# Simple ECEM Algorithms Using Function Values Only 

Philsu Kim and Sang Dong Kim*<br>Department of Mathematics, Kyungpook National University, Daegu, 702-701, Korea<br>e-mail: kimps@knu.ac.kr and skim@knu.ac.kr<br>Eunjung Lee<br>Department of Computational Science and Engineering, Yonsei University, Korea<br>e-mail: enujunglee@yonsei.ac.kr

Abstract. In this paper, we improve the error corrected Euler method(ECEM) introduced in [11] by evaluating function values only at local nodes in each time interval. As a result, one can avoid computations of Jacobian matrices on each time interval so that the algorithms become simpler to implement in solving various class of time dependent differential equations numerically. The proposed ECEM formula resembles to the Runge-Kutta method in its representations but both methods have different characteristic properties.

## 1. Introduction

Among well known many numerical methods for solving initial value problems(see [5, 6], [7]-[10] and [14]-[24], for example), recently the ECEM (error corrected Euler method) was reported in [11] to solve stiff initial value problems by aiming at two main goals; to avoid iteration steps for nonlinear discrete systems and to provide a good stability as implicit methods possess. The basic concept of ECEM algorithm is to obtain the next approximation by the forward Euler approximation plus a local correction term in each time interval.

The original ECEM algorithm introduced in [11] requires to compute the local Jacobian matrices and to solve local linear systems to get correction terms on each time steps. Evaluation of the Jabobian of the given function is a very expensive process and if one uses an approximation for Jacobian evaluation, again it may result in loosing some information. In this paper, we propose a rewritten ECEM algorithm which only uses the previous time step's data and function evaluations at

[^0]intermediate stages on each time step. The basic idea is the same with the original ECEM algorithm in [11] but we rearranged the algorithm and found a formula so that the Jabobian computation is not needed.

The proposed ECEM method can be considered as a one-step multistage linear method and the formula is similar to the well-known Rung-Kutta (RK) method. The explicit RK method is one of the most widely used numerical scheme in solving ordinary differential equations because it is very easy to implement and can provide a faster convergence (see [1], [2], [5], [6]). However, it is generally known that explicit RK methods are not suitable for solving stiff equations since their absolute stability region is small. On the other hand, the implicit RK method is unconditional stable while it needs some extra iteration process.

Although ECEM and RK have similar formulations and share the same order of convergence, they actually have several major differences as follows:

- To generate the next stage approximation, ECEM only uses $K_{0}=f\left(t_{m}, y_{m}\right)$ and function evaluations at local Chebyshev-Gauss-Lobatto-points in each time step while RK requires to use all previous intermediate information at any local nodes in each time step which demands a lot of storage specially in dealing with higher-dimensional problems.
- The weights (the coefficients) of ECEM formula depend on the function $f(t, y)$ while the ones of RK are constants. Hence, computing weights in ECEM needs an extra work than RK does. The weights by a numerical quadrature explained for ECEM are different from the usual CGL quadrature weights.
- On the stability regions of two methods for Dahlquist's test problem $\frac{d y}{d t}=\lambda y$, the ECEM possesses $A(\alpha)$ stability and almost $L$ stability even if it is an explicit time stepping method while the explicit RK methods possess limited stability regions. The stability regions of ECEM almost coincides with the implicit RK methods.

This paper is organized as follows. In section 2, the general ECEM algorithm is reviewed and stated with only function values. Further, it is shown that ECEM can be explained by a numerical quadrature using CGL nodes. In section 3 , the $2-$ stage ECEM2 algorithm is stated and its stability analysis for $\frac{d y}{d t}=\lambda(y-g)+\frac{d g}{d t}$ is compared with the explicit RK2 method. In the following two sections 4 and 5 , the $p=3,4$ stage ECEM algorithms are described in terms of function evaluations fully with their derivations. The weights for ECEM are given completely. In section 6 , the explicit RK methods and ECEM algorithms are compared. In final section, we derive some conclusions and further research topics.

## 2. General ECEM Algorithm

The time discretization technique ECEM was first introduced in [11]. We will review the necessary notations and definitions to propose a new ECEM formula
in terms of function evaluations only. Therefore, an interested reader should refer to [11]. The derivation of ECEM is based on the Euler Polygon and the local Chebyshev-Gauss-Lobatto (CGL) collocation points aiming at solving stiff initial value problems (IVP) such that

$$
\begin{equation*}
\frac{d y}{d t}=f(t, y(t)), \quad y\left(t_{0}\right)=y_{0} . \tag{2.1}
\end{equation*}
$$

Let $s_{j}=-\cos \frac{\pi j}{p}, 0 \leq j \leq p$ be the CGL points in $[-1,1]$. The Euler polygon on the interval $\left[t_{m}, t_{m+1}\right]$ for a given approximation $y_{m}$ at $t_{m}$ is known as

$$
\begin{equation*}
y(t)=y_{m}+\left(t-t_{m}\right) f\left(t_{m}, y_{m}\right) . \tag{2.2}
\end{equation*}
$$

Using the above Euler polygon $y(t)$, we define

$$
\begin{equation*}
\varphi(s)=\frac{1}{\tau^{2}}\left[f\left(t_{s}, y\left(t_{s}\right)+\tau^{2}\right)-f\left(t_{s}, y\left(t_{s}\right)\right)\right] \tag{2.3}
\end{equation*}
$$

where $\tau=t_{m+1}-t_{m}$ and

$$
\begin{equation*}
t_{s}=t_{m}+\frac{1+s}{2} \tau, \quad \text { for } s \in[-1,1] . \tag{2.4}
\end{equation*}
$$

Note that the finite difference approximation (2.3) for $f_{y}$ does not violate the convergence of ECEM. Let us define the matrix $\mathcal{A}$ and the vector $\mathcal{F}$ as

$$
\begin{equation*}
\mathcal{A}:=\left(a_{j k}\right)_{j, k=1, \cdots, p}, \quad p \leq 4 \tag{2.5}
\end{equation*}
$$

with

$$
\begin{equation*}
a_{j k}=\frac{d l_{k}}{d x}\left(s_{j}\right)-\frac{\tau}{2} \varphi\left(s_{j}\right) \delta_{j k}, \quad j, k=1, \cdots, p \tag{2.6}
\end{equation*}
$$

and

$$
\begin{equation*}
\mathcal{F}:=\left[F\left(t_{s_{1}}\right), \cdots, F\left(t_{s_{p}}\right)\right] \tag{2.7}
\end{equation*}
$$

with

$$
\begin{equation*}
F\left(t_{s_{k}}\right)=f\left(t_{s_{k}}, y\left(t_{s_{k}}\right)\right)-f\left(t_{m}, y_{m}\right), \quad k=1,2, \cdots, p . \tag{2.8}
\end{equation*}
$$

With the last component $\hat{\beta}$ of a solution $\mathbf{d}$ to

$$
\begin{equation*}
\mathcal{A} \mathbf{d}=\frac{\tau}{2} \mathcal{F} \tag{2.9}
\end{equation*}
$$

the $p$-th order ECEM $(p=2,3,4)$ is known as

$$
\begin{equation*}
y_{m+1}=y_{m}+\tau f\left(t_{m}, y_{m}\right)+\hat{\beta} \tag{2.10}
\end{equation*}
$$

This algorithm (2.10) now can be rewritten in terms of function values only with appropriate weights which actually are obtained by the last row of the matrix $\mathcal{A}^{-1}$. We address this observation in the following theorem.

Theorem 2.1. Let $\left(a_{1} a_{2} a_{3} a_{4}\right)$ be the last row vector of the matrix $\mathcal{A}^{-1}$. Then ECEMk $(k=2,3,4)$ can be written as
$y_{m+1}=y_{m}+a_{0} f\left(t_{m}, y_{m}\right)+\frac{\tau}{2} \sum_{j=1}^{4} a_{j} f\left(t_{m}+\frac{1+s_{j}}{2} \tau, y_{m}+\frac{1+s_{j}}{2} \tau f\left(t_{m}, y_{m}\right)\right)$
$(2.11)=y_{m}+a_{0} f\left(t_{m}, y_{m}\right)+\frac{\tau}{2} \sum_{j=1}^{4} a_{j} f\left(t_{s_{j}}, y\left(t_{s_{j}}\right)\right)$,
where $a_{0}$ satisfies

$$
\begin{equation*}
a_{0}=\tau-\frac{\tau}{2}\left(a_{1}+a_{2}+a_{3}+a_{4}\right) \tag{2.12}
\end{equation*}
$$

Note that $a_{3}=a_{4}=0, s_{j}=-\cos \frac{\pi j}{2}, j=1,2$ for ECEM2, $a_{4}=0, s_{j}=$ $-\cos \frac{\pi j}{3}, j=1,2,3$ for ECEM3 and $s_{j}=-\cos \frac{\pi j}{4}, j=1,2,3,4$ for ECEM4.
Proof. Note that from (2.9), the last component $\hat{\beta}$ of $\mathbf{d}$ can be written as

$$
\begin{equation*}
\hat{\beta}=\frac{\tau}{2}\left(a_{1} F\left(t_{s_{1}}\right)+a_{2} F\left(t_{s_{2}}\right)+a_{3} F\left(t_{s_{3}}\right)+a_{4} F\left(t_{s_{N}}\right)\right) \tag{2.13}
\end{equation*}
$$

where the vector $\left[a_{1} a_{2}, a_{3}, a_{4}\right]$ is the last row of the matrix $\mathcal{A}^{-1}$. Therefore, one may have the conclusion by combining (2.13) with (2.10) and (2.8).

The convergence of ECEM algorithms with (2.3) is proven in [11]. For reader's convenience, it is stated in the next theorem.

Theorem 2.2. Let $y(t)$ be the exact solution to (2.1) and $\left\{y_{m}\right\}$ be an approximation generated by (2.10) at time $t=t_{m}$. Under the assumption of $\sup _{t, y}\left\|f_{y}(t, y(t))\right\|<$ $\infty$, the actual error $e_{m}=y\left(t_{m}\right)-y_{m}$ satisfies for sufficiently small time size $\tau$

$$
\begin{equation*}
e_{m}=O\left(\frac{(1+C \tau)^{3 m}-1}{(1+C \tau)^{3}-1} \tau^{\min \{n+1,5\}}\right), \quad m \geq 0 \tag{2.14}
\end{equation*}
$$

where the positive constant $C$ is a generic constant independent of time size $\tau$, time step $m$ and the final time $T$, and $n$ is chosen as 2,3 or 4 which is the order $N=n$ of matrix (2.5).
Proof. See (4.13) in Theorem 4.4 of [11].

Under the assumption of Theorem 2.2 with $n=2,3,4$, it follows that from
(2.1), (2.14) and (2.11)

$$
\begin{align*}
\int_{t_{m}}^{t_{m+1}} f(t, y) d t & =\int_{t_{m}}^{t_{m+1}} y^{\prime}(t) d t \\
& =y\left(t_{m+1}\right)-y\left(t_{m}\right) \\
& =y\left(t_{m+1}\right)-y\left(t_{m}\right)+e_{m+1}-e_{m} \\
& =y_{m+1}-y_{m}+O\left(\tau^{\min \{n+1,5\}}\right) \\
& =a_{0} f\left(t_{m}, y_{m}\right)+\frac{\tau}{2} \sum_{j=1}^{4} a_{j} f\left(t_{s_{j}}, y\left(t_{s_{j}}\right)\right)+O\left(\tau^{\min \{n+1,5\}}\right) \tag{2.15}
\end{align*}
$$

From this (2.15), it is evident that the ECEM algorithms can be explained from numerical quadrature of (2.1). But the weights $a_{0}$ and $\frac{a_{i}}{2} \tau(i=2, \cdots, n)$ should be calculated from the function of $f(t, y)$ with the CGL nodes. This numerical quadrature is distinguished from the well known CGL quadrature rule (see [4]) which can be used for RK methods(see Butcher's table in [5], [6] for example). Now we are ready to propose a new formula for ECEM algorithm. They are explicitly presented in the following three sections.

## 3. Second-order ECEM2 Algorithm

In the section, we rewrite the ECEM2 algorithm for easy and convenient implementations using only function values of $f$. ECEM2 has a second order convergence. The CGL points are taken as $s_{0}=-1, s_{1}=0, s_{2}=1$ and the Chebyshev Lagrange interpolation quadratic polynomials using these three CGL points are

$$
\begin{equation*}
l_{0}(x)=\frac{1}{2} x(x-1), \quad l_{1}(x)=-x^{2}+1, \quad l_{2}(x)=\frac{1}{2} x(x+1) . \tag{3.1}
\end{equation*}
$$

The matrix $\mathcal{A}$ in (2.5) becomes

$$
\mathcal{A}=\left[\begin{array}{cc}
-\frac{\tau}{2} \varphi\left(s_{1}\right) & \frac{1}{2}  \tag{3.2}\\
-2 & \frac{3}{2}-\frac{\tau}{2} \varphi\left(s_{2}\right)
\end{array}\right]
$$

where

$$
\begin{aligned}
& \varphi\left(s_{1}\right)=\frac{1}{\tau^{2}}\left[f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} f\left(t_{m}, y_{m}\right)+\tau^{2}\right)-f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} f\left(t_{m}, y_{m}\right)\right)\right] \\
& \varphi\left(s_{2}\right)=\frac{1}{\tau^{2}}\left[f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)+\tau^{2}\right)-f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)\right)\right]
\end{aligned}
$$

and the vector $\mathcal{F}$ in (2.7) becomes

$$
\begin{align*}
F\left(t_{s_{1}}\right) & =f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} f\left(t_{m}, y_{m}\right)\right)-f\left(t_{m}, y_{m}\right)  \tag{3.3}\\
F\left(t_{s_{2}}\right) & =f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)\right)-f\left(t_{m}, y_{m}\right)
\end{align*}
$$

Then, solving (2.9) for $\mathbf{d}$ using (3.2) and (3.3), one may get

$$
\mathbf{d}=\frac{\tau}{2} \frac{1}{\operatorname{det}(\mathcal{A})}\left[\begin{array}{c}
\left(\frac{3}{2}-\frac{\tau}{2} \varphi\left(s_{2}\right)\right) F\left(t_{s_{1}}\right)-\frac{1}{2} F\left(t_{s_{2}}\right)  \tag{3.4}\\
2 F\left(t_{s_{1}}\right)-\frac{\tau}{2} \varphi\left(s_{1}\right) F\left(t_{s_{2}}\right)
\end{array}\right]
$$

where

$$
\begin{equation*}
\operatorname{det}(\mathcal{A})=\frac{\tau}{2} \varphi\left(s_{1}\right)\left(\frac{\tau}{2} \varphi\left(s_{2}\right)-\frac{3}{2}\right)+1 \tag{3.5}
\end{equation*}
$$

The second-order ECEM2 is the scheme of the Forward Euler plus the second component of the vector $\mathbf{d}$ which is

$$
\begin{align*}
y_{m+1} & =y_{m}+\tau f\left(t_{m}, y_{m}\right)+\frac{\tau}{2} \frac{1}{\operatorname{det}(\mathcal{A})}\left(2 F\left(t_{s_{1}}\right)-\frac{\tau}{2} \varphi\left(s_{1}\right) F\left(t_{s_{2}}\right)\right)  \tag{3.6}\\
& =y_{m}+\alpha f\left(t_{m}, y_{m}\right)+\beta f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} f\left(t_{m}, y_{m}\right)\right) \\
& +\gamma f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)\right)
\end{align*}
$$

where the weights $\alpha, \beta$ and $\gamma$ are

$$
\begin{equation*}
\alpha=\tau-(\beta+\gamma), \quad \beta=\frac{\tau}{\operatorname{det}(\mathcal{A})}, \quad \gamma=-\frac{\tau^{2}}{4} \frac{\varphi\left(s_{1}\right)}{\operatorname{det}(\mathcal{A})} . \tag{3.7}
\end{equation*}
$$

Combining all above these, the second-order ECEM2 can be read as follows:

$$
\begin{align*}
K_{1} & =f\left(t_{m}, y_{m}\right) \\
K_{2} & =f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} K_{1}\right) \\
K_{3} & =f\left(t_{m+1}, y_{m}+\tau K_{1}\right) \\
\varphi\left(s_{1}\right) & =\frac{1}{\tau^{2}}\left[f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} K_{1}+\tau^{2}\right)-K_{2}\right] \\
\varphi\left(s_{2}\right) & =\frac{1}{\tau^{2}}\left[f\left(t_{m+1}, y_{m}+\tau K_{1}+\tau^{2}\right)-K_{3}\right] \\
\operatorname{det}(\mathcal{A}) & =\frac{\tau}{2} \varphi\left(s_{1}\right)\left(\frac{\tau}{2} \varphi\left(s_{2}\right)-\frac{3}{2}\right)+1  \tag{3.8}\\
\alpha & =\tau-(\beta+\gamma) \\
\beta & =\frac{\tau}{\operatorname{det}(\mathcal{A})} \\
\gamma & =-\frac{\tau^{2}}{4} \frac{\varphi\left(s_{1}\right)}{\operatorname{det}(\mathcal{A})} \\
y_{m+1} & =y_{m}+\alpha K_{1}+\beta K_{2}+\gamma K_{3} . \tag{3.9}
\end{align*}
$$

The next question is when and how the ECEM2 algorithm occurs break-down. Definitely, if $\operatorname{det}(\mathcal{A})=0$ of the matrix $\mathcal{A}$ in (2.5) then ECEM2 should be broken
down. In this case, one may not get proper weights $\alpha, \beta$ and $\gamma$ in ECEM2. Let us put this observation as theorem:

Theorem 3.1. The algorithm will break down if the matrix $\mathcal{A}$ in (2.5) is singular.

We now derive the stability function $S_{2}(z)$ for ECEM2 for the case

$$
\begin{equation*}
f(t, y)=\lambda(y-g)+\frac{d g}{d t} \tag{3.10}
\end{equation*}
$$

in (2.1) which explains also Dalquist's test problem when $g=0$ (see [3]).
For this problem, one may get

$$
\begin{aligned}
K_{1} & =\lambda\left(y_{m}-g\right)+g^{\prime}(t), \\
K_{2} & =\lambda\left(y_{m}+\frac{\tau}{2} K_{1}-g\right)+g^{\prime}(t), \\
K_{3} & =\lambda\left(y_{m}+\tau K_{1}-g\right)+g^{\prime}(t)
\end{aligned}
$$

and

$$
\begin{equation*}
\alpha=\frac{\tau^{3} \lambda^{2}-2 \tau^{2} \lambda}{\tau^{2} \lambda^{2}-3 \tau \lambda+4}, \beta=\frac{4 \tau}{\tau^{2} \lambda^{2}-3 \tau \lambda+4} \gamma=\frac{-\tau^{2} \lambda}{\tau^{2} \lambda^{2}-3 \tau \lambda+4} . \tag{3.11}
\end{equation*}
$$

Then, applying all of these to (3.9) yields that

$$
\begin{equation*}
y_{m+1}=S_{2}(\tau \lambda) y_{m}+S_{1}(\tau \lambda) g+\tau S_{0}(\tau \lambda) \frac{d g}{d t} \tag{3.12}
\end{equation*}
$$

where

$$
\begin{equation*}
S_{2}(z)=\frac{z+4}{z^{2}-3 z+4}, \quad S_{1}(z)=\frac{z^{2}-4 z}{z^{2}-3 z+4}, \quad S_{0}(z)=\frac{4-z}{z^{2}-3 z+4} \tag{3.13}
\end{equation*}
$$

This yields that

$$
\begin{align*}
y_{m+1} & =S_{2}^{m+1}(\tau \lambda) y_{0}+\left(\sum_{k=0}^{m} S_{2}^{k}(\tau \lambda)\right) S_{1}(\tau \lambda) g+\tau\left(\sum_{k=0}^{m} S_{2}^{k}(\tau \lambda)\right) S_{0}(\tau \lambda) \frac{d g}{d t} \\
3.14) & =S_{2}^{m+1}(\tau \lambda) y_{0}+\frac{1-S_{2}^{m+1}(\tau \lambda)}{1-S_{2}(\tau \lambda)} S_{1}(\tau \lambda) g+\tau \frac{1-S_{2}^{m+1}(\tau \lambda)}{1-S_{2}(\tau \lambda)} S_{0}(\tau \lambda) \frac{d g}{d t} \tag{3.14}
\end{align*}
$$

On the other hand, the explicit RK2 method for this problem is

$$
\begin{equation*}
y_{m+1}=\hat{S}_{2}(\tau \lambda) y_{m}-\hat{S}_{1}(\tau \lambda) g+\tau \hat{S}_{0}(\tau \lambda) \frac{d g}{d t} \tag{3.15}
\end{equation*}
$$

where

$$
\begin{equation*}
\hat{S}_{2}(z)=\frac{1}{2} z^{2}+z+1, \quad \hat{S}_{1}(z)=\frac{1}{2} z^{2}+z, \quad \hat{S}_{0}(z)=\frac{1}{2}(z+1) . \tag{3.16}
\end{equation*}
$$

These two stability regions by $S_{2}(z)$ and $\hat{S}_{2}(z)$ are shown in figure 1 for this problem (3.10). According to these regions, ECEM2 shows much bigger stability region than RK2.

## 4. Third-order ECEM3 Algorithm

For the third-order ECEM3, we will use four CGL points such that $s_{0}=$ $-1, s_{1}=-\frac{1}{2}, s_{2}=\frac{1}{2}, s_{3}=1$. The Chebyshev Lagrange interpolation cubic polynomials using these CGL points are

$$
\begin{array}{ll}
l_{0}(x)=-\frac{2}{3}\left(x^{2}-\frac{1}{4}\right)(x-1), & l_{1}(x)=\frac{4}{3}\left(x^{2}-1\right)\left(x-\frac{1}{2}\right),  \tag{4.1}\\
l_{2}(x)=-\frac{4}{3}\left(x^{2}-1\right)\left(x+\frac{1}{2}\right), & l_{3}(x)=\frac{2}{3}\left(x^{2}-\frac{1}{4}\right) .
\end{array}
$$

Then the matrix $\mathcal{A}$ in (2.6) becomes

$$
\mathcal{A}=\left[\begin{array}{ccc}
\frac{1}{3}-\frac{\tau}{2} \varphi\left(s_{1}\right) & 1 & -\frac{1}{3}  \tag{4.2}\\
-1 & -\frac{1}{3}-\frac{\tau}{2} \varphi\left(s_{2}\right) & 1 \\
\frac{4}{3} & -4 & \frac{19}{6}-\frac{\tau}{2} \varphi\left(s_{3}\right)
\end{array}\right]
$$

where
$\varphi\left(s_{1}\right)=\frac{1}{\tau^{2}}\left[f\left(t_{m}+\frac{\tau}{4}, y_{m}+\frac{\tau}{4} f\left(t_{m}, y_{m}\right)+\tau^{2}\right)-f\left(t_{m}+\frac{\tau}{4}, y_{m}+\frac{\tau}{4} f\left(t_{m}, y_{m}\right)\right)\right]$
$\varphi\left(s_{2}\right)=\frac{1}{\tau^{2}}\left[f\left(t_{m}+\frac{3 \tau}{4}, y_{m}+\frac{3 \tau}{4} f\left(t_{m}, y_{m}\right)+\tau^{2}\right)-f\left(t_{m}+\frac{3 \tau}{4}, y_{m}+\frac{3 \tau}{4} f\left(t_{m}, y_{m}\right)\right)\right]$
$\varphi\left(s_{3}\right)=\frac{1}{\tau^{2}}\left[f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)+\tau^{2}\right)-f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)\right)\right]$,
and the vector $\mathcal{F}$ in (2.7) becomes

$$
\begin{align*}
F\left(t_{s_{1}}\right) & =f\left(t_{m}+\frac{\tau}{4}, y_{m}+\frac{\tau}{4} f\left(t_{m}, y_{m}\right)\right)-f\left(t_{m}, y_{m}\right) \\
F\left(t_{s_{2}}\right) & =f\left(t_{m}+\frac{3 \tau}{4}, y_{m}+\frac{3 \tau}{4} f\left(t_{m}, y_{m}\right)\right)-f\left(t_{m}, y_{m}\right)  \tag{4.3}\\
F\left(t_{s_{3}}\right) & =f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)\right)-f\left(t_{m}, y_{m}\right)
\end{align*}
$$

Solving (2.9) for the vector $\mathbf{d}$ using (4.2) and (4.3) and getting the last component of $\mathbf{d}$, we have the correction term $\hat{\beta}$ for the error correction to the Forward-Euler scheme. For this purpose, let us calculate the inverse $\mathcal{A}^{-1}$ for the matrix $\mathcal{A}$.

Lemma 4.1. For the matrix $\mathcal{A}$ in (4.2), consider its block matrix such that $\mathcal{A}=$ $\left[\begin{array}{ll}A & B \\ C & r\end{array}\right]$ where $r$ is the $(3,3)$ scalar component of $\mathcal{A}$ and the block matrix $A$ is the first principal $2 \times 2$ submatrix of $\mathcal{A}$. Then its inverse is

$$
\mathcal{A}^{-1}=\left(r-C A^{-1} B\right)^{-1}\left[\begin{array}{cc}
\left(r-C A^{-1} B\right)\left(A^{-1}+A^{-1} B C A^{-1}\right) & -A^{-1} B  \tag{4.4}\\
-C A^{-1} & 1
\end{array}\right]
$$

Proof. One may check it easily.

Because we need only the last component $\hat{\beta}$ of $\mathbf{d}$, it is enough to consider the last row of $\mathcal{A}^{-1}$. The submatrices $A$ and $C$ in (4.2) is in fact

$$
A=\left[\begin{array}{cc}
\frac{1}{3}-\frac{\tau}{2} \varphi\left(s_{1}\right) & 1  \tag{4.5}\\
-1 & -\frac{1}{3}-\frac{\tau}{2} \varphi\left(s_{2}\right)
\end{array}\right], \quad C=\left[\begin{array}{cc}
\frac{4}{3} & -4
\end{array}\right], \quad B=\left[\begin{array}{r}
-\frac{1}{3} \\
1
\end{array}\right] .
$$

The inverse $A^{-1}$ of $A$ becomes

$$
A^{-1}=\frac{1}{\operatorname{det}(A)}\left[\begin{array}{cc}
-\frac{1}{3}-\frac{\tau}{2} \varphi\left(s_{2}\right) & -1 \\
1 & \frac{1}{3}-\frac{\tau}{2} \varphi\left(s_{1}\right)
\end{array}\right]
$$

where

$$
\operatorname{det}(A)=\frac{8}{9}+\frac{\tau}{6}\left[\varphi\left(s_{1}\right)-\varphi\left(s_{2}\right)\right]+\frac{\tau^{2}}{4} \varphi\left(s_{1}\right) \varphi\left(s_{2}\right)
$$

A simple calculation shows us

$$
\begin{equation*}
C A^{-1}=\frac{1}{\operatorname{det}(A)}\left[-\frac{40}{9}-\frac{2 \tau}{3} \varphi\left(s_{2}\right) \quad-\frac{8}{3}+2 \tau \varphi\left(s_{1}\right)\right] \tag{4.6}
\end{equation*}
$$

and
(4.7) $\hat{r}^{-1}:=r-C A^{-1} B=\frac{19}{6}-\frac{\tau}{2} \varphi\left(s_{3}\right)-\frac{1}{\operatorname{det}(A)}\left[-\frac{32}{27}+\tau\left(2 \varphi\left(s_{1}\right)+\frac{2}{9} \varphi\left(s_{2}\right)\right)\right]$.

Then, using (4.6), (4.7) and (4.4), the third component $\hat{\beta}=\frac{\tau}{2}\left(\mathcal{A}^{-1} \mathcal{F}\right)_{3}$ becomes

$$
\begin{align*}
\hat{\beta} & =\frac{\tau}{2} \frac{\hat{r}}{\operatorname{det}(A)}\left(\frac{40}{9}+\frac{2 \tau}{3} \varphi\left(s_{2}\right)\right) F\left(t_{s_{1}}\right)  \tag{4.8}\\
& +\frac{\tau}{2} \frac{\hat{r}}{\operatorname{det}(A)}\left(\frac{8}{3}-2 \tau \varphi\left(s_{1}\right)\right) F\left(t_{s_{2}}\right)+\frac{\tau}{2} \hat{r} F\left(t_{s_{3}}\right)
\end{align*}
$$

Finally, combining all above these, we have the third-order ECEM3 read as follows:

$$
\begin{align*}
& K_{1}= f\left(t_{m}, y_{m}\right) \\
& K_{2}= f\left(t_{m}+\frac{\tau}{4}, y_{m}+\frac{\tau}{4} K_{1}\right) \\
& K_{3}= f\left(t_{m}+\frac{3 \tau}{4}, y_{m}+\frac{3 \tau}{4} K_{1}\right) \\
& K_{4}= f\left(t_{m+1}, y_{m}+\tau K_{1}\right) \\
& \varphi\left(s_{1}\right)= \frac{1}{\tau^{2}}\left[f\left(t_{m}+\frac{\tau}{4}, y_{m}+\frac{\tau}{4} K_{1}+\tau^{2}\right)\right. \\
&\left.-f\left(t_{m}+\frac{\tau}{4}, y_{m}+\frac{\tau}{4} K_{1}\right)\right] \\
& \varphi\left(s_{2}\right)= \frac{1}{\tau^{2}}\left[f\left(t_{m}+\frac{3 \tau}{4}, y_{m}+\frac{3 \tau}{4} K_{1}+\tau^{2}\right)\right. \\
&\left.-f\left(t_{m}+\frac{3 \tau}{4}, y_{m}+\frac{3 \tau}{4} K_{1}\right)\right] \\
& \varphi\left(s_{3}\right)= \frac{1}{\tau^{2}}\left[f\left(t_{m+1}, y_{m}+\tau K_{1}+\tau^{2}\right)-f\left(t_{m+1}, y_{m}+\tau K_{1}\right)\right] \\
& \operatorname{det}(A)= \frac{8}{9}+\frac{\tau}{6}\left[\varphi\left(s_{1}\right)-\varphi\left(s_{2}\right)\right]+\frac{\tau^{2}}{4} \varphi\left(s_{1}\right) \varphi\left(s_{2}\right) \\
& \hat{r}^{-1}= \frac{19}{6}-\frac{\tau}{2} \varphi\left(s_{3}\right)-\frac{1}{\operatorname{det}(A)}\left[-\frac{32}{27}+\tau\left(2 \varphi\left(s_{1}\right)+\frac{2}{9} \varphi\left(s_{2}\right)\right)\right] \\
& \alpha= \tau-(\beta+\gamma+\delta) \\
& \beta= \frac{\tau \hat{\gamma}}{\operatorname{det}(A)}\left[\frac{20}{9}+\frac{\tau}{3} \tau \varphi\left(s_{2}\right)\right] \\
& \gamma= \frac{\tau \hat{\gamma}}{\operatorname{det}(A)}\left[\frac{4}{3}-\tau \varphi\left(s_{2}\right)\right] \\
& \delta=\frac{\tau}{2} \hat{\gamma} \\
&(4.10)=y_{m}+\alpha K_{1}+\beta K_{2}+\gamma K_{3}+\delta K_{4} \tag{4.10}
\end{align*}
$$

The same question arises when and how the ECEM3 algorithm occurs breakdown. Definitely, if $\operatorname{det}(A)=0$ where the matrix $A$ in 4.5) is the principal submatrix of $\mathcal{A}$ in (4.2) then ECEM3 should be broken down. In this case, one may not get proper weights $\alpha, \beta, \gamma$ and $\delta$ in ECEM3 algorithm stated in (4.9). Let us put this observation as theorem:

Theorem 4.2. The algorithm will break down if the $2 \times 2$ submatrix $A$ of $\mathcal{A}$ is singular.

## 5. Fourth-order ECEM4 Algorithm

For the fourth-order ECEM4, we use five CGL points given by $s_{0}=-1, s_{1}=$ $-\frac{1}{\sqrt{2}}, s_{2}=0, s_{3}=\frac{1}{\sqrt{2}}, s_{4}=1$. The Chebyshev Lagrange interpolation polynomials using these CGL points are

$$
\begin{aligned}
& l_{0}(x)=x\left(x^{2}-\frac{1}{2}\right)(x-1) \\
& l_{1}(x)=-2 x\left(x^{2}-1\right)\left(x-\frac{1}{\sqrt{2}}\right), \quad l_{2}(x)=2\left(x^{2}-1\right)\left(x^{2}-\frac{1}{2}\right) \\
& l_{3}(x)=-2 x\left(x^{2}-1\right)\left(x+\frac{1}{\sqrt{2}}\right), \quad l_{4}(x)=x(x+1)\left(x^{2}-\frac{1}{2}\right)
\end{aligned}
$$

The matrix $\mathcal{A}$ in (2.6) becomes

$$
\mathcal{A}=\left[\begin{array}{cccc}
\frac{1}{\sqrt{2}}-\frac{\tau}{2} \varphi\left(s_{1}\right) & \sqrt{2} & -\frac{1}{\sqrt{2}} & 1-\frac{1}{\sqrt{2}}  \tag{5.1}\\
-\sqrt{2} & -\frac{\tau}{2} \varphi\left(s_{2}\right) & \sqrt{2} & -\frac{1}{2} \\
\frac{1}{\sqrt{2}} & -\sqrt{2} & -\frac{1}{\sqrt{2}}-\frac{\tau}{2} \varphi\left(s_{3}\right) & 1+\frac{1}{\sqrt{2}} \\
-4\left(1-\frac{1}{\sqrt{2}}\right) & 2 & -4\left(1+\frac{1}{\sqrt{2}}\right) & \frac{11}{2}-\frac{\tau}{2} \varphi\left(s_{4}\right)
\end{array}\right]
$$

where

$$
\begin{aligned}
\varphi\left(s_{1}\right)= & \frac{1}{\tau^{2}}\left[f\left(t_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right) f\left(t_{m}, y_{m}\right)+\tau^{2}\right)\right. \\
& \left.-f\left(t_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right) f\left(t_{m}, y_{m}\right)\right)\right] \\
\varphi\left(s_{2}\right)= & \frac{1}{\tau^{2}}\left[f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} f\left(t_{m}, y_{m}\right)+\tau^{2}\right)\right. \\
& \left.-f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} f\left(t_{m}, y_{m}\right)\right)\right] \\
\varphi\left(s_{3}\right)= & \frac{1}{\tau^{2}}\left[f\left(t_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right) f\left(t_{m}, y_{m}\right)+\tau^{2}\right)\right. \\
& \left.\quad-f\left(t_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right) f\left(t_{m}, y_{m}\right)\right)\right] \\
\varphi\left(s_{4}\right)= & \frac{1}{\tau^{2}}\left[f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)+\tau^{2}\right)-f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)\right)\right] .
\end{aligned}
$$

The vector $\mathcal{F}$ in (2.7) becomes

$$
\begin{align*}
& F\left(t_{s_{1}}\right)=f\left(t_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right) f\left(t_{m}, y_{m}\right)\right)-f\left(t_{m}, y_{m}\right) \\
& F\left(t_{s_{2}}\right)=f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} f\left(t_{m}, y_{m}\right)\right)-f\left(t_{m}, y_{m}\right)  \tag{5.2}\\
& F\left(t_{s_{1}}\right)=f\left(t_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right) f\left(t_{m}, y_{m}\right)\right)-f\left(t_{m}, y_{m}\right) \\
& F\left(t_{s_{4}}\right)=f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)\right)-f\left(t_{m}, y_{m}\right)
\end{align*}
$$

Solving (2.9) for the vector $\mathbf{d}$ using (5.1) and (5.2) and getting the last component of $\mathbf{d}$, we have the correction term $\hat{\beta}$ for the error correction to the Forward-Euler scheme. Note that we only need the fourth component of $\frac{\tau}{2} \mathcal{A}^{-1} \mathcal{F}$. Hence, it is enough to calculate the fourth row of $\mathcal{A}^{-1}$. For this purpose, let us rewrite the matrix $\mathcal{A}$ in a block matrix form such that

$$
\mathcal{A}=\left[\begin{array}{ll}
A & B  \tag{5.3}\\
C & D
\end{array}\right]
$$

where
(5.4) $A=\left[\begin{array}{cc}\frac{1}{\sqrt{2}}-\frac{\tau}{2} \varphi\left(s_{1}\right) & \sqrt{2} \\ -\sqrt{2} & -\frac{\tau}{2} \varphi\left(s_{2}\right)\end{array}\right] \quad B=\left[\begin{array}{cc}-\frac{1}{\sqrt{2}} & 1-\frac{1}{\sqrt{2}} \\ \sqrt{2} & -\frac{1}{2}\end{array}\right]$
$(5.5) C=\left[\begin{array}{cc}\frac{1}{\sqrt{2}} & -\sqrt{2} \\ -4\left(1-\frac{1}{\sqrt{2}}\right) & 2\end{array}\right] \quad D=\left[\begin{array}{cc}-\frac{1}{\sqrt{2}}-\frac{\tau}{2} \varphi\left(s_{3}\right) & 1+\frac{1}{\sqrt{2}} \\ -4\left(1+\frac{1}{\sqrt{2}}\right) & \frac{11}{2}-\frac{\tau}{2} \varphi\left(s_{4}\right)\end{array}\right]$

Lemma 5.1. For the matrix $\mathcal{A}$ in (5.1), its inverse is

$$
\mathcal{A}^{-1}=\left[\begin{array}{cc}
A^{-1}+A^{-1} B\left(D-C A^{-1} B\right)^{-1} C A^{-1} & -A^{-1} B\left(D-C A^{-1} B\right)^{-1}  \tag{5.6}\\
-\left(D-C A^{-1} B\right)^{-1} C A^{-1} & \left(D-C A^{-1} B\right)^{-1}
\end{array}\right]
$$

Proof. One may check it easily.
From a series computation, it follows that

$$
C A^{-1}=\frac{1}{\operatorname{det}(A)}\left[\begin{array}{cc}
-2-\frac{\tau}{2 \sqrt{2}} \varphi\left(s_{2}\right) & -2+\frac{\tau}{\sqrt{2}} \varphi\left(s_{1}\right) \\
2 \sqrt{2}+(2-\sqrt{2}) \tau \varphi\left(s_{2}\right) & 5 \sqrt{2}-4-\tau \varphi\left(s_{1}\right)
\end{array}\right]
$$

where

$$
\begin{equation*}
\operatorname{det}(A)=\frac{\tau^{2}}{4} \varphi\left(s_{1}\right) \varphi\left(s_{2}\right)-\frac{\tau}{2 \sqrt{2}} \varphi\left(s_{2}\right)+2 \tag{5.7}
\end{equation*}
$$

and

$$
\begin{aligned}
& C A^{-1} B=\frac{1}{\operatorname{det}(A)} \times \\
& {\left[\begin{array}{cc}
-\sqrt{2}+\tau\left(\varphi\left(s_{1}\right)+\frac{\varphi\left(s_{2}\right)}{4}\right) & -1+\sqrt{2}+\tau\left(\frac{-\varphi\left(s_{1}\right)}{2 \sqrt{2}}+\left(\frac{1}{4}-\frac{1}{2 \sqrt{2}}\right) \varphi\left(s_{2}\right)\right. \\
8-4 \sqrt{2}+\tau\left(\sqrt{2} \varphi\left(s_{1}\right)+(1-\sqrt{2}) \varphi\left(s_{2}\right)\right) & \frac{-1}{\sqrt{2}}+\tau\left(\frac{\varphi\left(s_{1}\right)}{2}+(3-2 \sqrt{2}) \varphi\left(s_{2}\right)\right)
\end{array}\right]}
\end{aligned}
$$

In order to calculate $\left(D-C A^{-1} B\right)^{-1}$, let us denote the matrix as

$$
D-C A^{-1} B=\left[\begin{array}{ll}
a & b \\
c & d
\end{array}\right]_{2 \times 2}
$$

where
$a=\left(-\frac{1}{\sqrt{2}}+\sqrt{2} \frac{1}{\operatorname{det}(A)}\right)+\tau\left[-\frac{1}{\operatorname{det}(A)} \varphi\left(s_{1}\right)-\frac{1}{4} \frac{1}{\operatorname{det}(A)} \varphi\left(s_{2}\right)-\frac{1}{2} \varphi\left(s_{3}\right)\right]$,
$b=1+\frac{1}{\sqrt{2}}+(1-\sqrt{2}) \frac{1}{\operatorname{det}(A)}+\tau\left[\frac{1}{2 \sqrt{2}} \frac{1}{\operatorname{det}(A)} \varphi\left(s_{1}\right)+\frac{\sqrt{2}-1}{4} \frac{1}{\operatorname{det}(A)} \varphi\left(s_{2}\right)\right]$,
$c=-4\left(1+\frac{1}{\sqrt{2}}\right)-(8-4 \sqrt{2}) \frac{1}{\operatorname{det}(A)}+\tau\left[\sqrt{2} \frac{1}{\operatorname{det}(A)} \varphi\left(s_{1}\right)+(\sqrt{2}-1) \frac{1}{\operatorname{det}(A)} \varphi\left(s_{2}\right)\right]$,
$d=\frac{11}{2}+\frac{1}{\sqrt{2}} \frac{1}{\operatorname{det}(A)}+\tau\left[-\frac{1}{2} \frac{1}{\operatorname{det}(A)} \varphi\left(s_{1}\right)+(2 \sqrt{2}-3) \frac{1}{\operatorname{det}(A)} \varphi\left(s_{2}\right)-\frac{1}{2} \varphi\left(s_{4}\right)\right]$.
Then

$$
\left(D-C A^{-1} B\right)^{-1}=\frac{1}{a d-b c}\left[\begin{array}{cc}
d & -b  \tag{5.8}\\
-c & a
\end{array}\right] .
$$

The last row $\left(a_{41} a_{42} a_{43} a_{44}\right)$ of matrix $\mathcal{A}^{-1}$, which is the second-row of the submatrices

$$
-\left(D-C A^{-1} B\right)^{-1} C A^{-1}, \quad \text { and } \quad\left(D-C A^{-1} B\right)^{-1}
$$

in (5.6), becomes

$$
\left[\begin{array}{l}
a_{41}  \tag{5.9}\\
a_{42} \\
a_{43} \\
a_{44}
\end{array}\right]=\frac{1}{a d-b c}\left[\begin{array}{c}
\frac{-1}{\operatorname{det}(A)}\left[c\left(2+\frac{1}{2 \sqrt{2}} \tau \varphi\left(s_{2}\right)\right)+a\left(2 \sqrt{2}+(2-\sqrt{2}) \tau \varphi\left(s_{2}\right)\right)\right] \\
\frac{-1}{\operatorname{det}(A)}\left[c\left(2-\frac{1}{2} \tau \varphi\left(s_{1}\right)\right)+a\left(5 \sqrt{2}-4-\tau \varphi\left(s_{1}\right)\right)\right] \\
-c \\
a
\end{array}\right]
$$

Therefore, the fourth component $\hat{\beta}=\frac{\tau}{2}\left(\mathcal{A}^{-1} \mathcal{F}\right)_{4}$ becomes

$$
\begin{equation*}
\hat{\beta}=\tau \frac{a_{41}}{2} F\left(t_{s_{1}}\right)+\tau \frac{a_{42}}{2} F\left(t_{s_{2}}\right)+\tau \frac{a_{43}}{2} F\left(t_{s_{3}}\right)+\tau \frac{a_{44}}{2} F\left(t_{s_{4}}\right) . \tag{5.10}
\end{equation*}
$$

Then, using (5.2), the ECEM4 becomes
(5.11) $y_{m+1}=y_{m}+\alpha F\left(t_{m}, y_{m}\right)$

$$
\begin{aligned}
& +\beta f\left(t_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right) f\left(t_{m}, y_{m}\right)\right) \\
& +\quad \gamma f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2} f\left(t_{m}, y_{m}\right)\right) \\
& +\quad \delta f\left(t_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right) f\left(t_{m}, y_{m}\right)\right) \\
& +\quad \eta f\left(t_{m+1}, y_{m}+\tau f\left(t_{m}, y_{m}\right)\right)
\end{aligned}
$$

where the weights are

$$
\begin{equation*}
\alpha=\tau-(\beta+\gamma+\delta+\eta), \beta=\frac{a_{41}}{2} \tau, \gamma=\frac{a_{42}}{2} \tau, \delta=\frac{a_{43}}{2} \tau, \eta=\frac{a_{44}}{2} \tau \tag{5.12}
\end{equation*}
$$

Finally, combining all above these, we have the fourth-order ECEM4 read as: iterate the followings;

1. Calculate the weights $\alpha, \beta, \gamma, \delta$ and $\gamma$ in (5.12) using (5.7) and (5.9).
2. Calculate $K_{i}(i=1,2,3,4,5)$

$$
\begin{aligned}
K_{1} & =f\left(t_{m}, y_{m}\right) \\
K_{2} & =f\left(t_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right) K_{1}\right) \\
K_{3} & =f\left(t_{m}+\frac{\tau}{2}, y_{m}+\frac{\tau}{2}\left(1-\frac{1}{\sqrt{2}}\right) K_{1}\right) \\
K_{4} & =f\left(t_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right), y_{m}+\frac{\tau}{2}\left(1+\frac{1}{\sqrt{2}}\right) K_{1}\right) \\
K_{5} & =f\left(t_{m+1}, y_{m}+\tau K_{1}\right)
\end{aligned}
$$

3. $y_{m+1}=y_{m}+\alpha K_{1}+\beta K_{2}+\gamma K_{3}+\delta K_{4}+\eta K_{5}$.

Definitely, if $\operatorname{det}(A)=0$ where the matrix $A$ in (5.4) is the principal submatrix of $\mathcal{A}$ in (5.1) and $\operatorname{det}\left(D-C A^{-1} B\right)=0$ in (5.4-5.5), then ECEM4 should be broken down. In this case, one may not get proper weights $\alpha, \beta, \gamma, \delta$ and $\eta$ in ECEM4.

Theorem 5.2. The algorithm will break down if the $2 \times 2$ submatrix $A$ in (5.4) of $\mathcal{A}$ and the matrix $D-C A^{-1} B$ are singular.

## 6. Comparison with Runge-Kutta Methods

The Runge-Kutta method is one of the most popular one-step multistage method which uses intermediate steps to obtain a higher convergence but then discard all previous informations when it moves to the next step. In terms of using the intermediate steps before taking a next step, ECEM can be regarded as one-step multistage method. In order to compare ECEM with RK method, we first briefly introduce the well-known RK method up to order 4 and then will make comparisons. The typical explicit p-stage Runge-Kutta method ( $\mathrm{RK} p$ for $p=2,3,4$ ) is given by

$$
\begin{equation*}
y_{m+1}=y_{m}+\tau \sum_{j=1}^{p} b_{j} \hat{K}_{j}, \quad \text { for } p=2,3,4 \tag{6.1}
\end{equation*}
$$

where $\hat{K}_{1}=f\left(t_{m}, y_{m}\right)$ and

$$
\begin{equation*}
\hat{K}_{j}=f\left(t_{m}+\tau c_{j}, y_{m}+\tau\left(\hat{a}_{j, 1} \hat{K}_{1}+\cdots+\hat{a}_{j, j-1} \hat{K}_{j-1}\right)\right), \text { for } j=2, \cdots, s \tag{6.2}
\end{equation*}
$$

with $b_{j}(1 \leq j \leq p), c_{j}(2 \leq j \leq p)$, and $\hat{a}_{j i}(1 \leq i \leq j-1)$ real numbers. In order to specify a particular $p$-stage Runge-Kutta method, we have to provide the coefficients $a_{j i}$ for $1 \leq i<j \leq p$, weights $b_{j}$ for $j=1,2,3,4$ and nodes $c_{j}$ for $j=2,3,4$. It is known that the explicit Runge-Kutta method is consistent if $\sum_{i=1}^{j-1} \hat{a}_{j i}=c_{j}$ for $j=2,3,4$ (see [2], [5] and [6] for further details).

Based on the formula (2.11), $p$-th order ECEM also is summarized as

$$
\begin{equation*}
y_{m+1}=y_{m}+\sum_{j=0}^{p} a_{j} K_{j}, \quad \text { for } p=2,3,4 \tag{6.3}
\end{equation*}
$$

where
(6.4)
$K_{j}=f\left(t_{s_{j}}, y\left(t_{s_{j}}\right)\right) \quad$ with $t_{s_{j}}=t_{m}+\frac{1+s_{j}}{2} \tau$ and $s_{j}=-\cos \frac{j \pi}{p} \quad$ for $0 \leq j \leq p$.
The $K_{j}$ can be rewritten as $K_{0}=f\left(t_{m}, y_{m}\right)$ which is

$$
\begin{equation*}
K_{j}=f\left(t_{m}+\tau \frac{1+s_{j}}{2}, y_{m}+\tau \frac{1+s_{j}}{2} K_{0}\right) \quad \text { for } 1 \leq j \leq p \tag{6.5}
\end{equation*}
$$

Here the coefficients $a_{j}$ satisfies $\sum_{j=0}^{p} a_{j}=\tau$. Both ECEM $p$ and RK $p$ yield the same $p$-order of convergence. As one can easily see from two formulas (6.1) and (6.3), the biggest difference between ECEM and RK is the function evaluation on each stage. ECEM only uses $K_{0}=f\left(x_{m}, y_{m}\right)$, that is, function evaluation at CGLpoints while RK requires to use all previous intermediate information to generate the next stage function value. In 1- or 2-dimensional problems, keeping all previous stage information $\hat{K}_{1}, \cdots, \hat{K}_{p}$ may not be a big trouble, however, in dealing with higher-dimensional problems, it can cause a trouble. Another difference is that the coefficients of ECEM formula is depending on the function $f$ while the ones of RK are constants. Table 1 summarizes the above.

The most critical comparison can be made in stability area. The following figure 1 compares stability regions for ECEM and RK with order 2,3 and 4. As shown in figure 1, ECEM has much bigger stability region than RK.

## 7. Conclusion

The developed ECEM algorithm in [11] was not stated as the way for explicit RK methods even though the usages of finite difference for $\varphi(s)$ is mentioned for the Jacobian $f_{y}(t, y)$. As this result, one may have some difficulties its application to various kind of time-dependent partial differential equations. Once ECEM algorithms are stated like the class of explicit RK methods, one may ask how much they are different from RK methods. It is shown that ECEM can be explained by a numerical quadrature using CGL nodes in section 2. Still, it is remained how we can determine the numerical weights for chosen internal nodes to get convergence of ECEM algorithms. As we see the discussion in section 6 , one may note that

|  | ECEM $p$ | RK $p$ |
| :---: | :---: | :---: |
| Convergence | order $p$-convergence globally | order $p$-convergence globally |
| $y_{m+1}$ | $y_{m}+\sum_{j=0}^{p} a_{j} K_{j}$ | $y_{m}+\tau \sum_{j=1}^{p} b_{j} K_{j}$ |
| Multi-stage evaluation | $\begin{gathered} K_{0}=f\left(t_{m}, y_{m}\right) \\ K_{j}=f\left(t_{m}+\tau \frac{1+s_{j}}{2}, \tilde{y}_{m}\right) \\ \tilde{y}_{m}=y_{m}+\tau \frac{1+s_{j}}{2} K_{0} \\ 1 \leq j \leq p \end{gathered}$ | $\begin{gathered} K_{1}=f\left(t_{m}, y_{m}\right) \\ \hat{K}_{j}=f\left(t_{m}+\tau c_{j}, \hat{y}_{m}\right) \\ \hat{y}_{m}=y_{m}+\tau \sum_{i=1}^{j-1} \hat{a}_{j, i} \hat{K}_{i} \\ 2 \leq j \leq p \end{gathered}$ |
| $\begin{gathered} K_{j}, \hat{K}_{j} \\ \text { evaluation cost } \end{gathered}$ | $<$ |  |
| on each step <br> storage requirements | ( specially in 3D computations ) |  |
| Weights | $f$-depending weights $a_{j}$ $\sum_{j=0}^{p} a_{j}=\tau$ | constant weights $b_{j}$ $\tau \sum_{j=1}^{p} b_{j}=\tau$ <br> Butcher's table |
| weights evaluation cost | $\gg$ |  |

Table 1: Comparison of ECEM and RK
the critical difference between RK methods and ECEM algorithms rely on whether the employment of previous all function values to get next function evaluations. As a result, the stability properties of RK and ECEM are quite different. In this stability sense, ECEM is even better than explicit RK methods possessing same convergence order. If one may change the platform from a local Euler polygon to a polynomial of degree $q$, then one may have its convergence order $2 q+2$ (see [12]). With such polynomials platform, it is still remained to investigate the concise form of algorithms like RK methods. For a $2 \times 2$ system of initial value problems such that

$$
\left[\begin{array}{l}
\frac{d u}{d t}  \tag{7.1}\\
\frac{d v}{d t}
\end{array}\right]=\left[\begin{array}{ll}
a & b \\
c & d
\end{array}\right]\left[\begin{array}{l}
f(t, u) \\
g(t, v)
\end{array}\right],
$$

one may use ECEM methods easily. However, an application of ECEM will be provided later for a general linear system with $f(t, u, v)$ and $g(t, u, v)$ in (7.1).


Figure 1: Stability regions of ECEM and RK for Dalquist's problem : shaded area is the stable region

## References

[1] J. C. Butcher, On Runge-Kutta processes of high order, Journal of the Australian Mathematical Society, 4(1964), 179-194.
[2] J. C. Butcher Numerical Methods for Ordinary Differential Equations, New York: John Wiley and Sons(2003).
[3] G. Dahlquist, A special stability problem for linear multistep methods, BIT 3(1963), 27-43.
[4] P. J. Davis, P. Rabinowitz Methods of Numerical Integration, Academic Press, New York(1975).
[5] E. Hairer, S.P Norsett, G. Wanner, Solving ordinary differential equations I, nonstiff problems, Springer, Berlin(1993).
[6] E. Hairer, G. Wanner, Solving ordinary differential equations. II Stiff and DifferentialAlgebraic Problems, Springer Series in Computational Mathematics, Springer(1996).
[7] A. C. Hindmarsh, ODEPACK, A systematized collection of ODE solvers, in: R. S. Stepleman et al., eds, Scientific Computing(North-Holland, Amsterdam, 1983), 55-64
[8] F. Iavernaro, F. Mazzia, Solving ordinary differential equations by generalized Adams methods: properties and implementation techniques, Appl. Numer. Math., 28(1998), 107-126.
[9] F. Iavernaro, F. Mazzia, Block-boundary value methods for the solution of ordinary differential equations, SIAM J. Sci. Comput., 21(1999), 323-339.
[10] L. G. Ixaru, Exponentially fitted variable two-step BDF algorithm for first order ODEs, Comput. Phys. Comm., 150(2003), 116-128.
[11] P. Kim, X. Piao and S. Kim, An error corrected Euler method for solving stiff problems based on Chebyshev collocation, SIAM J. Numer. Anal., 49(2011), 2211-2230.
[12] S. D. Kim, X. Piao, D. H. Kim, P. Kim Convergence on error correction methods for solving initial value problems, J. Comput. Appl. Math., 236(2012), 4448-4461.
[13] J. Kwon, S. D. Kim, X. Piao, P. Kim, A Chebyshev collocation method for a stiff initial value problem and its stability, Kyungpook Math. J., 51(2011), 435-456.
[14] W. Liniger, R. A. Willoughby, Efficient integration methods for stiff systems of ordinary differential equations, SIAM J. Numer. Anal., 7(1970), 47-66.
[15] A. Prothero, A. Robinson, On the stability and accuracy of one-step methods for solving stiff systems of ordinary differential equations, Math. Comput., 28(1974), 145-162.
[16] H. Ramos, R. Garcia-Rubio, Analysis ofa Chebyshev-based backward differentiation formulae and relation with Runge-Kutta collocation methods, Int. J. Comput. Math., 88(2011), 555-577.
[17] H. Ramos, J. Vigo-Aguiar, A fourth-order Runge-Kutta method based on BDF-type Chebyshev approximations, J. Comp. Appl. Numer., 204(2007), 124-136.
[18] H. Ramos, A non-standard explicit integration scheme for initial-value problems, Appl. Math. Comp., 189(2007), 710-718.
[19] H. H. Rosenbrock, Some general implicit processes for the numerical solution of differential equations, Computer J., 5(1962/63), 329-330.
[20] L. F. Shampine, M.W. Reichelt, The MATLAB ODE suite, SIAM J. Sci. Comput., 18(1997), 1-22.
[21] M.M. Stabrowski, An efficient algorithm for solving stiff ordinary differential equations, Simul. Pract. Theory 5(1997), 333-344.
[22] J. G. Verwer, E. J. Spee, J. G. Blom, W.H. Hundsdorfer, A second-order Rosenbrock method applied to photochemical dispersion problems, SIAM J. Sci. Comput., 20(1999), 1456-1480.
[23] J. Vigo-Aguiar, H. Ramos, A family of A-stable Runge-Kutta collocation methods of higher order for initial-value problems, IMA J. Numer. Anal., 27(2007), 798-817.
[24] X. Y. Wu, J. L. Xia, Two low accuracy methods for stiff systems, Appl. Math. Comput., 123(2001), 141-153.


[^0]:    * Corresponding Author.

    Received November 5, 2013; accepted December 4, 2013.
    2010 Mathematics Subject Classification: 65M55, 65N30, 49J20, 49K20.
    Key words and phrases: Runge-Kutta method, Butcher's table, Error Corrected Euler Method.
    This research was supported by Kyungpook National University Research Fund, 2012.

