Solving Singular Eigenvalue Problem Using Semi-Analytic Technique

Luma. N. M. Tawfiq and Hassan. Z. Mjthap

Department of Mathematics, College of Education Ibn Al-Haitham, Baghdad University

*Author to whom correspondence should be addressed; E-Mail:drlumanaji@yahoo.com

Article history: Received 6 March 2013, Received in revised form 17 June 2013, Accepted 18 June 2013, Published 19 June 2013.

Abstract: In this paper, we present a semi-analytic technique to solve the singular eigenvalue problem using osculator interpolation polynomial. The technique finds eigenvalue and corresponding nonzero eigenvector which represent the solution of equation in a certain domain. Numerical example is presented, which confirm the theoretical predictions and a comparation between suggested technique and other methods.

Keywords: ODE, BVP, Interpolation, Eigenvalue.

Mathematics Subject Classification (2000): 65L10, 65D30, 26D10, 41A05, 30E25, 45Exx, 34Bxx, 41A05, 45C05, 141A80.

1. Introduction

The present paper is concerned with a semi-analytic technique to solve singular eigenvalue problems using osculator interpolation polynomial. The eigenvalue problem (EP) [1] involves finding an eigenvalue \( \lambda \) and corresponding nonzero eigenvector that satisfy the solution of the problem.

The eigenvalue problems can be used in a variety of problems in science and engineering. For example, quadratic eigenvalue problems arise in oscillation analysis with damping [2], [3] and stability problems in fluid dynamics [4], and the three-dimensional (3D) Schrödinger equation can result in a cubic eigenvalue problem [5].

Copyright © 2013 by Modern Scientific Press Company, Florida, USA
Similarly, the study of higher-order systems of differential equations leads to a matrix polynomial of degree greater than one [5]. However, its applications are more complicated than standard and generalized eigenvalue problems. One reason is in the difficulty in solving the EPs. Polynomial eigenvalue problems are typically solved by linearization [6], [7], which promotes the $k$-th order $n \times n$ matrix polynomial into the larger $k n \times k n$ linear eigenvalue problem. Other methods, such as Arnoldi shift and invert strategy [8], can be used when several eigenvalues are desired. A disadvantage of the shift-and-invert Arnoldi methods is that a change of the shift parameter requires a new Krylov subspace to be built. Another approach is a direct solution obtained by means of the Jacobi-Davidson method [9], although this method has been investigated far less extensively.

In the present paper, we propose a series solution of singular eigenvalue problems by means of the oscillatator interpolation polynomial. The proposed method enables us to obtain the eigenvalue and corresponding nonzero eigenvector of the $2^{nd}$ order singular boundary value problem (SBVP).

2. Osculator Interpolation Polynomial

In this paper we use two-point osculatory interpolation polynomial; essentially this is a generalization of interpolation using Taylor polynomials. The idea is to approximate a function $y$ by a polynomial $P$ in which values of $y$ and any number of its derivatives at given points are fitted by the corresponding function values and derivatives of $P$ [10].

We are particularly concerned with fitting function values and derivatives at the two end points of a finite interval, say $[0, 1]$, i.e., $P^{(j)}(x_i) = f^{(j)}(x_i), \; j = 0, \ldots, n, \; x_i = 0, 1$, where a useful and succinct way of writing osculatory interpolant $P_{2n+1}$ of degree $2n + 1$ was given for example by Phillips [11] as:

$$P_{2n+1}(x) = \sum_{j=0}^{n} \left\{ y^{(j)}(0) q_{j}(x) + (-1)^{j} y^{(j)}(1) q_{j}(1-x) \right\}, \quad (1)$$

$$q_{j}(x) = (x^j / j!)(1-x)^{n+1} \sum_{s=0}^{n-j} \begin{pmatrix} n + s \\ s \end{pmatrix} x^s = Q_{j}(x) / j! \quad (2)$$

so that (1) with (2) satisfies:

$$y^{(j)}(0) = P_{2n+1}^{(j)}(0), \quad y^{(j)}(1) = P_{2n+1}^{(j)}(1), \quad j = 0, 1, 2, \ldots, n.$$

implying that $P_{2n+1}$ agrees with the appropriately truncated Taylor series for $y$ about $x = 0$ and $x = 1$. We observe that (1) can be written directly in terms of the Taylor coefficients $a_i$ and $b_i$ about $x = 0$ and $x = 1$ respectively, as:
\[ P_{2n+1}(x) = \sum_{j=0}^{n} \{ a_j Q_j(x) + (-1)^j b_j Q_j(1-x) \}, \] (3)

3. Singular Boundary Value Problem

The general form of the 2\(^{nd}\) order two point boundary value problem (TPBVP) is:

\[ y'' + P(x)y' + Q(x)y = 0 , \quad a \leq x \leq b \] (4)

\[ y(a) = A \quad \text{and} \quad y(b) = B, \quad \text{where} \ A, B \in R \]

There are two types of a point \( x_0 \in [0,1] \): Ordinary point and Singular point.

A function \( y(x) \) is analytic at \( x_0 \) if it has a power series expansion at \( x_0 \) that converges to \( y(x) \) on an open interval containing \( x_0 \). A point \( x_0 \) is an ordinary point of the ODE (4), if the functions \( P(x) \) and \( Q(x) \) are analytic at \( x_0 \). Otherwise \( x_0 \) is a singular point of the ODE. On the other hand if \( P(x) \) or \( Q(x) \) are not analytic at \( x_0 \) then \( x_0 \) is said to be a singular point [12], [13].

There is, at present, no theoretical work justifying numerical methods for solving problems with irregular singular points. The main practical occurrence of such problems seems to be semi-analytic technique [14].

4. Solution of Second Order Eigenvalue Problem

In this section, we suggest a semi-analytic technique which is based on osculatory interpolating polynomials \( P_{2n+1} \) and Taylor series expansion to solve 2\(^{nd}\) order eigenvalue Problem. A general form of 2\(^{nd}\) order BVP's is:

\[ y''(x) = \lambda f(x, y, y'), 0 \leq x \leq 1 \] (5a)

Subject to the boundary condition (BC):

In the case Dirichlet BC: \( y(0)= A, \ y(1) = B, \ \text{where} \ A, B \in R \) (5b)

In the case Neumann BC: \( y'(0)= A, \ y'(1) = B, \ \text{where} \ A, B \in R \) (5c)

In the case Cauchy or mixed BC: \( y(0)= A, \ y'(1) = B, \ \text{where} \ A, B \in R \) (5d)

Or \( y'(0)= A, \ y(1) = B, \ \text{where} \ A, B \in R \) (5d)

where \( f \) are in general nonlinear functions of their arguments.

Now, to solve the problem by suggested method doing the following steps:

**Step one:**
Evaluate Taylor series of \( y(x) \) about \( x = 0 \):
\[ y = \sum_{i=0}^{\infty} a_i x^i = a_0 + a_1 x + \sum_{i=2}^{\infty} a_i x^i \]  
(6)

where \( y(0) = a_0, y'(0) = a_1, y''(0) / 2! = a_2, \ldots, y^{(i)}(0) / i! = a_i, i = 3, 4, \ldots \)

And evaluate Taylor series of \( y(x) \) about \( x = 1 \):

\[ y = \sum_{i=0}^{\infty} b_i (x-1)^i = b_0 + b_1 (x-1) + \sum_{i=2}^{\infty} b_i (x-1)^i \]  
(7)

where \( y(1) = b_0, y'(1) = b_1, y''(1) / 2! = b_2, \ldots, y^{(i)}(1) / i! = b_i, i = 3, 4, \ldots \)

**Step two:**

Insert the series form (6) into equation (5a) and put \( x = 0 \), then equate the coefficients of powers of \( x \) to obtain \( a_2 \).

Insert the series form (7) into equation (5a) and put \( x = 1 \), then equate the coefficients of powers of \( (x-1) \) to obtain \( b_2 \).

**Step three:**

Derive equation (5a) with respect to \( x \), to get new form of equation say (8) as follows:

\[ y'''(x) = \lambda \frac{df(x, y, y')}{dx} \]  
(8)

Then, insert the series form (6) into equation (8) and put \( x = 0 \) and equate the coefficients of powers of \( x \) to obtain \( a_3 \), again insert the series form (7) into equation (8) and put \( x = 1 \), then equate the coefficients of powers of \( (x-1) \) to obtain \( b_3 \).

**Step four:**

Iterate the above process many times to obtain \( a_4, b_4 \) then \( a_5, b_5 \) and so on, that is, to get \( a_i \) and \( b_i \) for all \( i \geq 2 \), the resulting equations can be solved using MATLAB version 7.10, to obtain \( a_i \) and \( b_i \) for all \( i \geq 2 \).

**Step five:**

The notation implies that the coefficients depend only on the indicated unknowns \( a_0, a_1, b_0, b_1 \), and \( \lambda \), use the BC to get two coefficients from these, therefore, we have only two unknown coefficients and \( \lambda \). Now, we can construct two point osculatory interpolating polynomial \( P_{2n+1}(x) \) by insert these coefficients \( ( a_b, b_b ) \) into equation (3).

**Step six:**

To find the unknowns coefficients integrate equation (5a) on \([0, x]\) to obtain:

\[ y'(x) - y'(0) - \lambda \int_{0}^{x} f(x, y, y') \, dx = 0 \]  
(9a)
and again integrate equation (9a) on \([0, x]\) to obtain:

\[
y(x) - y(0) - y'(0) x - \lambda \int_{0}^{x} (1-x) f(x, y, y') \, dx = 0
\]  

(9b)

**Step seven:**

Putting \(x = 1\) in equations (9) to get:

\[
b_1 - a_1 - \lambda \int_{0}^{1} f(x, y, y') \, dx = 0
\]  

(10a)

and

\[
b_0 - a_0 - a_1 - \lambda \int_{0}^{1} (1-x) f(x, y, y') \, dx = 0
\]  

(10b)

**Step eight:**

Use \(P_{2n+1}(x)\) which constructed in step five as a replacement of \(y(x)\), we see that equations (10) have only two unknown coefficients \(a_0, a_1, b_0, b_1\) and \(\lambda\). If the BC is Dirichlet boundary condition, that is, we have \(a_0\) and \(b_0\), then equations (10) has two unknown coefficients \(a_1, b_1\) and \(\lambda\). If the BC is Neumann, that is, we have \(a_1\) and \(b_1\), then equations (10) has two unknown coefficients \(a_0, b_0\) and \(\lambda\). Finally, if the BC is Cauchy boundary condition or mixed condition, i.e., we have \(a_0\) and \(b_1\) or \(a_1\) and \(b_0\), then equations (10) has two unknown coefficients \(a_1, b_0\) or \(a_0, b_1\) and \(\lambda\).

**Step nine:**

In the case Dirichlet BC, we have:

\[
F(a_1, b_1, \lambda ) = b_1 - a_1 - \lambda \int_{0}^{1} f(x, y, y') \, dx = 0
\]  

(11a)

\[
G(a_1, b_1, \lambda ) = b_0 - a_0 - a_1 - \lambda \int_{0}^{1} (1-x) f(x, y, y') \, dx = 0
\]  

(11b)

\[
(\partial F/\partial a_1)(\partial G/\partial b_1) - (\partial F/\partial b_1)(\partial G/\partial a_1) = 0
\]  

(11c)

In the case Neumann BC, we have:

\[
F(a_0, b_0, \lambda ) = b_1 - a_1 - \lambda \int_{0}^{1} f(x, y, y') \, dx = 0
\]  

(12a)

\[
G(a_0, b_0, \lambda ) = b_0 - a_0 - a_1 - \lambda \int_{0}^{1} (1-x) f(x, y, y') \, dx = 0
\]  

(12b)
\[
(\partial F/\partial a_0)(\partial G/\partial b_0) - (\partial F/\partial b_0)(\partial G/\partial a_0) = 0 \tag{12c}
\]

In the case mixed BC, we have:

\[
F(a_1, b_0, \lambda) = b_1 - a_1 - \lambda \int_0^1 f(x, y, y') \, dx = 0 \tag{13a}
\]

\[
G(a_1, b_0, \lambda) = b_0 - a_0 - a_1 - \lambda \int_0^1 (1-x)f(x, y, y') \, dx = 0 \tag{13b}
\]

\[
(\partial F/\partial a_1)(\partial G/\partial b_0) - (\partial F/\partial b_0)(\partial G/\partial a_1) = 0 \tag{13c}
\]

Or

\[
F(a_0, b_1, \lambda) = b_1 - a_1 - \lambda \int_0^1 f(x, y, y') \, dx = 0 \tag{14a}
\]

\[
G(a_0, b_1, \lambda) = b_0 - a_0 - a_1 - \lambda \int_0^1 (1-x)f(x, y, y') \, dx = 0 \tag{14b}
\]

\[
(\partial F/\partial a_0)(\partial G/\partial b_1) - (\partial F/\partial b_1)(\partial G/\partial a_0) = 0 \tag{14c}
\]

So, we can find these coefficients by solving the system of algebraic equations (11) or (12) or (13) or (14) using MATLAB, so insert the value of the unknown coefficients into equation (3), thus equation (3) represent the solution of the problem.

**Note:** Extensive computations have shown that this generally provides a more accurate polynomial representation for a given \( n \).

**5. Examples**

In this section, we investigate the theory using example of singular eigenvalue problem. The algorithm was implemented in MATLAB 7.10.

The bvp4c solver of MATLAB has been modified accordingly so that it can solve some class of singular eigenvalue problem as effectively as it previously solved eigenvalue problem.

Also, we report a more conventional measure of the error, namely the error relative to the larger of the magnitude of the solution component and taking advantage of having a continuous approximate solution, we report the largest error found at 10 equally spaced points in \([0, 1]\). This problem arises in a study of heat and mass transfer in a porous spherical catalyst with a first order reaction. There is a singular coefficient arising from the reduction of a partial differential equation to an ODE by symmetry [13]. The eigenvalue problem is:
There are two physical parameters $\alpha = 40$ and $\beta = 0.2$.

The BC are $y(1) = 1$ and the symmetry condition $y'(0) = 0$ (mixed BC).

Now, we solve this problem by suggested method. Here equations (14) become:

\[
 F(a_0, b_1, \lambda) = a_0 - 1 - b_1 + \lambda^2 \int_0^{a(1-y)e^{1+\beta(1-y)}} x \, ye^{1+\beta(1-y)} \, dx = 0 \tag{15a}
\]

\[
 G(a_0, b_1, \lambda) = a_0 - 1 + \lambda^2 \int_0^{(1-x)f(x, y, y') \, dx = 0} \tag{15b}
\]

\[
 (\partial F/\partial a_0)(\partial G/\partial b_1) - (\partial F/\partial b_1)(\partial G/\partial a_0) = 0 \tag{15c}
\]

Now, we have to solve equations (15) for the unknowns $a_0, b_1, \lambda$ using MATLAB, then we have $a_0 = 0.64, b_1 = 0, \lambda = 0.6$.

Then from equation (3) we have:

\[
 P_9 = 18.63018391074087 \, x^9 - 81.86548669822514 \, x^8 + 136.0606573968419 \, x^7 - 101.7163034006953 \, x^6 + 29.5919367027597 \, x^5 - 1.076125341642182 \, x^4 + 0.7351374287263839 \, x^3 + 0.64
\]

For more details, table 1 gives the results for different nodes in the domain, for $n = 4$, i.e., $P_9$ we note that there is no difficulty in taking higher values of $n$ if we wished to refine this value and figure 1 illustrate suggested method for $P_9$.

![Figure 1: The suggested solution $P_9$ of example](image)

Copyright © 2013 by Modern Scientific Press Company, Florida, USA
Table 1: The suggested solution for \( n = 4 \), i.e., \( P_9 \) of Example

<table>
<thead>
<tr>
<th>( a_0 )</th>
<th>0.64</th>
</tr>
</thead>
<tbody>
<tr>
<td>( b_1 )</td>
<td>0</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>0.6</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>40</td>
</tr>
<tr>
<td>( \beta )</td>
<td>0.2</td>
</tr>
<tr>
<td>( x_i )</td>
<td>( P_9 )</td>
</tr>
<tr>
<td>0</td>
<td>0.6400000000000000</td>
</tr>
<tr>
<td>0.1</td>
<td>0.647450770857784</td>
</tr>
<tr>
<td>0.2</td>
<td>0.672184812352611</td>
</tr>
<tr>
<td>0.3</td>
<td>0.719954943427546</td>
</tr>
<tr>
<td>0.4</td>
<td>0.790618839607907</td>
</tr>
<tr>
<td>0.5</td>
<td>0.871531211116071</td>
</tr>
<tr>
<td>0.6</td>
<td>0.942127912371548</td>
</tr>
<tr>
<td>0.7</td>
<td>0.986127990009496</td>
</tr>
<tr>
<td>0.8</td>
<td>1.001813596864667</td>
</tr>
<tr>
<td>0.9</td>
<td>1.001642140819012</td>
</tr>
<tr>
<td>1</td>
<td>1.000000000000000</td>
</tr>
</tbody>
</table>

Kubiček et al [15, 13] solved this problem by a collection method (there are three solutions) assembled by consider a range of parameter values: \( \lambda, \beta, \alpha \), such that, the values \( \lambda = 0.6, \alpha = 0.1, \beta = 0.2 \) used in ex6bvp.m lead to three solutions that are displayed in figure 2. Table 2 compares the solutions in [13] at the origin to values reported in [15].

Another solution of this problem gave in [16] using collocation method with different code in MATLAB and Fortran given in table 3 and figure 3.

Table 2: Computed \( y(0) \) for three solutions of the problem

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9071</td>
<td>0.9070</td>
</tr>
<tr>
<td>0.3637</td>
<td>0.3639</td>
</tr>
<tr>
<td>0.0001</td>
<td>0.0001</td>
</tr>
</tbody>
</table>
Table 3: Comparisons for different code of solution in [16] with TOL = 10^{-7}

<table>
<thead>
<tr>
<th></th>
<th>bvp4c</th>
<th>CW-4</th>
<th>CW-6</th>
<th>sbvp4</th>
<th>sbvp4g</th>
<th>sbvp6</th>
<th>sbvp6g</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>205</td>
<td>41</td>
<td>21</td>
<td>57</td>
<td>57</td>
<td>22</td>
<td>15</td>
</tr>
<tr>
<td>f_count</td>
<td>6856</td>
<td>1080</td>
<td>840</td>
<td>6051</td>
<td>3699</td>
<td>3741</td>
<td>1431</td>
</tr>
</tbody>
</table>

Where:

– bvp4c (MATLAB 6.0 routine): which is based on collocation at three Lobatto points [16]. This is a method of order 4 for regular problems.

– COLNEW (Fortran 90 code): The basic method here is collocation at Gaussian points, we chose the polynomial degrees $m = 4$ (CW-4) and $m = 6$ (CW-6), which results in (super convergent) methods of orders 8 and 12 respectively (for regular problems).

– sbvp is used with equidistant (sbvp4 and sbvp6) and Gaussian (sbvp4g and sbvp6g) collocation points and polynomial degrees 4 and 6 respectively.

Although for reasons mentioned here, the comparison of all three codes is difficult. Table 3 shows the number of mesh points ($N$) and the number of function evaluations ($f_{count}$) that the different solvers required to reach tolerance (TOL).
6. Conclusions

In the present paper, we have proposed a semi-analytic technique to solve singular eigenvalue problems. By using osculator interpolation, the result shown that the Semi - Analytic technique can be used successfully for finding the solution of nonlinear singular eigenvalue problem with boundary conditions of second order with singular point of first, second and third kind. It may be concluded that this technique is a very powerful and efficient in finding highly accurate solutions for a large class of differential equations.

Finally, the bvp4c solver of MATLAB has been modified accordingly so that it can solve some class of singular eigenvalue problem with boundary conditions as effectively as it previously solved non-singular BVP.

References


