Generalized Chebyshev Collocation Method

Junghan Kim, Wonkyu Chung, Sunyoung Bu, Philsu Kim

Abstract—In this paper, we introduce a generalized Chebyshev collocation method (GCCM) based on the generalized Chebyshev polynomials for solving stiff systems. For employing a technique of the embedded Runge-Kutta method used in explicit schemes, the property of the generalized Chebyshev polynomials is used, in which the nodes for the higher degree polynomial are overlapped with those for the lower degree polynomial. The constructed algorithm controls both the error and the time step size simultaneously and further the errors at each integration step are embedded in the algorithm itself, which provides the efficiency of the computational cost. For the assessment of the effectiveness, numerical results obtained by the proposed method and the Radau IIA are presented and compared.

Keywords—Generalized Chebyshev Collocation method, Generalized Chebyshev Polynomial, Initial value problem.

I. INTRODUCTION

The embedded Runge-Kutta (ERK) method is a popular strategy for solving initial value problems described by

\[
\frac{d\phi}{dt} = f(t, \phi(t)), \quad t \in [t_0, t_f] ; \quad \phi(t_0) = \phi_0.
\]

However, it is well known that ERK doesn’t work well when the stiffness of system (1) is high (see [1]). The aim of this paper is to construct an algorithm of ERK type so called a generalized Chebyshev collocation method (GCCM) for solving the stiff system. The developed GCCM is based on the generalized Chebyshev polynomials introduced by [2], [3] and extensively used for approximating functions and integrals([7], [8]). The main theory of the generalized Chebyshev polynomial approximation is to use node points duplicately for Chebyshev polynomials of consequent degrees. The GCCM of ERK type is constructed from this idea together with the Chebyshev collocation method (CCM) developed by [5], [6], [9], [11]. In particular, it uses two different orders of CCM together with two Chebyshev polynomials of different degrees \(p\) and \(q\) \((q > p)\) and hence we would like to denote GCCM\(p(q)\) simply. Practically, the solution at each integration scheme is calculated with the CCM of order 4 denoted by CCM4 using the Chebyshev interpolating polynomial on the Chebyshev-Gauss-Lobatto (CGL) nodes given by

\[
\{\tau_k\} = \{s_j\} \cup \{\cos\left(\frac{3}{8}\pi\right), \cos\left(\frac{5}{8}\pi\right)\}, \quad k = 0, 1, \cdots, 6.
\]

Assume that \(\phi_m\) is known approximation for \(\phi(t_m)\) and \(e_m\) is known approximation of \(E_m = \phi(t_m) - \phi_0\) at time \(t_m\). Our aim is to find approximations at time \(t_{m+1}, \phi_{m+1}\) and \(e_{m+1}\). To find these values, we have to solve the IVP given by

\[
\begin{align*}
\phi'(t) & = f(t, \phi(t)), \quad t \in [t_m, t_{m+1}], \\
\phi(t_m) & = \phi_m + E_m.
\end{align*}
\]

First, to use Chebyshev Collocation Method, let \(\tilde{\phi}(s) := \tilde{\phi}(t(s))\) be a function on linear transformed domain from \([t_m, t_{m+1}]\) to \([-1, 1]\). Then, the IVP is changed to

\[
\begin{align*}
\tilde{\phi}'(s) & = \frac{d}{dt}f(t(s), \tilde{\phi}(s)), \quad s \in [-1, 1], \\
\tilde{\phi}(1) & = \phi_m + E_m.
\end{align*}
\]

We now try to introduce generalized Chebyshev interpolating polynomials for approximating a given function \(g\). From now on, we define

\[
t_k = \begin{cases} s_k, & \text{if } N = 4, \\ \tau_k, & \text{if } N = 6, \end{cases}
\]

where \(N\) means degree of polynomial. Let \(p_N(s)\) be the Chebyshev interpolating polynomial of degree \(N\) satisfying \(p_N(t_k) = g(t_k)\), \(0 \leq k \leq N\). Then, \(p_N(s)\) can be expressed by the Lagrangian interpolation as follows:

\[
p_N(s) := \sum_{k=0}^{N} q(t_k)l_k(s), \quad l_k(s) := \frac{q_N(s)}{(s-t_k)q_N(t_k)}
\]

where \(q_N(s) := \frac{d}{ds}q_N(s)\) and \(q_N(s)\) is a polynomial of degree \(N + 1\) defined by

\[
q_N(s) := \prod_{j=0}^{N}(s-t_j).
\]

Using the points \(t_j\) and basis \(l_k\), define a matrix \(L\) by \(L = (L_{jk})\), where \(L_{jk} = l_k(t_j)\). Then, for fourth-degree,

\[
L = \begin{pmatrix}
\frac{1}{\sqrt{2}} & \sqrt{2} & -\frac{1}{\sqrt{2}} & 1 - \frac{1}{\sqrt{2}} \\
-\sqrt{2} & 0 & \sqrt{2} & -\frac{1}{\sqrt{2}} \\
\frac{1}{2} & 1 & \frac{1}{2} & \frac{1}{2} \\
2(\sqrt{2} - 2) & 2 & -2(2 + \sqrt{2}) & \frac{1}{2}
\end{pmatrix}
\]

and for sixth-degree, one can get \(L\) similarly. Using the Chebyshev interpolating polynomial \(p_N(s)\), let

\[
\tilde{\phi}(s) = p_N(s) + p_N(s)
\]
to approximate the solution $\tilde{\phi}$, where $\rho_N$ is the local truncation error. After differentiating both sides of (3) and collocating the resulted equation at the points $t_j$, we can obtain a discrete system of equations as follows:

$$\frac{h}{2} f(t_{m_j}, \tilde{\phi}(t_j)) - \sum_{k=0}^{N} \tilde{\phi}(t_k) \hat{h}_k(t_j) = \rho_N(t_j), \quad 1 \leq j \leq N,$$

where $t_{m_j} = t_m + \frac{h}{2}(1 + t_j)$. By truncating the vector $\rho_N(t_j)$ in (4) and perturbing the initial value of (2) with $\phi_m + \epsilon_m$, the system (4) leads to

$$\sum_{k=1}^{N} \alpha_k \hat{h}_k(t_j) - \frac{h}{2} f(t_{m_j}, \alpha_j) = - (\phi_m + \epsilon_m) \hat{h}_0(t_j)$$

(5)

for $1 \leq j \leq N$, where $\alpha_j$ means an approximation for the exact solution $\tilde{\phi}(t_j)$, $j = 1, \cdots, N$. By introducing $\alpha$ defined by

$$\alpha := [\alpha_1, \cdots, \alpha_N]^T,$$

the discrete system (5) can be simplified as follows.

$$L \alpha - \frac{h}{2} f(t, \alpha) = -L_0(\phi_m + \epsilon_m).$$

(6)

Then, the nonlinear system (6) made from GCCM is solved with a Newton iteration method. By solving system (6), we can summarize the developed algorithm GCCM4(6) by

$$\begin{cases}
\phi_{m+1} = F(\phi_m, \epsilon_m), \\
\epsilon_{m+1} = G(\phi_m, \phi_{m+1}, \epsilon_m),
\end{cases}$$

where the above two equations are obtained by the CCM4 and CCM6, respectively. In particular, the error formula of $\epsilon_{m+1}$ is obtained with the observation that there is a Newton’s formula between Chebyshev interpolating polynomial $p_0(s)$ and $p_4(s)$ given by

$$p_0(s) := p_4(s) + \sum_{k=1}^{2} \epsilon_k(T_{4-k}(s) - T_{4+k}(s)),$$

where coefficients $\{\epsilon_k\}$ are determined. Indeed, $\epsilon = p_6 - p_4 = \sum_{k=1}^{2} \epsilon_k(T_{4-k}(s) - T_{4+k}(s))$.

For the time step control, we use the formula for a given tolerance $\text{tol}$ and atol

$$h_{new} := \alpha \times \frac{h_{old}}{E},$$

where $\alpha$ is the safety factor and $E := \text{atol} + \text{rtol} \times \max(\phi_m, \phi_{m+1})$. The proposed algorithm controls both the errors and the time step size simultaneously and it must be noted that the errors at each integration step are embedded in the algorithm itself. This is a novelty of GCCM and reduce an accumulation of the estimated errors. In other words, by giving the usage of estimated error, we can improve the capability of the existing method, while existing method uses the estimated error only for the step-size selection.

II. NUMERICAL RESULT

To assess the improvement and effectiveness of the proposed scheme, Lambert’s problem [4] and Prothero-Robinson equation [10] are solved. For a comparison of the numerical result, a popular existing algorithm Radau5 [1] is used.

Example 1. Consider a problem from Lambert given by

$$y'_1(t) = -2y_1(t) + y_2(t) + 2 \sin(t),$$

$$y'_2(t) = 998y_1(t) - 999y_2(t) + 999(\cos(t) - \sin(t)),$$

(7)

with initial conditions $y_1(0) = 2$, $y_2(0) = 3$. The analytic solution is given by

$$y_1(t) = 2e^{-t} + \sin(t), \quad y_2(t) = 2e^{-t} + \cos(t).$$

(8)

Note that the stiffness is high. We have numerically solved the problem on the interval [0,10] and use the numerical solutions as the sum of the approximate solution $\phi_m$ and the estimated error $\epsilon_m$ to give more accurate results. As a measure of the effectiveness, we calculate both the required number of function evaluations (nfeval) and the computational time (cputime) to solve the problem. We calculate the $L_\infty$ norm for the absolute error in log-scale at the final time for each problem. The problem is solved by varying the relative tolerance from $1.0 \times 10^{-5}$ to $1.0 \times 10^{-8}$ and absolute tolerance from $1.0 \times 10^{-7}$ to $1.0 \times 10^{-10}$. The numerical results are presented in Table I,II and displayed on Fig.1 (a) and (b), where $y$ and $x$ axes represent the absolute errors and either cputime or nfeval, respectively. All the marked points in figure from left to right are corresponding to the given tolerances from large to small, respectively. One can see that the proposed scheme is more efficient than the other scheme. For example, let us consider the point in the right corner. In Fig.1 (b), the point for Radau5 evaluate 2108 number of function with the error $7.0 \times 10^{-10}$, however, GCCM4(6) needs 505 nfeval only with the error $1.0 \times 10^{-12}$.

<table>
<thead>
<tr>
<th>Ex1</th>
<th>GCCM46</th>
<th>Radau5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rtol</td>
<td>Err</td>
<td>CPUtime</td>
</tr>
<tr>
<td>$10^{-1}$</td>
<td>2.31 $\times 10^{-9}$</td>
<td>0.0266</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1.77 $\times 10^{-10}$</td>
<td>0.0378</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1.2 $\times 10^{-11}$</td>
<td>0.0548</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1.0 $\times 10^{-12}$</td>
<td>0.0850</td>
</tr>
</tbody>
</table>

Table I: Comparison of errors spent cputime

<table>
<thead>
<tr>
<th>Ex1</th>
<th>GCCM46</th>
<th>Radau5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rtol</td>
<td>nfeval</td>
<td>CPUtime</td>
</tr>
<tr>
<td>$10^{-1}$</td>
<td>162</td>
<td>1.41 $\times 10^{-7}$</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>235</td>
<td>2.39 $\times 10^{-8}$</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>335</td>
<td>5.1 $\times 10^{-9}$</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>505</td>
<td>7.0 $\times 10^{-10}$</td>
</tr>
</tbody>
</table>

Table II: Comparison of errors versus number of function-evaluations

Example 2. We test the Prothero-Robinson equation, which is a particular case of the family of scalar equation proposed by Prothero and Robinson and constitutes a stiff problem, $\phi'(t) = \nu(\phi(t) - g(t)) + g'(t)$, $t \in (0,10]$; $\phi(0) = 1$.
where the eigenvalue $\nu$ is $\nu = -10^6$ and $g(t) = \cos(t)$. The exact solution is given by $\phi(t) = \sin(t)$. The problem is solved by varying the relative tolerance from $1.0 \times 10^{-7}$ to $1.0 \times 10^{-10}$ and absolute tolerance from $1.0 \times 10^{-9}$ to $1.0 \times 10^{-12}$. The numerical results are displayed in Fig. 2 and Table III, from which one can see that the presented method are better than Radau5.

### TABLE III

<table>
<thead>
<tr>
<th>Ex2</th>
<th>GCCM46</th>
<th>Radau5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rtol</td>
<td>Err</td>
<td>CPUtime</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>$3.59 \times 10^{-11}$</td>
<td>0.0031</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>$3.58 \times 10^{-12}$</td>
<td>0.0036</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>$2.1 \times 10^{-13}$</td>
<td>0.00803</td>
</tr>
<tr>
<td>$10^{-10}$</td>
<td>$5.0 \times 10^{-14}$</td>
<td>0.0190</td>
</tr>
</tbody>
</table>

### III. CONCLUSION

In summary, a generalized Chebyshev collocation method of ERK-type based on generalized Chebyshev approximations is newly introduced for solving stiff initial value problem.

Using the CCM of two different convergence orders 4 and 6, the solution and the error are calculated efficiently and also we suggest a methodology that contains itself the estimated error at each integration step. One can see that the suggested algorithm is more efficient than Radau5 throughout several numerical tests.

### ACKNOWLEDGMENT

This work was supported by basic science research program through the National Research Foundation of Korea(NRF) funded by the ministry of education, science and technology (grand number 2011-0009825). The third author Dr. Bu was supported by the basic science research program through the

---

**Fig. 1.** Comparison of errors spent cputime (a) and versus number of function-evaluations (b)

**Fig. 2.** Comparison of errors spent cputime (a) and versus number of function-evaluations (b)

**TABLE IV**

<table>
<thead>
<tr>
<th>Ex2</th>
<th>GCCM46</th>
<th>Radau5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rtol</td>
<td>Err</td>
<td>nfeval</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>$3.59 \times 10^{-11}$</td>
<td>22</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>$3.58 \times 10^{-12}$</td>
<td>28</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>$2.1 \times 10^{-13}$</td>
<td>65</td>
</tr>
<tr>
<td>$10^{-10}$</td>
<td>$5.0 \times 10^{-14}$</td>
<td>158</td>
</tr>
</tbody>
</table>
National Research Foundation of Korea(NRF) funded by the ministry of education, science and technology (grand number NRF-2013R1A1A2062783). Also, the fourth author Prof. Kim was supported by the basic science research program through the National Research Foundation of Korea(NRF) funded by the ministry of education, science and technology (grand number 2011-0029013).

REFERENCES


