n, y_n; x_{n-1}, y_{n-1}; \cdots; x_{n-k+1}, y_{n-k+1}$.
2. *Use a k-step method for (2.1a) to find an approximation* x^0_{n+1} *to* $x(t_{n+1})$.
3. *Predict* y^0_{n+1} *using (3.47):*

$$y^0_{n+1} = (GB)^{-1}(Gf + g_t)|_{(t_{n+1}, x^0_{n+1})}.$$

Clearly, if $k = 1$, the numerical method in Step 2 is a one-step method.

EXAMPLE 4.2. *Consider the problem in Example 4.1. To numerically integrate the regularized problem (2.3)-(2.4), we use the fourth-order classical explicit Runge-Kutta method (3.37). We apply the two-step (second-order) Adams-Bashforth method for the differential equation (2.1a) to predict* x *at the next time step* t_{n+1} *and then apply (3.47) to predict* y^0_{n+1}. *For this higher-order Runge-Kutta method, the internal stage at* $t_n + h_n/2$ *is involved. So we need to predict* $y^0_{n+1/2}$ *at the internal stage points* $t_n + h_n/2$. *We use the following averaging scheme:*

$$y^0_{n+1/2} = \frac{1}{2}(y^0_n + y^0_{n+1}). \tag{4.2}$$

Implementation issues, such as predicting the initial guess y *at the internal stage points of general high-order Runge-Kutta methods, will be discussed in the next section. For this prediction scheme (i.e., two-step Adams-Bashforth plus (3.47) and (4.2)), we have*

$$\max_{n,i} |(y(t_{ni}) - y^0_{ni}))| = O(h^2).$$

By Theorem 3.1 and the remark in §3, we expect that one SRM iteration would achieve $O(\epsilon h^2 + h^4/\epsilon^2) = O(h^{8/3})$ *accuracy if* $\epsilon = O(h^{2/3})$. *The numerical errors are shown in Table 4.3 where* $h = 0.001$. *The order of convergence at* $t = 1$ *is calculated in the case* $\epsilon = h^{2/3}$ *by halving* h *(and* ϵ*).*

y updating	error at →	t=.1	t=.3	t=.5	t=.7	t=1.0
(3.46)	ex	1.326e-8	9.672e-9	6.197e-9	3.735e-9	2.168e-9
	ey	1.765e-8	6.160e-8	1.261e-7	1.945e-7	2.869e-7
	drift	1.243e-12	1.352e-10	3.369e-10	6.229e-10	1.212e-9
(3.48)	ex	9.123e-9	6.644e-9	4.301e-9	2.711e-9	1.838e-9
	ey	1.206e-8	1.256e-8	8.268e-9	2.933e-9	2.343e-9
	drift	1.246e-12	1.354e-10	3.375e-10	6.239e-10	1.214e-9

TABLE 4.3

Errors for Example 4.1 using the explicit fourth-order Runge-Kutta scheme (3.37) with Algorithm 4.1 and one SRM iteration.

The result is shown in Table 4.4 and is close to our theoretical prediction.

y updating	variable x_1	variable x_2	variable y
(3.46)	2.76	2.63	2.07
(3.48)	2.77	2.63	2.59

TABLE 4.4

Numerical order with $\epsilon = h^{2/3}$ *for Example 4.1 using the explicit fourth-order Runge-Kutta scheme (3.37) with Algorithm 4.1 and one SRM iteration.*

From the results, we can see that the y*-updating scheme (3.46) works for the prediction algorithm based on differential equation (2.1), although there may be an order reduction for variable* y. *The* y*-updating scheme (3.48) still has better accuracy than (3.46).*

It is also worth mentioning that the prediction Algorithm 4.1 is based on the difference schemes directly applied to the differential equations (2.1a). The stability of differential equations (2.1a) is different from that of the DAEs (2.1). For example, for linear problems, the stability of the DAEs

(2.1) is determined by the stability of the EUODE (3.9). But the stability of the EUODE (3.9) does not imply the stability of the differential equations (2.1a); that is, a direct solution of (2.1a) may be unstable. The SRM is a reformulation solely based on the EUODE. However, even if the prediction algorithm is based on an unstable problem, one iteration of the SRM is able to correct the poor predicted value and obtain a good approximate solution of the original stable problem. The following example supports these claims.

EXAMPLE 4.3. *Consider the DAE (2.1) with*

$$f = \begin{pmatrix} -\sin t + (1+t^2)\sin t\cos t \\ -\cos t - 5\sin t - 5x_2 \end{pmatrix}, \quad B = \begin{pmatrix} 1+t^2 \\ 0 \end{pmatrix},$$
$$g = e^{-2t}(x_1 + x_2 - \cos t + \sin t),$$

subject to $x_1(0) = 1,\ x_2(0) = 0$.

The exact solution is $x = (\cos t,\ \ -\sin t)$, $y = \sin t\cos t$. *The differential part (2.1a) of the DAE is unstable. This can be seen from the results in Table 4.5 that are obtained by applying the three-step (third-order) Adams-Bashforth method to (2.1a) for computing* x *and by using (3.47) for computing* y.

error at →	t=1	t=3	t=5	t=7	t=10
ex	6.078e-8	3.308e-6	1.806e-4	9.859e-3	3.977
ey	6.059e-8	6.615e-7	1.389e-5	3.944e-4	7.876e-2
drift	8.218e-9	8.198e-9	8.198e-9	8.198e-9	8.198e-9

TABLE 4.5

Errors for Example 4.3 using only the prediction algorithm 4.1.

We then use our predicted SRM with the classical fourth-order Runge-Kutta method to the regularized problem (2.3)-(2.4). The prediction algorithm is chosen to be the same as in Example 4.2, i.e., two-step Adams-Bashforth plus (3.47) and (4.2). We have pointed out in Example 4.2 that if $\epsilon = O(h^{2/3})$, *the accuracy of the method is* $O(h^{8/3})$ *after one SRM iteration — the order is close to the three-step Adams-Bashforth method. Table 4.6 shows the corrected results to those in Table 4.5. Again, we see that both* y*-updating schemes work well. We also see some order reduction in computing* y *by (3.46).*

y updating	error at →	t=1	t=3	t=5	t=7	t=10
(3.46)	ex	1.660e-9	1.876e-9	1.224e-8	4.137e-8	7.706e-8
	ey	1.658e-7	3.750e-8	9.342e-8	1.642e-7	1.513e-7
	drift	2.247e-10	4.651e-12	5.558e-13	3.440e-14	1.588e-16
(3.48)	ex	1.663e-9	1.880e-9	1.227e-8	4.146e-8	7.721e-8
	ey	1.664e-9	3.760e-10	9.436e-10	1.658e-9	1.529e-9
	drift	2.251e-10	4.661e-12	5.569e-13	3.447e-14	1.591e-16

TABLE 4.6

Errors for Example 4.3 using the explicit fourth-order Runge-Kutta scheme (3.37) with Algorithm 4.1 and one SRM iteration.

Finally in this section, we also consider an example given in [4]. It is a linear DAE with GB singular at $t = 0.5$.

EXAMPLE 4.4. *Consider the DAE (2.1) with*

$$f = \begin{pmatrix} -x_1 + x_2 - \sin t \\ 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 1-2t \end{pmatrix},$$
$$g = (1-2t)(x_1^2 + x_2^2) - (1-2t)(e^{-t} + \sin t),$$

subject to $x_1(0) = 1,\ x_2(0) = 0$.

The exact solution is $x = (\, e^{-t}, \quad \sin t\,)$, $y = \frac{\cos t}{1-2t}$. *A singularity is located at* $t = 0.5$, *at which point the exact solution* y *becomes unbounded whereas the quantity* By *remains bounded. Using the explicit fourth-order Runge-Kutta method (3.37) and the two-step Adams-Bashforth predicting scheme with a uniform* $h = 0.001$ *and* $\epsilon = h^{2/3}$ *as in Example 4.2, we list the computational results in Table 4.7. From these results, we observe order reductions near the singularity. As pointed out in*

y updating	error at →	t=0.1	t=0.3	near t=0.5	t=0.7	t=1.0
(3.46)	ex	7.069e-8	7.895e-8	3.177e-4	6.135e-7	3.143e-7
	ey	1.435e-6	1.038e-5	1.989e+3	1.508e-5	1.358e-6
	drift	8.344e-9	1.687e-8	3.166e-7	1.873e-8	7.275e-9
(3.48)	ex	5.133e-8	6.601e-8	6.734e-4	1.897e-5	1.038e-5
	ey	8.193e-8	3.074e-7	1.344e+3	9.498e-5	2.073e-5
	drift	8.482e-8	1.694e-8	6.718e-7	1.860e-8	7.249e-9

TABLE 4.7

Errors for Example 4.4 using the explicit fourth-order Runge-Kutta scheme (3.37) and the two-step Adams-Bashforth predicting with (4.2) and one SRM iteration.

the previous section, the By*-updating formula (3.48) does not work well near the singularity because* $B(GB)^{-1}g_t$ *does not satisfy a Lipschitz condition with respect to* x *at the singular point. So we observe that the SRM updating formula (3.46) for* By *works better than the exact updating formula (3.48) for* By. *However, there is still an order reduction near the singularity for the formula (3.46). The reason is that in predicting* $B(t_{n+1})y^0_{n+1}$, *we have used the exact formula (3.48) for* By *and in the half-step formula (4.2) for* By *(replacing the quantity* y *by the quantity* By*)*

$$\frac{1}{2}\left((B(t_n)y^0_n + B(t_{n+1})y^0_{n+1}\right)$$

does not belong to Range($B(t_{n+1/2})$*) at the singularity. This may also cause the loss of accuracy as indicated in [5].*

An alternative is to replace the exact formula (3.48) for By *with (3.52) near the singularity as mentioned in the previous section for predicting both* $B(t_{n+1})y^0_{n+1}$ *and the internal stage* $B(t_{n+1/2})y^0_{n+1/2}$. *From the values obtained from the two-step Adams-Bashforth predicting scheme, we can obtain* $x'(t_{n+1})$, $x(t_{n+1/2})$, *and* $x'(t_{n+1/2})$ *by the formulas*

$$x'(t_{n+1}) = \frac{x(t_{n-1}) - 4x(t_n) + 3x(t_{n+1})}{2h},$$

$$x(t_{n+1/2}) = \frac{-x(t_{n-1}) + 6x(t_n) + 3x(t_{n+1})}{8},$$

and

$$x'(t_{n+1/2}) = \frac{x(t_{n+1}) - x(t_n)}{h}.$$

From these values, we can then predict By *using (3.52) near the singularity. The computational errors are shown in Table 4.8. We observe no order reduction for variable* x *near the singularity* $t = 0.5$.

5. The SRM as a dynamic iterative method and high-order schemes. The basic idea of a dynamic iterative method, specifically waveform relaxation, is to solve a sequence of differential equations having solutions $x^1(t), \ldots, x^\sigma(t)$, given an initial guess $x^0(t)$, such that this solution sequence converges to the solution $x(t)$ of the original differential equation over the entire interval of integration. WR algorithms were originally introduced by Lelarasmee et al. [26] and perform

y updating	error at $\rightarrow$	t=0.1	t=0.3	near t=0.5	t=0.7	t=1.0
(3.46)	ex	4.791e-8	3.220e-8	5.708e-8	1.299e-8	7.126e-9
	ey	8.940e-8	1.733e-7	1.862e-2	6.403e-8	7.535e-9
	drift	1.767e-10	3.984e-11	3.593e-11	1.484e-11	7.614e-11

TABLE 4.8

Errors for Example 4.4 using the explicit fourth-order Runge-Kutta scheme (3.37) with the two-step Adams-Bashforth predicting and (3.52) and one SRM iteration.

efficiently on problems arising from electrical network modeling (see e.g., [35]). Convergence properties of WR algorithms for linear problems have been investigated in [31], and generalized to linear constant-coefficient differential-algebraic systems in [30]. Other work on WR related to DAEs includes [25] for linear RC-circuits, [22] for some index-1 DAEs, and [21] for implicit differential equations, emphasizing the implementation of the internal iteration in solving nonlinear systems. It is also important to note the works [32], which introduced and proved convergence results under certain dominance conditions for *asynchronous* WR algorithms for ODEs; [9], which introduced asynchronous WR algorithms for fully implicit index-1 DAEs; and [16], which compared the effects of multisplittings with different overlaps on convergence and computational efficiency of WR algorithms. Also, the work of [34] examined accelerated WR algorithms, where a method is given that is theoretically optimal in terms of computational complexity for a class of parabolic PDEs. Our emphasis in the section, however, will be on writing the PSRM as a waveform relaxation algorithm for the nonlinear index-2 DAEs (2.1) so that implementation techniques (such as higher-order methods, local error control, etc.) in the waveform relaxation context (see, e.g., [14]) can be implemented directly in the context of the PSRM. We will also remark on the guidance for waveform splitting for DAEs given in [30].

We write the index-2 DAE (2.1) as

$$x' = f(x,t) - B(x,t)y, \tag{5.1a}$$

$$y = y + \frac{1}{\epsilon}T(x,t)g(x,t), \tag{5.1b}$$

where the parameter ϵ and the matrix T are the same as those defined in §2. Let

$$M(x,y,t) = \begin{pmatrix} f(x,t) - B(x,t)y \\ -y + \frac{1}{\epsilon}T(x,t)g(x,t) \end{pmatrix}, \quad N(x,y,t) = \begin{pmatrix} 0 \\ y \end{pmatrix}. \tag{5.2}$$

We then formulate a waveform relaxation algorithm:

$$\begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x^\sigma \\ y^\sigma \end{pmatrix}' = M(x^\sigma, y^\sigma, t) + N(x^{\sigma-1}, y^{\sigma-1}, t). \tag{5.3}$$

The WR algorithm (5.3) is clearly equivalent the SRM (2.3)-(2.4). Theorem 2.1 shows the convergence of this WR algorithm at least in a region away from the initial point $t = 0$. The convergence rate $O(\epsilon^\sigma)$ is very fast compared with usual WR algorithms.

Next, we consider numerical methods used to solve the waveform iterations. Results in [35] show that the discretized iterates will converge provided the global discretization errors in the numerical approximation are driven to zero. However, it is pointed out in [14] that these results do not offer any bounds on the convergence rates of the discretized waveform iterates and the norm used to prove the contraction mapping is problem dependent. In addition, it is not clear what is meant by driving the global discretization errors in the numerical approximations to zero. These are not problems with our SRM-WR method (5.3). Error estimates have been given in §3 for linear problems. We only perform a few number of iterations (or only one iteration for the PSRM), so the error will not accumulate much during these few iterations. From the estimates in Theorems 3.1-3.2, (3.49), and

(3.50), the error depends on the error $y - y^0$ at internal stage points. Therefore, obtaining higher-order approximations of y at internal stage points becomes very important in designing higher-order methods for (5.3) or (2.3), (2.4).

Again, because of the favorable stiffness properties of the SRM, we focus on explicit methods. We consider an explicit s-stage Runge-Kutta method (cf. (3.10) and (3.11) or [14]):

$$x_n^\sigma = x_{n-1}^\sigma + h_n \sum_{i=1}^{s} b_j F(t_{ni}, X_{ni}^\sigma, y^{\sigma-1}), \tag{5.4}$$

$$X_{ni}^\sigma = x_{n-1}^\sigma + h_n \sum_{j=1}^{i-1} a_{ij} F(t_{nj}, X_{nj}^\sigma, y^{\sigma-1}), \tag{5.5}$$

where $F(t, x^\sigma, y^{\sigma-1}) = -\frac{1}{\epsilon} B(GB)^{-1} g(x^\sigma, t) + f(x^\sigma, t) - B(x^\sigma, t) y^{\sigma-1}$, and $A = (a_{ij})$, $b = (b_i)$, and $c = (c_i)$ satisfy (3.14).

For linear problems (cf. Theorem 3.2), the error of the internal approximation X_{ni}^1 depends on the truncation error τ_{ni}^I at the internal stage. Because we are using explicit Runge-Kutta methods, the difference scheme for X_{n2}^1 at the internal stage is the forward Euler method, and thus its local truncation error $\tau_{n2}^I = O(h_n^2)$. Therefore, by Theorem 3.2, the accuracy at internal stage points X_{ni}^1 is generally at most $O(h^2)$. Then the initial guess y_{ni}^0 computed based on X_{ni}^1 and formula (3.51) is generally of $O(h^2)$. So if we use X_{ni}^1 to predict the initial guess y^0 at the internal stage (i.e., y_{ni}^0), the SRM and the PSRM are generally at most $O_\epsilon(h^3)$ accurate by Theorem 3.1.

We thus look for an approximation of $x(t_{ni})$ with better accuracy than $O(h^2)$. An approach that involves generating a continuous approximation at internal stage points by interpolation was summarized in [14] for implementing discrete WR methods. In [12], an s-stage continuous Runge-Kutta method is considered in which the interpolation is supplied by natural continuous extensions (see [36]). In these methods, values at internal stage points can be calculated by

$$x^\sigma(t_{n-1} + \theta h_n) = x_{n-1}^\sigma + h_n \sum_{i=1}^{s} b_i(\theta) F(t_{ni}, X_{ni}^\sigma, y^{\sigma-1}), \tag{5.6}$$

where $b_i(\theta)$ satisfy $b_i(0) = 0, b_i(1) = b_i$, $i = 1, \cdots, k$ and are polynomials of degree d where $[\frac{p+1}{2}] \le d \le p$, and p is the underlying order of the Runge-Kutta method. A list of such methods is given in e.g., [12, 36]. The well-known continuous extension of the Dormand and Prince method given in [18] is a convenient choice for our purposes. It is a 6-stage method with a fourth-order continuous extension that can be implemented without any extra right-hand side evaluations. The solution has fifth-order accuracy when $\theta = 1$. The fourth-order accuracy in computing y^0 at internal stage points will be achieved by the fourth-order continuous extension. Hence, by Theorem 3.1 and Remark 3.1, the accuracy of one SRM iteration for this scheme is generally equal to

$$\epsilon O_\epsilon(h^4) + O(h^5/\epsilon^2).$$

With this estimate, we expect only a small order reduction when ϵ is chosen to be relatively large. Tables 5.1-5.2 give computational errors with $h = 0.001$ and the y-updating scheme (3.48) for problems in Examples 4.1 and 4.3, indicating that for both linear and nonlinear problems the error decreases when ϵ increases from $O(h)$ to $O(h^{1/2})$. Because we only need one iteration for the predicted SRM, the error control is similar to that used by common explicit one-step methods. Therefore, by combining this continuous extension with the embedded Dormand-Prince 5(4) Runge-Kutta formula, we can obtain a higher-order method for the SRM or PSRM iteration, where local error estimation and step-size control can potentially be done. We report on these aspects elsewhere.

The study of WR algorithms thus provides implementational techniques for the SRM. Conversely, the study of the SRM provides new insight for the WR algorithms. We collect our results into the following remarks.

ϵ	error at $\rightarrow$	t=0.1	t=0.3	t=0.5	t=0.7	t=1.0
0.005	ex	2.510e-12	1.549e-12	4.315e-13	4.572e-13	1.645e-12
	ey	5.466e-12	3.702e-12	1.113e-12	1.120e-12	4.562e-12
	drift	2.300e-12	1.328e-12	4.165e-13	4.596e-13	1.646e-12
0.05	ex	2.776e-15	1.443e-15	6.661e-16	3.331e-16	2.220e-15
	ey	6.217e-15	3.775e-15	8.882e-16	8.882e-16	4.884e-15
	drift	2.603e-15	1.242e-15	5.412e-16	3.331e-16	1.943e-15

TABLE 5.1

Errors for Example 4.1 using the fifth-order Dormand-Prince Runge-Kutta scheme with a fourth-order continuous extension and one SRM iteration.

ϵ	error at $\rightarrow$	t=1	t=3	t=5	t=7	t=10
0.005	ex	8.734e-13	3.140e-12	3.512e-12	3.006e-13	7.036e-13
	ey	8.751e-13	6.426e-13	2.686e-13	3.830e-15	2.093e-14
	drift	1.183e-13	7.835e-15	1.592e-16	2.010e-19	1.658e-21
0.05	ex	8.882e-16	1.255e-14	2.359e-14	5.873e-14	1.007e-13
	ey	2.609e-15	1.199e-14	2.776e-16	6.939e-15	5.884e-15
	drift	1.804e-16	2.006e-17	7.996e-19	2.234e-20	8.644e-23

TABLE 5.2

Errors for Example 4.3 using the fifth-order Dormand-Prince Runge-Kutta scheme with a fourth-order continuous extension and one SRM iteration.

REMARK 5.1. *From the practical point of view, it is not necessary to consider the limit as the number of iterations σ goes to infinity. For the case of (5.3), after only a few iterations or even only one iteration of the PSRM, the WR solution is already very close to the true solution. Nor is it necessary to consider the convergence on the whole interval $[0,T]$ of t. It is possible (e.g., (5.3)) to design a WR algorithm that is convergent on a part (e.g., $[\delta,T]$) of the interval but not convergent in the other part (e.g., $[0,\delta)$) of the interval. And the solution in the interval $[0,\delta)$ may be obtained afterwards using the solution in the interval $[\delta,T]$. For example, we prove in Theorem 2.1 that the WR algorithm (5.3) converges in the interval away from the initial layer region, i.e., the interval $[\delta,T]$, where $\delta > O(\epsilon)$. The solution in $[0,\delta)$ can be obtained by solving the following terminal-value problem in a small interval $[0,2\delta]$ as in [4]:*

$$(x^\sigma)' = f(x^\sigma,t) - B(x^\sigma,t)y^\sigma, \tag{5.7}$$

where

$$y^\sigma = y^{\sigma-1} - \frac{1}{\epsilon}T(x^\sigma,t)g(x^\sigma,t), \tag{5.8}$$

subject to the terminal condition $x^\sigma(2\delta) = x(2\delta)$, where $x(2\delta)$ is taken from the previous calculation. Note that we change the sign in front of $\frac{1}{\epsilon}T(x^\sigma,t)g(x^\sigma,t)$ from positive in (2.4) to negative in (5.8) so that the problem (5.7), (5.8) has a terminal layer instead of an initial layer as in the regularized problem (2.3), (2.4). We can thus obtain a good approximation of the solution in the interval $[0,\delta)$ because the interval is outside of the terminal-layer region $(\delta,2\delta]$.

REMARK 5.2. *A nice discussion appears in [30] on how to split a DAE system for dynamic iteration based on the case of linear constant-coefficient DAEs. One result is that if it can be assumed that the number of state variables does not change when the system is split for the iteration, the nilpotent structure of the original matrix pencil should be preserved in the splitting in order to assure convergence to the correct solution (cf. Theorem 3 of [30]). For the splitting of (5.3), the original index-2 system is recast into an index-1 system, thereby changing the nilpotent structure of the original matrix pencil and the number of state variables (according to the definition in [30]). So in general, if we allow a different number of state variables, the nilpotent structure can be changed in the waveform splitting, and the WR algorithm can still converge rapidly to the correct solution of the original problem (although the convergence is not on the whole interval of t).*

Acknowledgment The authors would like to thank the reviewers for their valuable comments. The first author would like to thank Dr. Ben Leimkuhler for pointing out the possible relation between the SRM and WR, and providing a few references.

REFERENCES

[1] F. Amirouche, *Computational Methods in Multibody dynamics*, Prentice-Hall, 1992.

[2] U. Ascher, *On some difference schemes for singular singularly-perturbed boundary value problems*, Numer. Math. 46 (1985), 1-30.

[3] U. Ascher, R. Mattheij, and R. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equation*, Prentice-Hall, 1988.

[4] U. Ascher and P. Lin, *Sequential regularization methods for higher index DAEs with constraint singularities: The linear index-2 case*, SIAM J. Numer. Anal. 33 (1996), 1921-1940.

[5] U. Ascher and P. Lin, *Sequential regularization methods for nonlinear higher index differential-algebraic equations*, SIAM J. Sci. Comput. 18 (1997), 160-181.

[6] U. Ascher and P. Lin, *Sequential regularization methods for simulating mechanical systems with many closed loops*, SIAM J. Sci. Comput., **24** (4) (2000), 1244-1262.

[7] U. Ascher and L. Petzold, *Projected implicit Runge-Kutta methods for differential-algebraic equations*, SIAM J. Numer. Anal. 28 (1991), 1097-1120.

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

[9] J. Bahi, E. Griepentrog, and J.-C. Miellou, *Parallel treatment of a class of differential-algebraic systems*, SIAM J. Numer. Anal., **23** (5) (1996), 1969-1980.

[10] J. Baumgarte, *Stabilization of constraints and integrals of motion in dynamical systems*, Comp. Math. Appl. Mech. Eng., 1 (1972), 1-16.

[11] E. Bayo and A. Avello, *Singularity-free augmented Lagrangian algorithms for constrained multibody dynamics*, J. Nonlinear Dynamics, 5 (1994), 209-231.

[12] A. Bellen, Z. Jackiewicz, and M. Zennaro, *Contractivity of waveform relaxation Runge-Kutta methods for dissipative systems in the maximum norm*, SIAM J. Numer. Anal., 31 (1994), 499-523.

[13] K. Brenan, S. Campbell, and L. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, Elsevier Science Publishers, 1989.

[14] K. Burrage, *Parallel and Sequential Methods for Ordinary Differential Equations*, Clarendon Press, Oxford, 1995.

[15] J.R. Dormand and P.J. Prince, *A family of embedded Runge-Kutta formulae*, J. Comp. Appl. Math., 6 (1980), 19-26.

[16] A. Frommer and B. Pohl, *A comparison result for multisplitting based on overlapping blocks and its application to waveform relaxation methods*, Numer. Linear Algebra Appl., 2 (1995), 335-346.

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

[18] E. Hairer, S.P. Norsett, and G. Wanner, *Solving Ordinary Differential Equations I*, Springer-Verlag, 1987.

[19] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II*, Springer-Verlag, 1991.

[20] M. Hanke, *Regularization of differential-algebraic equations revisited*, Preprint Nr. 92-19, Humboldt-Univ. Berlin, Fachbereich Mathematik, 1992.

[21] P. J. van der Houwen and W. A. van der Veen, *Waveform relaxation methods for implicit differential equations*, Advances in Computational Math., 7 (1997), 183-197.

[22] Z. Jackiewicz and M. Kwapisz, *Convergence of waveform relaxation methods for differential-algebraic systems*, SIAM J. Numer. Anal. 33 (1996), 2303-2317.

[23] L. Kalachev and R. O'Malley, Jr., *The regularization of linear differential-algebraic equations*, SIAM J. Math. Appl., 1996.

[24] L. Kalachev and R. O'Malley, Jr., *Boundary value problems for differential-algebraic equations*, Num. Func. Anal. Appl. 16 (1995), 363-378.

[25] B. Leimkuhler, U. Miekkala, and O. Nevanlinna, *Waveform relaxation for linear RC-circuits*, IMPACT, July, 1991.

[26] E. Lelarasmee, A. Ruehli, and A. Sangiovanni-Vincentelli, *The waveform relaxation method for time domain analysis of large scaled integrated circuits*, IEEE Trans. on CAD of IC and Syst., 1 (1982), 131-145.

[27] P. Lin, *A sequential regularization method for time-dependent incompressible Navier-Stokes equations*, SIAM J. Numer. Anal. 34 (1997), 1051-1071.

[28] P. Lin and D. Yang, *An iterative perturbation method for the pressure equation in the simulation of miscible displacement in porous media*, SIAM J. Sci. Comput. 19 (1998).

[29] P. Lotstedt, *On a penalty function method for the simulation of mechanical systems subject to constraints*, Royal Inst. of Tech. TRITA-NA-7919, Stockholm, 1979.

[30] U. Miekkala, *Dynamic iteration methods applied to linear DAE systems*, J. Comput. Appl. Math., 25 (1989), 133-151.

[31] U. Miekkala and O. Nevanlinna, *Convergence of dynamic iterations for initial value problems*, SIAM J. Sci. Stat. Comp., 8 (1987), 459-482.

[32] D. Mitra, *Asynchronous relaxations for the numerical solution of differential equations by parallel processes*, SIAM J. Sci. Stat. Comput., **8** (1) (1987), 43-58.

[33] K. Park and J. Chiou, *Stabilization of computational procedures for constrained dynamical systems*, J. of Guidance 11 (1988), 365-370.

[34] J. Simeons and S. Vandewalle, *Waveform relaxation with fast direct methods as preconditioner*, SIAM J. Sci. Comput., 21 (2000), 1755-1773.

[35] J. White and A. L. Sangiovanni-Vincentelli, *Relaxation Techniques for the Simulation of VLSI Circuits*, Kluwer Academic Publishers, Boston, 1987.

[36] M. Zennaro, *Natural continuous extensions of Runge-Kutta methods*, Math. Comp., 46 (1986), 119-133.