Turkish Journal of Mathematics

Volume 46 | Number 8 Article 14

1-1-2022

A matrix-collocation method for solutions of singularly perturbed differential equations via Euler polynomials

DENİZ ELMACI

ŞUAYİP YÜZBAŞI

NURCAN BAYKUŞ SAVAŞANERİL

Turkish Journal of Mathematics

http://journals.tubitak.gov.tr/math/

Research Article

Turk J Math
(2022) 46: 3260 – 3275
© TÜBİTAK
doi:10.55730/1300-0098.3332

A matrix-collocation method for solutions of singularly perturbed differential equations via Euler polynomials

Deniz ELMACI[1,*], **Şuayip YÜZBAŞI**[2], **Nurcan BAYKUŞ SAVAŞANERİL**[3]
[1]Dokuz Eylül University, Bergama Vocational School, İzmir, Turkey
[2]Akdeniz University, Faculty of Science, Department of Mathematics, Antalya, Turkey
[3]Dokuz Eylül University, İzmir Vocational School, İzmir, Turkey

Received: 17.09.2022 • **Accepted/Published Online:** 20.09.2022 • **Final Version:** 09.11.2022

Abstract: In this paper, a matrix-collocation method which uses the Euler polynomials is introduced to find the approximate solutions of singularly perturbed two-point boundary-value problems (BVPs). A system of algebraic equations is obtained by converting the boundary value problem with the aid of the collocation points. After this algebraic system, the coefficients of the approximate solution are determined. This error analysis includes two theorems which consist of an upper bound of errors and an error estimation technique. The present method and error analysis are applied to three numerical examples of singularly perturbed two-point BVPs. Numerical examples and comparisons with other methods are given. These applications and comparisons show that the method gives effective results.

Key words: Euler polynomials, singular perturbed differential equations, boundary value problems, error analysis, collocation method, matrix method

1. Introduction

Singularly perturbed differential equations are often utilized for modeling a physical phenomena such as aerodynamics, elasticity, fluid mechanics, magneto-hydrodynamics, fluid dynamics, plasma dynamics, oceanography, etc. These problems are containing a small parameter ϵ. While the solution of the problem changes slowly in some parts of the domain, it changes rapidly in other parts. Additionally, the solution function of such problems changes rapidly near the boundary points. Expansion methods asymptotically are available for solving these type problems. However, it is difficult to find the appropriate asymptotic expansions in the inner and outer regions. Therefore, for solving the singularly perturbed boundary value problems more simpler and efficient computational techniques are required.

In recent years, the papers including various methods such as the B-spline collocation method [9], the seventh order numerical method [1], the finite difference methods [8], the B-spline method [10], the finite difference method with second order spline [12], a other numerical algorithm of the finite difference method [6], the Bessel collocation method [15] and the Laguerre method [16] have been published for solving the mentioned problems.

In this article, the singularly perturbed differential equation is considered in the form

$$\epsilon y''(t) + a(t)y'(t) + b(t)y(t) = g(t), \quad t \in [0,1], \tag{1.1}$$

*Correspondence: deniz.elmaci@deu.edu.tr
2010 *AMS Mathematics Subject Classification:* 11B68, 34E15, 34K10, 65G99, 65L60, 40C05.

with conditions

$$y(0)=\xi_0, \quad y(1)=\xi_1 \ , \tag{1.2}$$

where $0<\epsilon\leq 1, \xi_0$, and ξ_1 show the known constants, $y(t)$ shows the unknown function of the problem, $a(t), b(t)$ and $g(t)$ are smooth functions sufficiently in [0,1].

To find an approximate solution of Eq. (1.1), we will seek the approximate solution in the form of the truncated Euler series

$$y(t)\cong y_N(t)=\sum_{n=0}^{N} a_n \mathrm{E}_n(t), \qquad 0\leqslant t\leqslant 1 \ . \tag{1.3}$$

In Section 2, the Euler polynomials are introduced. Section 3 describes the matrix relations of the unknown function and its derivatives in the orginal problem. The Euler matrix-collocation method is presented in Section 4. In Section 5, a theorem regarding error estimates is derived by using the residual correction and also another theorem which contains upper bound of the errors of the suggested technique is presented. As for Section 6, the applications are made and the obtained results are discussed with the aid of some examples.

2. The Euler polynomials

Most polynomials play an important role in solving many problems in applied mathematics. One of them, the Euler polynomials, is successful in numerical methods [2, 3]. The purpose of this work is to find the solutions of the singularly perturbed differential equations by using the Euler polynomials. In this section, some significant results of these polynomials are shown.

The Euler polynomials $E_n(t)$ which are described as

$$\frac{2e^{xt}}{e^t+1}=\sum_{n=0}^{\infty} E_n(x)\frac{t^n}{n!}, \quad |t|<\pi \ . \tag{2.1}$$

The Euler polynomials can be obtained with the recursive calculation by utilizing the following formula [24];

$$E_n(t)+\sum_{k=0}^{n}\left(\begin{array}{c} n \\ k \end{array}\right) E_k(t)=2t^n, \quad n=1,2,\ldots \tag{2.2}$$

The recurrence relations between the Euler polynomials and their derivatives is given by

$$E'_m(t)=mE_{m-1}(t), \quad m\geqslant 1. \tag{2.3}$$

By means of Eq. (2.1), Eq. (2.2) or Eq. (2.3), some of the Euler polynomials can be computed as follows:

$$E_0(t)=1, \quad E_1(t)=t-\frac{1}{2}, \quad E_2(t)=t^2-t, \quad E_3(t)=t^3-\frac{3}{2}t^2+\frac{1}{4},$$

$$E_4(t)=t^4-2t^3+t, \quad E_5(t)=t^5-\frac{5}{2}t^4+\frac{5}{2}t^2-\frac{1}{2},$$

$$E_6(t)=t^6-3t^5+5t^3-3t, \quad E_7(t)=t^7-\frac{7}{2}t^6+\frac{35}{4}t^4-\frac{21}{2}t^2+\frac{17}{8} \ .$$

3. Matrix relations

In this section, the solution form and its derivatives will be expressed in matrix forms by means of the Euler polynomials. Now let us see the relevant lemma and proofs. For this purpose, let us begin by writing the solution form (1.3) in the following matrix form

$$y(t) \cong y_N(t) = \mathbf{E}(t)\,\mathbf{A}\ , \tag{3.1}$$

where

$$\mathbf{E}(t) = [E_0(t)\ \ E_1(t)\ \ \cdots\ \ E_N(t)]\,, \quad \mathbf{A} = [a_0\ a_1\ \cdots\ a_N]^T$$

.

Lemma 3.1 *The vector* $\boldsymbol{E}(t)$ *can be written as*

$$\mathbf{E}(t) = \mathbf{T}(t)\,\mathbf{D}\ , \tag{3.2}$$

where

$$\mathbf{T}(t) = \left[1\ t\ \cdots\ t^N\right],$$

$$\left(\mathbf{D}^{-1}\right) = \begin{bmatrix} 1 & \frac{1}{2}\binom{1}{0} & \frac{1}{2}\binom{2}{0} & \cdots & \frac{1}{2}\binom{N}{0} \\ 0 & 1 & \frac{1}{2}\binom{2}{1} & \cdots & \frac{1}{2}\binom{N}{1} \\ 0 & 0 & 1 & \cdots & \frac{1}{2}\binom{N}{2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}_{(N+1)\times(N+1)}$$

Proof From the right side, we multiply the vector $\mathbf{T}(t)$ with the matrix $\mathbf{D}$. Hence we have the vector $\mathbf{E}(t) = \mathbf{T}(t)\,\mathbf{D}$. □

Lemma 3.2 *The derivatives of the solution form Eq. (3.2) can be expressed by the matrix relations*

$$y'(t) \cong y_N'(t)\ = \mathbf{E}'(t)\,\mathbf{A} = \mathbf{T}'(t)\,\mathbf{D} = \mathbf{T}(t)\,\mathbf{BDA} \tag{3.3}$$

$$y^{(k)}(t) \cong {y_N}^{(k)}(t)\ = \mathbf{E}^{(k)}(t)\,\mathbf{A} = \mathbf{T}(t)\,\mathbf{B}^k\mathbf{DA},\ k = 0,1,2,\ldots\,, \tag{3.4}$$

where

$$\mathbf{B} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & N \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{B}^0 = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

.

Proof It must be cleary expressed the matrix representations $\mathbf{E}'(t)$ in Eq. (3.3). For this reason, it will be used the derivative relation of $\mathbf{E}_n(t)$ in Eq. (2.3). Due to the fact that the derivative relation is $E'_m(t) = mE_{m-1}(t)$, the matrix representation for the derivatives can be written as

$$\mathbf{T}'(t) = \mathbf{T}(t)\mathbf{B}$$

$$\mathbf{T}''(t) = \mathbf{T}(t)\mathbf{B}^2$$

$$\vdots$$

$$\mathbf{T}^k(t) = \mathbf{T}(t)\mathbf{B}^k$$

. Lastly, by putting these relations in the regarding expressions, the matrix representations Eqs. (3.3) and (3.4) are obtained. □

4. The Euler matrix-collocation method

In this section, the method is described by using matrix relations in the previous section and equally spaced collocation points.

Lemma 4.1 *It is assumed that the problem in Eq. (1.1) has an approximate solution in the form Eq. (1.3). The matrix representation is obtained as follows:*

$$\epsilon\mathbf{T}(t)\mathbf{B}^2\mathbf{DA} + a(t)\mathbf{T}(t)\mathbf{BDA} + b(t)\mathbf{T}(t)\mathbf{DA} = g(t)\ . \tag{4.1}$$

Proof The desired relation is obtained when the obtained matrix representations Eqs. (3.2) and (3.4) are substituted in Eq. (1.1). □

In next theorem, we use the equally spaced collocation points

$$t_i = a + \frac{b-a}{N}i, \quad i = 0, 1, \ldots, N.$$

Theorem 4.2 *We suppose that a polynomial solution of the singular perturbed problem (1.1) is sought in the form of the truncated Euler series (1.3). Then by utilizing the equally spaced collocation points, Equation (1.1) is reduced to the following system of algeraic equations*

$$\mathbf{WA} = \mathbf{G} \quad \Leftrightarrow \quad [\mathbf{W} : \mathbf{G}]\ . \tag{4.2}$$

Proof If the collocation points are written in Eq. (4.1), then we get

$$\varepsilon\mathbf{T}(t_i)\mathbf{B}^2\mathbf{DA} + \mathbf{a}(t_i)\mathbf{T}(t_i)\mathbf{BDA} + \mathbf{b}(t_i)\mathbf{T}(t_i)\mathbf{DA} = g(t_i) \tag{4.3}$$

Briefly Eq. (4.3) can be written as $\mathbf{WA} = \mathbf{G}$. Here

$$\mathbf{W} = \varepsilon\mathbf{T}(t_i)\mathbf{B}^2\mathbf{D} + \mathbf{a}(t_i)\mathbf{T}(t_i)\mathbf{BD} + \mathbf{b}(t_i)\mathbf{T}(t_i)\mathbf{D}$$

$$\varepsilon = \begin{bmatrix} \epsilon & 0 & \cdots & 0 \\ 0 & \epsilon & 0 & \vdots \\ \vdots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & \epsilon \end{bmatrix}, \quad \mathbf{T} = \begin{bmatrix} \mathbf{T}(t_0) \\ \mathbf{T}(t_1) \\ \vdots \\ \mathbf{T}(t_N) \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} g(t_0) \\ g(t_1) \\ \vdots \\ g(t_N) \end{bmatrix}$$

$$\mathbf{a} = \begin{bmatrix} a(t_0) & 0 & \cdots & 0 \\ 0 & a(t_1) & 0 & \vdots \\ \vdots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & a(t_N) \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b(t_0) & 0 & \cdots & 0 \\ 0 & b(t_1) & 0 & \vdots \\ \vdots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & b(t_N) \end{bmatrix}$$

. Hence, this completes the proof of the theorem.

□

Lemma 4.3 *The conditions Eq. (1.2) can be written in the matrix forms based on the Euler polynomials as follows:*

$$\boldsymbol{T}(0)\,\boldsymbol{DA} = \xi_0, \quad \boldsymbol{T}(1)\,\boldsymbol{DA} = \xi_1 \quad or \quad \left[\begin{array}{cc} \mathbf{U}: & \xi \end{array}\right], \tag{4.4}$$

where

$$\mathbf{U} = \begin{bmatrix} u_{00} & u_{01} & \cdots & u_{0N} \\ u_{10} & u_{11} & \cdots & u_{1N} \end{bmatrix}$$

,

$$\xi = \begin{bmatrix} \xi_0 \\ \xi_1 \end{bmatrix}$$

.

Proof By writing the values of the solution form (3.1) and the relation (3.2) at $t = 0$ and $t = 1$ in the conditions of (1.2), the proof is completed.

□

Theorem 4.4 *We assume that a polynomial solution of the problem (1.1)–(1.2) is sought in the form of the truncated Euler series (1.3). Then by utilizing the equally spaced collocation points, the problem (1.1)-(1.2) is reduced to the following system of algeraic equations as* $\widetilde{\mathbf{W}}\ \mathbf{A} = \widetilde{\mathbf{G}}$. *Here,* $[\widetilde{\mathbf{W}}; \widetilde{\mathbf{G}}]$ *is obtained by replacing any two rows* $[\mathbf{W}; \mathbf{G}]$ *by* $[\mathbf{U} : \xi]$.

Proof A new matrix system is created by writing two rows $[\mathbf{U} : \xi]$ obtained for the conditions Eq. (1.2) in Lemma 4.3 instead of any two rows of the system of algebraic equations $[\mathbf{W}; \mathbf{G}]$ in Theorem 4.2. This new system is also represented by $\widetilde{\mathbf{W}}\ \mathbf{A} = \ \widetilde{\mathbf{G}}$. Thus, the proof is finished.

□

Corollary 4.5 *The system of linear algebraic equations in Theorem 4.4 is solved and then the unknown Euler coefficients* a_n *are calculated. Thus, the coefficients are determined by the Euler polynomial approach. Finally, the approximate solution is acquired as follows:*

$$y_N(t) = \mathbf{E}(t)\mathbf{A} = \boldsymbol{T}(t)\boldsymbol{DA}\ . \tag{4.5}$$

5. Error analysis

In this part of the study, error analysis is studied. Initially, a theorem about the upper bound of error is stated then after, a theorem for estimating the error is given.

Theorem 5.1 *Let $y(t)$ be the exact solution and $y_N(t) = \boldsymbol{E}(t)\,\boldsymbol{A}$ be the Euler polynomial solution of the problem Eqs. (1.1)–(1.2) having degree N. Also it is assumed that $y_N^M(t) = \boldsymbol{T}(t)\,\widetilde{\boldsymbol{A}}$ denotes the N-th degree Maclaurin series [14] of $y(t)$. Then, an upper bound of the absolute error of the Euler polynomial solution $y_N(t)$ becomes as below*

$$||y(t) - y_N(t)||_\infty \leq \frac{1}{\Gamma(N+2)}||y^{N+1}(c_t)||_\infty + (||\widetilde{\boldsymbol{A}}||_\infty + ||\boldsymbol{S}||_\infty||\boldsymbol{A}||_\infty)\ , \tag{5.1}$$

where $\boldsymbol{T}(t) = \left[1\ t\ \cdots\ t^N\right]$, $\boldsymbol{E}(t) = [E_0(t)\ \ E_1(t)\ \ \cdots\ E_N(t)], t \in [0,1]$ *and* $\Delta\boldsymbol{A} = \boldsymbol{A} - \widetilde{\boldsymbol{A}}$.

Proof Initially, let us add and subtract the Maclaurin expansion $y_N^M(t)$ with N-th degree in $||y(t) - y_N(t)||_\infty$ and then let us use the triangle inequality. Hence we have

$$\begin{aligned} ||y(t) - y_N(t)||_\infty &= ||y(t) - y_N^M(t) + y_N^M(t) - y_N(t)||_\infty \\ &\leq ||y(t) - y_N^M(t)||_\infty + ||y_N^M(t) - y_N(t)||_\infty. \end{aligned} \tag{5.2}$$

From Lemma 3.1, we can express the Euler polynomial solution $y_N(t) = \mathbf{E}(t)\,\mathbf{A}$ by the matrix form $y_N(t) = \mathbf{T}(t)\,\mathbf{SA}$. From the hypothesis, we know that $y_N^M(t) = \mathbf{T}(t)\,\widetilde{\mathbf{A}}$ shows the N-th degree Maclaurin series of $y(t)$. From these informations, we can write

$$\begin{aligned} ||y_N^M(t) - y_N(t)||_\infty &= ||\mathbf{T}(t)\left(\widetilde{\mathbf{A}} - \mathbf{DA}\right)||_\infty \\ &\leq ||\mathbf{T}(t)||_\infty\left(||\widetilde{\mathbf{A}}||_\infty + ||\mathbf{D}||_\infty||\mathbf{A}||_\infty\right), \quad 0 \leq t \leq 1. \end{aligned} \tag{5.3}$$

Otherwise, it is known that the remainder term of the Maclaurin series $y_N^M(t)$ is $\sum_{n=N+1}^{\infty}\frac{y^n(0)}{\Gamma(n+1)}t^n$. So the following inequality can be written

$$||y(t) - y_N^M(t)||_\infty \leq |\sum_{n=N+1}^{\infty}\frac{y^n(0)}{\Gamma(n+1)}|, \quad 0 \leq t \leq 1\ . \tag{5.4}$$

Then, by using Eqs. (5.2), (5.3) and (5.4), we obtain

$$||y(t) - y_N(t)||_\infty \leq |\sum_{n=N+1}^{\infty}\frac{y^n(0)}{\Gamma(n+1)}| + ||\mathbf{T}(t)||_\infty\left(||\widetilde{\mathbf{A}}||_\infty + ||\mathbf{D}||_\infty||\mathbf{A}||_\infty\right), \quad 0 \leq t \leq 1. \tag{5.5}$$

Since there exists $c_t \in (0,1)$ such that

$$\sum_{n=N+1}^{\infty}\frac{y^n(0)}{\Gamma(n+1)}t^n = \frac{t^{N+1}}{\Gamma(n+2)}y^{(N+1)}(c_t), \quad 0 \leq t \leq 1$$

in the remainder term of Maclaurin series, the inequality Eq. (5.5) becomes as below

$$||y(t)-y_N(t)||_\infty \leq \frac{1}{\Gamma(N+2)}||y^{N+1}(c_t)||_\infty + (||\widetilde{\mathbf{A}}||_\infty + ||\mathbf{D}||_\infty||\mathbf{A}||_\infty), \quad 0 \leq t \leq 1. \tag{5.6}$$

Hence, the proof of the theorem is carried out. □

Theorem 5.2 *Let $y(t)$ be the exact solution and $y_N(t)$ be the Euler polynomial solution of the problem Eqs. (1.1)–(1.2), respectively. To estimate the error $e_N(t)$, we get the error problem as follows:*

$$\epsilon e_N''(t) + a(t)e_N'(t) + b(t)e_N(t) = -R_N(t), \quad t \in [0,1], \tag{5.7}$$

with conditions

$$e_N(0) = 0, \quad e_N(1) = 0\ . \tag{5.8}$$

Here,

$$e_N(t) = y(t) - y_N(t)$$

$$R_N(t) = L[y_N(t)] - g(t)$$

represents the residual function obtained by subsitituting $y_N(t)$ instead of $y(t)$ in the orginal problem Eq. (1.1).

Proof We consider the N-th degree Euler polynomial solution given by

$$y_N(t) = \sum_{n=0}^{N} a_n \, \mathrm{E}_n(t),$$

of the problem Eqs. (1.1)–(1.2). It yields Eq. (1.1) and so it can be written as below

$$R_N(t) = \epsilon y_N''(t) + a(t)y_N'(t) + b(t)y_N(t) - g(t). \tag{5.9}$$

It can be obtained the homogeneous conditions because of the approximate solution Eq. (4.5) satisfies conditions Eq. (1.2)

$$e_N(0) = 0, \quad e_N(1) = 0$$

. So the error problem is constructed as follows:

$$\epsilon e_N''(t) + a(t)e_N'(t) + b(t)e_N(t) = -R_N(t),$$
$$e_N(0) = 0, \quad e_N(1) = 0\ . \tag{5.10}$$

Consequently, the proof is completed. □

Corollary 5.3 *We solve this problem given by (5.10) by using the suggested method in Section 4 and so we have*

$$e_{N,M}(t) = \sum_{k=0}^{M} a_k^* \, \mathrm{E}_{k,N}(t)\ . \tag{5.11}$$

The approximate solution $e_{N,M}(t)$ in Eq. (5.11) becomes an estimation function of the actual error function $e_N(t)$.

6. Numerical illustrations

In this section, to show the correctness and effectiveness of the suggested method some numerical examples are given. A code written in Matlab on PC (which has 8 GB RAM, i7-8550M processor and 1.80 Ghz) has been used.

Besides, CPU times on Matlab for different values of N and ϵ have been computed. Moreover, using a distributed convergence computational order formula in [5], the experimental order of the time complexity (EOT_N) with respect to N can be calculated for the present method as follows:

$$EOT_N = \frac{\log|\tau_{N-1}/\tau_N|}{\log|(N-1)/N|}.$$

Here, τ_N shows an exact time run by the timing module according to N. As such, the timing complexity of the suggested method is given in the numerical examples.

Example 6.1. [8, 15] Let us first solve the singularly perturbed differential equation

$$-\epsilon y''(t) + \frac{1}{t+1}y'(t) + \frac{1}{t+2}y(t) = g(t), \quad t \in [0,1]$$

with the boundry conditions $y(0) = 1 + 2^{\frac{-1}{\epsilon}}$ and $y(1) = 2 + e$.

Here, $a(t) = \frac{1}{t+1}$, $b(t) = \frac{1}{t+2}$, $g(t) = \left(-\epsilon + \frac{1}{t+1} + \frac{1}{t+2}\right)e^t + \frac{1}{t+2}2^{\frac{-1}{\epsilon}}(t+1)^{1+\frac{1}{\epsilon}}$

and the exact solution of the problem is given by $y(t) = e^t + 2^{\frac{-1}{\epsilon}}(t+1)^{1+\frac{1}{\epsilon}}$.

We apply the procedure step by step for various values of N and ϵ in Section 4, we calculate the approximate solutions of problem. We can note some of them as follows:

$$y_5^\epsilon(t) = 0.07394t^5 + 0.35210t^4 + 0.79222t^3 + 1.12502t^2 + 1.31251t + 1.06250, \quad \epsilon = 2^{-2}$$

$$y_8^\epsilon(t) = 0.04864t^8 + 0.12169t^7 + 0.34399t^6 + 0.49430t^5 + 0.53535t^4 + 0.49462t^3 + 0.64063t^2 + 1.03516t + 1.00391 \quad \epsilon = 2^{-3}$$

$$\begin{aligned} y_{10}^\epsilon(t) &= 3.51114t^{10} - 7.89988t^9 + 10.73619t^8 - 7.33270t^7 + 3.69018t^6 - 0.90083t^5 + \\ &\quad 0.24944t^4 + 0.16207t^3 + 0.50237t^2 + 1.00030t + 1.00002 \qquad \epsilon = 2^{-4} \end{aligned}$$

$$\begin{aligned} y_{13}^\epsilon(t) &= 289.75410t^{13} - 1338.37672t^{12} + 2815.69446t^{11} - 3523.29676t^{10} + 2895.33642t^9 - 1632.13061t^8 \\ &\quad + 640.87426t^7 - 174.33856t^6 + 31.97318t^5 - 3.66845t^4 + 0.40140t^3 + 0.49582t^2 + 0.99974t + 1 \qquad \epsilon = 2^{-5}. \end{aligned}$$

The maximum absolute errors, the CPU running time results and the experimental order of timing complexity for various values of (N, M) and ϵ and comparison of the other methods are tabulated for Example 6.1 in Tables 1–4, respectively. And also, the comparisons of the absolute error functions for different values of ϵ are seen in Figures 1–3.

It is seen from Tables 2 and 3 that the CPU running time results with the EOT_N are good. In Table 4, we compare our obtained results with other methods ϵ-Uniformly convergent fitted mesh finite difference

Table 1. Maximum absolute errors for some values of N, M and ϵ in Example 6.1.

	Absolute errors (actual, estimated, improved)					
ϵ	$\lvert e_8^\epsilon\rvert$	$\lvert e_{15}^\epsilon\rvert$	$\lvert e_{8,9}^\epsilon\rvert$	$\lvert e_{15,16}^\epsilon\rvert$	$\lvert E_{8,9}^\epsilon\rvert$	$\lvert E_{15,16}^\epsilon\rvert$
2^{-2}	1.1390e-8	7.3719e-14	1.0911e-8	9.0405e-14	4.8199e-10	3.3751e-14
2^{-3}	7.6066e-6	7.3719e-14	7.6062e-6	2.4682e-14	3.6894e-10	3.6415e-14
2^{-4}	9.6302e-3	1.2517e-9	7.2947e-3	1.2326e-9	2.3508e-3	1.9057e-11
2^{-5}	1.6123e-1	4.8337e-4	6.6685e-2	3.5261e-4	9.5627e-2	1.3192e-4
2^{-6}	5.3014e-1	7.3393e-2	1.1104e-1	2.5260e-2	4.2784e-1	4.8293e-2
2^{-7}	9.3779e-1	4.1172e-1	1.2304e-1	5.5020e-2	8.3517e-1	3.5788e-1

Table 2. CPU running time results in Example 6.1.

	$y_5^\epsilon \quad \epsilon = 2^{-2}$	$y_5^\epsilon \quad \epsilon = 2^{-7}$	$y_{10}^\epsilon \quad \epsilon = 2^{-2}$	$y_{10}^\epsilon \quad \epsilon = 2^{-7}$	$y_{14}^\epsilon \quad \epsilon = 2^{-2}$	$y_{14}^\epsilon \quad \epsilon = 2^{-7}$
CPU time	1.042 s	1.002 s	1.123 s	1.037 s	1.169 s	1.093 s

Table 3. Time complexity in Example 6.1.

N/EOT$_N$	10	11	12	13	14	15
$\epsilon = 2^{-2}$	0.044	0.010	0.075	0.012	0.527	0.569
$\epsilon = 2^{-3}$	0.262	0.020	0.287	0.125	0.073	0.537
$\epsilon = 2^{-4}$	0.009	0.088	0.021	0.162	0.294	0.296
$\epsilon = 2^{-5}$	0.288	0.010	0.170	0.011	0.184	0.324
$\epsilon = 2^{-6}$	0.081	0.020	0.171	0.263	0.218	0.052
$\epsilon = 2^{-7}$	0.293	0.058	0.136	0.135	0.096	0.090

Table 4. Comparison of absolute errors in Example 6.1.

	Proposed method		**SFDM - FFDM**[8]		**Bessel function** [15]	
ϵ	$\lvert e_{15}^\epsilon\rvert$	$\lvert E_{15,16}^\epsilon\rvert$	$n = 64$	$n = 64$	$\lvert e_{13}^\epsilon\rvert$	$\lvert E_{13,14}^\epsilon\rvert$
2^{-2}	7.3719e-14	3.3751e-14	0.78e-4	0.78e-4	2.5091e-14	5.4179e-14
2^{-3}	7.3719e-14	3.6415e-14	0.30e-3	0.30e-3	5.5733e-14	1.3545e-14
2^{-4}	1.2517e-9	1.9057e-11	0.11e-2	0.11e-2	5.9422e-7	3.3387e-8
2^{-5}	4.8337e-4	1.3192e-4	0.40e-2	0.40e-2	4.5215e-3	1.5796e-3
2^{-6}	7.3393e-2	4.8293e-2	0.16e-2	0.16e-2	1.4935e-1	1.0686e-1
2^{-7}	4.1172e-1	3.5788e-1	0.70e-1	0.17e-1	5.3316e-1	4.7055e-1

methods given in [8] and the Bessel function method given in [15]. From these comparison, it is seen that they are the results of other one while our results are similar to the results of the Bessel function method.

Absolute error functions for Example 6.1 are shown in Figures 1–3. It can be clearly inferred that when N is increased, they approach to zero.

Example 6.2. [9] Consider the singularly perturbed differential equation

$$\epsilon y''(t) + y'(t) = 1 + 2t, \quad t \in [0, 1]$$

with the boundary conditions $y(0) = 0$ and $y(1) = 1$.

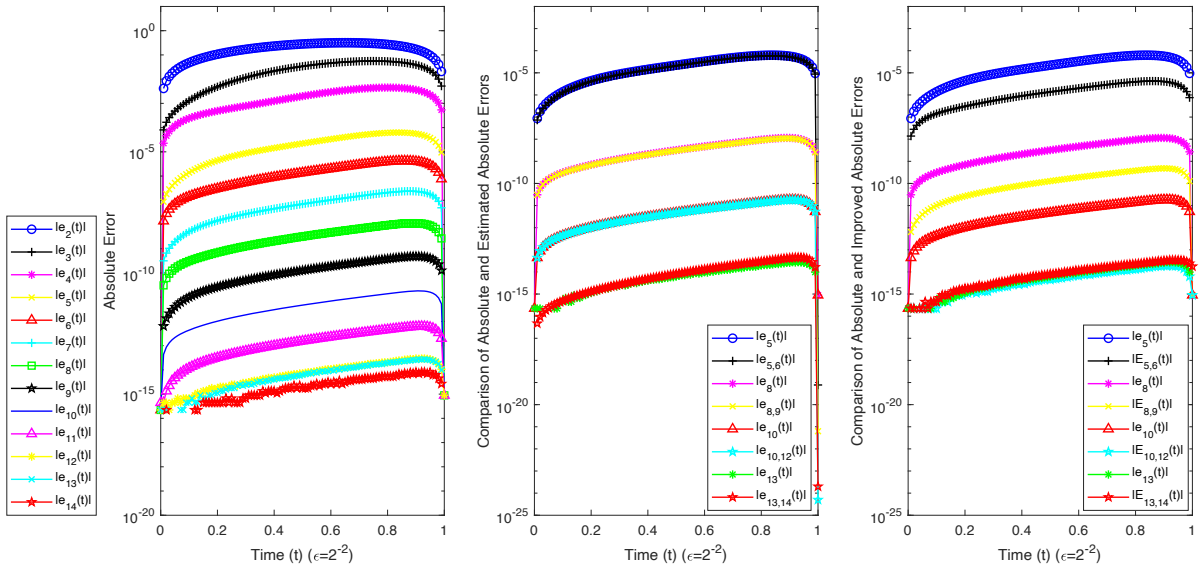

Figure 1. (a) Comparison of the absolute error (actual) functions for $\epsilon = 2^{-2}$ and $N = 2, N = 14$, (b) comparing the estimated absolute error functions for $\epsilon = 2^{-2}$ and some values of (N, M), (c) comparing the improved absolute error functions for $\epsilon = 2^{-2}$ and some values of (N, M).

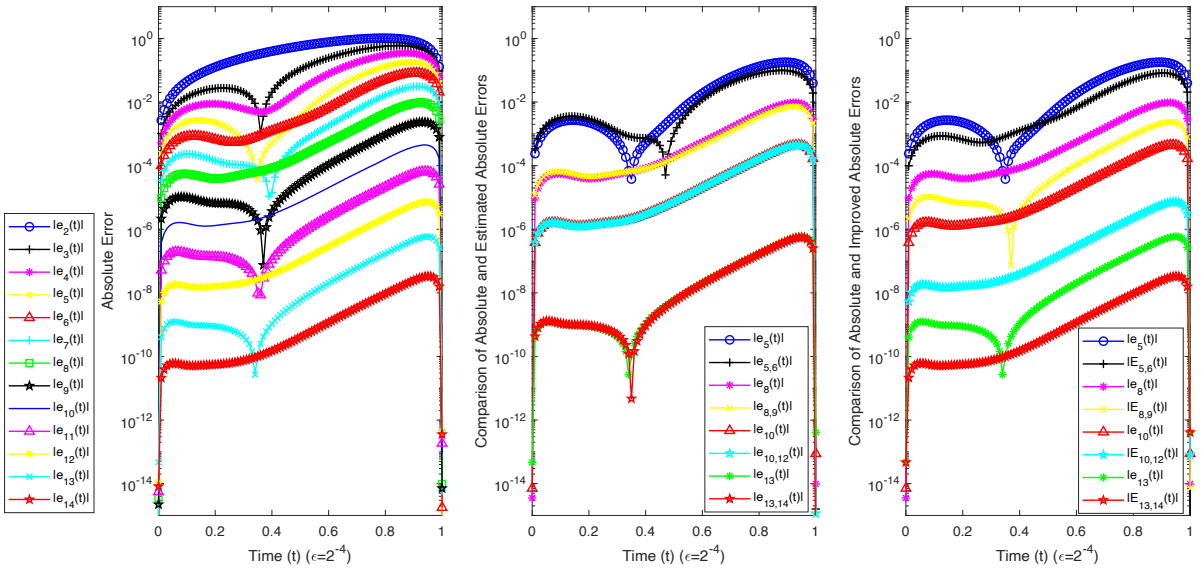

Figure 2. (a) Comparison of the absolute error (actual) functions for $\epsilon = 2^{-4}$ and $N = 2, N = 14$, (b) comparing the estimated absolute error functions for $\epsilon = 2^{-4}$ and some values of (N, M), (c) comparing the improved absolute error functions for $\epsilon = 2^{-4}$ and some values of (N, M).

$a(t) = 0,\ b(t) = 1,\ g(t) = 1 + 2t$

and the problem's exact solution is $y(t) = t(t + 1 - 2\epsilon) + (2\epsilon - 1)\dfrac{1 - e^{\frac{-x}{\epsilon}}}{1 - e^{\frac{-1}{\epsilon}}}$.

The maximum absolute errors, the CPU times and the results of complexity for various values of (N, M)

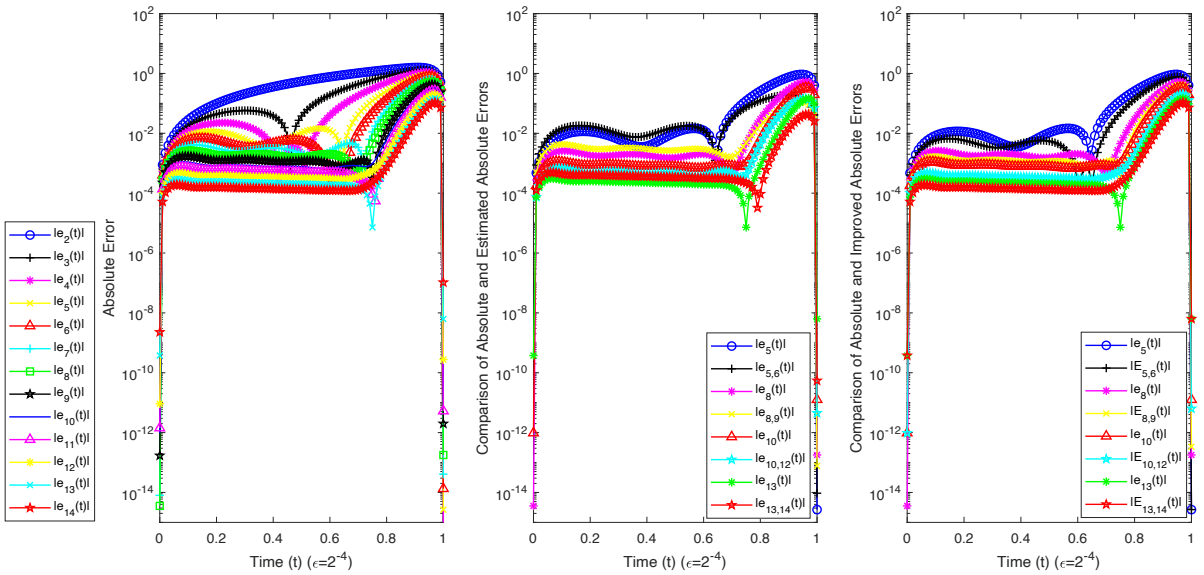

Figure 3. (a) Comparison of the absolute error (actual) functions for $\epsilon = 2^{-6}$ and $N = 2, N = 14$, (b) comparing the estimated absolute error functions for $\epsilon = 2^{-6}$ and some values of (N, M), (c) comparing the improved absolute error functions for $\epsilon = 2^{-6}$ and some values of (N, M).

and ϵ and comparison of the other methods are tabulated for Example 6.2 in Tables 5–8, respectively. And also, the comparisons of the absolute error functions for different values of ϵ are seen in Figures 4 and 5.

Table 5. Maximum absolute errors for some values of (N, M) and ϵ in Example 6.2.

	Absolute errors (actual, estimated, improved)					
ϵ	$\lvert e_4^\epsilon \rvert$	$\lvert e_{15}^\epsilon \rvert$	$\lvert e_{4,5}^\epsilon \rvert$	$\lvert e_{15,16}^\epsilon \rvert$	$\lvert E_{4,5}^\epsilon \rvert$	$\lvert E_{15,16}^\epsilon \rvert$
2^{-2}	3.2898e-1	1.1086e-10	3.7097e-1	1.2094e-10	4.2464e-2	1.0072e-11
2^{-4}	8.9121e-1	7.1073e-3	9.4578e-2	9.5174e-3	8.5807e-1	2.4103e-3
2^{-8}	9.9219e-1	9.9362e-1	1.3949e-1	3.1384e-2	1.0051	9.9224e-1
2^{-12}	9.9951e-1	1.0008	1.3983e-1	3.1343e-2	1.0122	9.9948e-1

Table 6. CPU running time results in Example 6.2.

	$y_4^\epsilon \quad \epsilon = 2^{-2}$	$y_4^\epsilon \quad \epsilon = 2^{-12}$	$y_8^\epsilon \quad \epsilon = 2^{-2}$	$y_8^\epsilon \quad \epsilon = 2^{-12}$	$y_{16}^\epsilon \quad \epsilon = 2^{-2}$	$y_{16}^\epsilon \quad \epsilon = 2^{-12}$
CPU time	1.056 s	1.038 s	1.102 s	1.047 s	1.336 s	1.129 s

Table 7. Time complexity in Example 6.2.

N/EOT$_N$	10	11	12	13	14	15
$\epsilon = 2^{-2}$	0.037	0.153	0.220	0.212	0.150	0.213
$\epsilon = 2^{-4}$	0.111	0.141	0.076	0.094	0.226	0.107
$\epsilon = 2^{-8}$	0.046	0.203	0.109	0.129	0.263	0.053
$\epsilon = 2^{-12}$	0.111	0.202	0.065	0.035	0.389	0.066

Table 8. Comparison of absolute errors in Example 6.2.

	Proposed method		**B-spline method without artificial viscosity**[9]		**B-spline method with artificial viscosity**[9]	
ϵ	$\lvert e_{15}^{\epsilon}\rvert$	$\lvert E_{15,16}^{\epsilon}\rvert$	$n=16$	$n=32$	$n=16$	$n=32$
2^{-2}	1.1086e-10	1.0072e-11	8.572e-4	2.135e-4	1.083e-3	2.716e-4
2^{-4}	7.1073e-3	2.4103e-3	3.023e-2	6.894e-3	7.816e-3	1.978e-3
2^{-8}	9.9362e-1	9.9224e-1	8.039e-1	5.956e-1	5.127e-2	2.272e-2
2^{-12}	1.0008	9.9948e-1	7.967	2.114	5.814e-2	2.980e-2

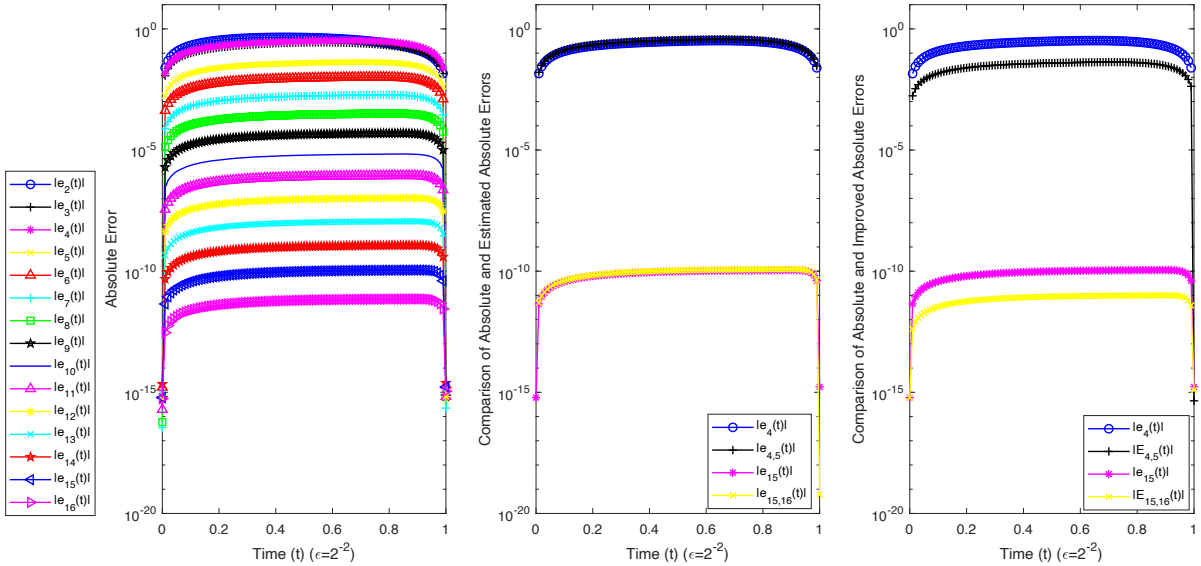

Figure 4. (a) Comparison of the absolute error (actual) functions for $\epsilon = 2^{-2}$ and $N = [2, 16]$, (b) comparing the estimated absolute error functions for $\epsilon = 2^{-2}$ and $(N, M) = (4, 5), (15, 16)$, (c) comparing the improved absolute error functions for $\epsilon = 2^{-2}$ and $(N, M) = (4, 5), (15, 16)$.

Tables 6 and 7 denote the CPU times and the results with the EOT_N for Example 6.2. Table 8 shows comparison of the results obtained by our method and the B-spline collocation method given in [9] of Example 6.2. As can be inferred from the tables, the result obtained by the present method gives better results for small values of N. Absolute error functions for Example 6.2 are compared for some values of N in Figures 4 and 5. It can be clearly inferred that when N is increased, they approach to zero.

Example 6.3. [8] Finally, consider the singularly perturbed differential equation

$$\epsilon y''(t) + (1+t)^2 y'(t) + 2(1+t)y(t) = g(t), \quad t \in [0,1]$$

with the boundry conditions $y(0) = 0$ and $y(1) = e^{\frac{-1}{2}} - e^{\frac{-7}{3\epsilon}}$.

$$a(t) = (1+t)^2,\ b(t) = 2(1+t),\ g(t) = e^{\frac{-t}{2}} \frac{(1+t)(3-t) + \frac{\epsilon}{2}}{2}$$

and the exact solution of the problem is $y(t) = e^{\frac{-t}{2}} + e^{\frac{-t(t^2+3t+3)}{3\epsilon}}$.

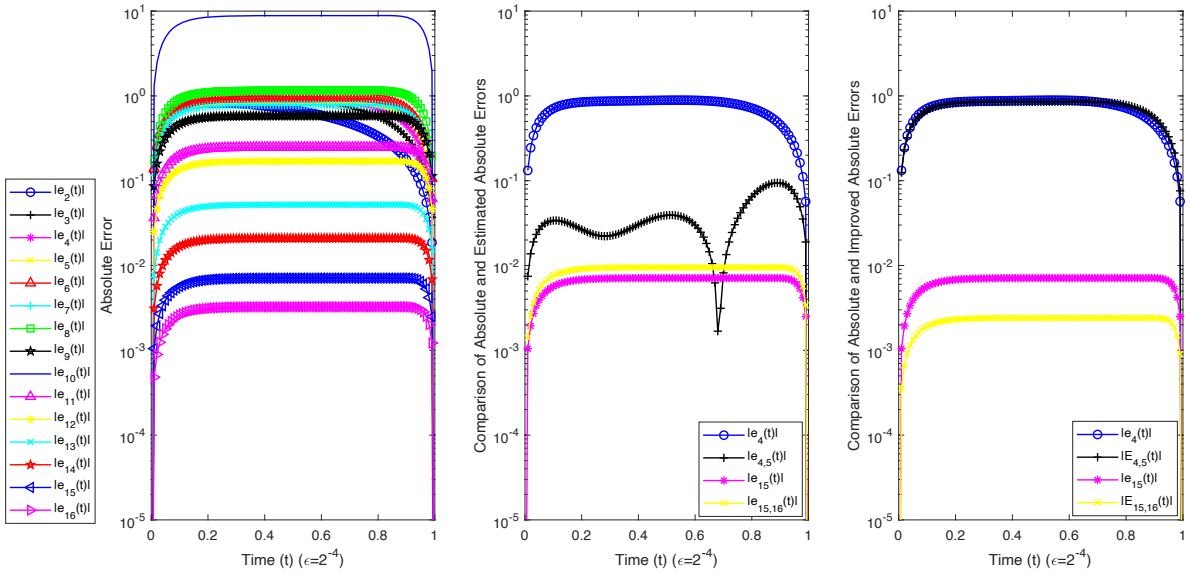

Figure 5. (a) Comparison of the absolute error (actual) functions for $\epsilon = 2^{-4}$ and $N = [2, 16]$, (b) comparing the estimated absolute error functions for $\epsilon = 2^{-4}$ and $(N, M) = (4, 5), (15, 16)$, (c) comparing the improved absolute error functions for $\epsilon = 2^{-4}$ and $(N, M) = (4, 5), (15, 16)$.

The maximum absolute errors, the CPU running time results and the experimental order of timing complexity for various values of (N, M) and ϵ and comparison of the other methods are tabulated for Example 6.3 in Tables 9–12, respectively. And also, the absolute error functions for various values of ϵ can be compared in Figures 6 and 7.

Table 9. Maximum absolute errors for various values of (N, M) and ϵ in Example 6.3.

	Absolute errors (actual, estimated, improved)					
ϵ	$\lvert e_8^\epsilon \rvert$	$\lvert e_{15}^\epsilon \rvert$	$\lvert e_{8,9}^\epsilon \rvert$	$\lvert e_{15,16}^\epsilon \rvert$	$\lvert E_{8,9}^\epsilon \rvert$	$\lvert E_{15,16}^\epsilon \rvert$
2^{-1}	2.8012e-3	8.3114e-8	2.3484e-3	1.3710e-7	4.5281e-4	5.3981e-8
2^{-4}	1.2761	6.0226e-2	3.8216e-1	6.3733e-2	8.9393e-1	3.5075e-3
2^{-8}	9.5528e-1	9.5537e-1	1.5926e-2	7.9790e-3	9.5538e-1	9.5530e-1
10^{-4}	9.8006e-1	9.8006e-1	1.6006e-2	8.0044e-3	9.8031e-1	9.8027e-1
10^{-5}	9.8027e-1	9.8032e-1	1.6008e-2	8.0039e-3	9.8032e-1	9.8027e-1
10^{-6}	9.8027e-1	9.8032e-1	1.6008e-2	8.0168e-3	9.8032e-1	9.8027e-1

Table 10. CPU running time results in Example 6.3.

	$y_8^\epsilon \quad \epsilon = 2^{-1}$	$y_8^\epsilon \quad \epsilon = 10^{-4}$	$y_{12}^\epsilon \quad \epsilon = 2^{-1}$	$y_{12}^\epsilon \quad \epsilon = 10^{-4}$	$y_{16}^\epsilon \quad \epsilon = 2^{-1}$	$y_{16}^\epsilon \quad \epsilon = 10^{-4}$
CPU time	1.078 s	1.090 s	1.129 s	1.133 s	1.184 s	1.211 s

In Tables 10 and 11, the CPU times and the results of the EOT_N are tabulated for Example 6.3. In Table 12, we compare the results of our method with the results of ϵ-Uniformly convergent fitted mesh finite difference methods given in [8] for Example 6.3. As can be inferred from the tables, the result obtained by the present method gives better results than the ones obtained by ϵ-Uniformly convergent fitted mesh finite difference methods.

Table 11. Time complexity in Example 6.3.

N/EOT$_N$	10	11	12	13	14	15
$\epsilon = 2^{-1}$	0.090	0.216	0.231	0.045	0.121	0.333
$\epsilon = 2^{-4}$	0.206	0.088	0.158	0.011	0.232	0.270
$\epsilon = 2^{-8}$	0.238	0.040	0.076	0.071	0.251	0.252
$\epsilon = 10^{-4}$	0.264	0.096	0.114	0.034	0.012	0.116
$\epsilon = 10^{-5}$	0.173	0.294	0.042	0.137	0.049	0.541
$\epsilon = 10^{-6}$	0.045	0.166	0.085	0.148	0.315	0.090

Table 12. Comparison of absolute errors in Example 6.3.

	Proposed method		**Fitted mesh difference method (FMFDM)**[8]		
ϵ	$\lvert e^{\epsilon}_{15}\rvert$	$\lvert E^{\epsilon}_{15,16}\rvert$	$n = 64$	$n = 128$	$n = 256$
2^{-1}	8.3114e-8	5.3981e-8	1.2e-2	2.9e-3	7.3e-4
2^{-4}	6.0226e-2	3.5075e-3	2.9e-1	7.3e-2	1.8e-2
2^{-8}	9.5537e-1	9.5530e-1	3.900	1.200	3.8e-1
10^{-4}	9.8006e-1	9.8027e-1	3.800	1.100	3.7e-1
10^{-5}	9.8032e-1	9.8027e-1	3.800	1.100	3.7e-1
10^{-6}	9.8032e-1	9.8027e-1	3.800	1.100	3.7e-1

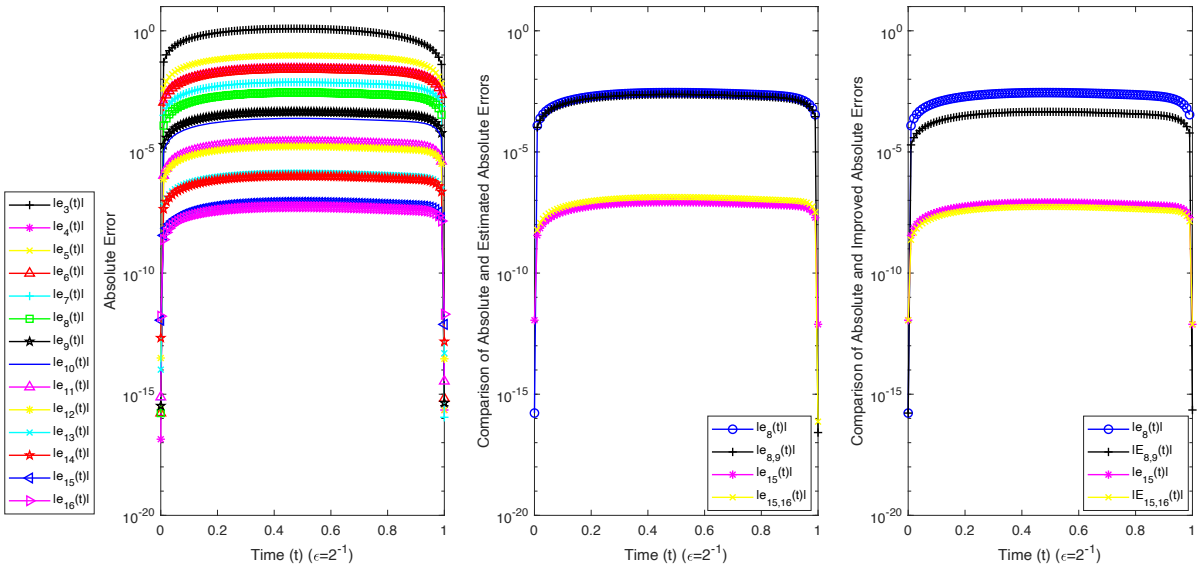

Figure 6. (a) Comparison of the absolute error (actual) functions for $\epsilon = 2^{-1}$ and $N = [3, 16]$, (b) comparing the estimated absolute error functions for $\epsilon = 2^{-1}$ and $(N, M) = (8, 9), (15, 16)$, (c) comparing the improved absolute error functions for $\epsilon = 2^{-1}$ and $(N, M) = (8, 9), (15, 16)$.

7. Conclusion

In this study, Euler matrix-collocation method for solutions of the singular-perturbation problems numerically is presented. Besides, verification of solutions is performed using the defined techniques. Additionally, an estimate of the error is provided with the residual error function. Also an upper bound for the errors is given. And also,

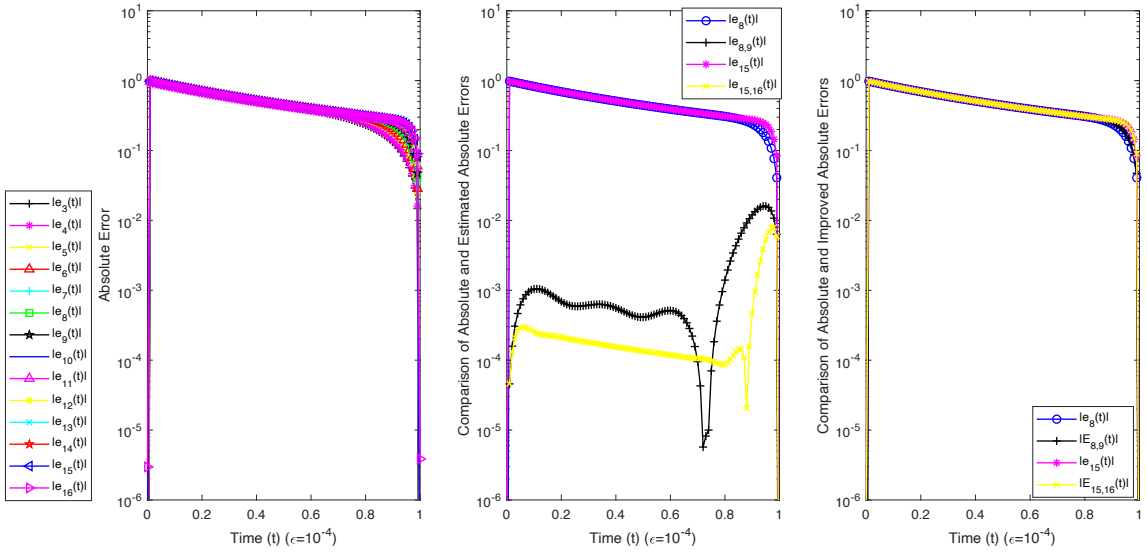

Figure 7. (a) Comparing the actual absolute error functions for $\epsilon = 10^{-4}$ and $N = [3, 16]$, (b) comparing the estimated absolute error functions for $\epsilon = 10^{-4}$ and $(N, M) = (8, 9), (15, 16)$, (c) comparing the improved absolute error functions for $\epsilon = 10^{-4}$ and $(N, M) = (8, 9), (15, 16)$.

the estimated errors are close to the actual errors. Tables and figures indicate that improvement technique is good. Applications of the present method in numerical examples show that the method is very effective. From comparisons with other methods, it is seen that the method gives better results than other methods. Besides all these, it is seen from the CPU times and the results of EOT_N that the problems are solved efficiently and rapidly without the need for detailed procedures. In future, the method can be developed to solve singular perturbed partial differential equations.

References

[1] Chakravarthy PP, Phaneendra K, Reddy YN, A seventh order numerical method for singular perturbation problems. Applied Mathematics and Computation 2007; 186 (1): 860-871. https://doi.org/10.1016/j.amc.2006.08.022

[2] Elmaci D, Savasaneril NB, Dal F, Sezer M. Euler and Taylor poynomials method for solving Volterra type integro-differential equations with nonlinear terms. Journal of Science and Arts 2021; 21 (2): 395-406. https://doi.org/10.46939/J.Sci.Arts-21.2-a07

[3] Elmaci D, Savasaneril NB, Dal F, Sezer M. On the application of Euler's method to linear integro differential equations and comparison with existing methods. Turkish Journal of Mathematics 2022; 46 (1): 99-122. https://doi.org/10.3906/mat-2106-83

[4] Gasparo MG, Maconi M. New initial value method for singularly perturbed boundary value problems. Journal of Optimization Theory and Applications 1989; 63: 213-224. https://doi.org/10.1007/BF00939575

[5] Grau-Sánchez M, Noguera M, Gutiérrez JM. On some computational orders of convergence. Applied Mathematics Letters 2010; 23 (4): 472-478. https://doi.org/10.1016/j.aml.2009.12.006

[6] Habib HM, El-Zahar ER. An algorithm for solving singular perturbation problems with mechanization. Applied Mathematics and Computation 2007; 188 (1): 286-302. https://doi.org/10.1016/j.amc.2006.09.132

[7] Johnson RS. Singular perturbation theory: mathematical and analytical techniques with applications to engineering. Springer Science Business Media, 2012.

[8] Kadalbajoo MM, Patidar KC. ϵ-Uniformly convergent fitted mesh finite difference methods for general singular perturbation problems. Applied Mathematics and Computation 2006; 179 (1): 248-266. https://doi.org/10.1016/j.amc.2005.11.096

[9] Kadalbajoo MK, Parora P. B-spline collocation method for the singular - perturbation problem using artificial viscosity. Computers and Mathematics with Applications 2009; 57 (4): 650-663. https://doi.org/10.1016/j.camwa.2008.09.008

[10] Kadalbajoo MK, Parora P. B-splines with artificial viscosity for solving singularly perturbed boundary value problems. Mathematical and Computer Modelling 2010; 52 (5-6): 654-666. https://doi.org/10.1016/j.mcm.2010.04.012

[11] Kevorkian J, Cole JD. Perturbation Methods in Applied Mathematics. Springer-Verlag: New York, 1981.

[12] Kumar M. A second order spline finite difference method for singular two-point boundary value problems. Applied Mathematics and Computation 2003; 142 (2-3): 283-290. https://doi.org/10.1016/S0096-3003(02)00302-8

[13] Natesan S, Ramanujam N. A computational method for solving singularly perturbed turning point problems exhibiting twin boundary layers. Applied Mathematics and Computation 1998; 93 (2-3): 259-275. https://doi.org/10.1016/S0096-3003(97)10056-X

[14] Odibat ZM, Shawagfeh NT. Generalized Taylor's formula. Applied Mathematics and Computation 2007; 186 (1): 286-293. https://doi.org/10.1016/j.amc.2006.07.102

[15] Yüzbaşı Ş. A collocation method based on the Bessel functions of the first kind for singular perturbated differential equations and residual correction. Mathematical Methods in the Applied Sciences 2015; 38 (14), 3033-3042. https://doi.org/10.1002/mma.3278

[16] Yüzbaşı Ş. A Laguerre approach for the solutions of singular perturbated differential equations. International Journal of Computational Methods 2017; 14 (04). https://doi.org/10.1142/S0219876217500347